University of Computer Studies (Mandalay)

Department of Advanced Science and Technology

Ministry of Science and Technology

The Republic of the Union of Myanmar

JOURNAL OF INFORMATION TECHNOLOGY, RESEARCH AND INNOVATION

June, 2025

JiTRi 2025

JOURNAL OF INFORMATION TECHNOLOGY, RESEARCH AND INNOVATION

June, 2025

Organized by

University of Computer Studies

(Mandalay)

Volume-5, Issue-1

JOURNAL OF INFORMATION TECHNOLOGY, RESEARCH AND INNOVATION

JITRI, 2025

Vol-5, Issue-1

UNIVERSITY OF COMPUTER STUDIES (MANDALAY)

Journal of Information Technology, Research and Innovation (JITRI), 2025 Committee

University of Computer Studies (Mandalay)

Acknowledgements for Reviewers and Organizing members of JITRI

The Journal of Information Technology, Research and Innovation (JITRI) 2025 technical program committee would like to express the deepest appreciation to the external reviewers and organizing members for collaborating to publish this journal successfully. JITRI (2025) is published as the eighth time in order to undertake research in their respective fields of studies. Finally, we would like to thank all the authors who had submitted their research papers by making their great efforts in the Journal of Information Technology, Research and Innovation (JITRI), 2025, Vol-5, Issue-1.

Table of Contents

Sr. No.	Title	Page
1.	Biometric Iris Recognition System Using Deep Convolutional Neural Networks for Improved Security *Yu Thin Zar Myo, Atar Mon , Kyi Pyar Zaw , and Hnin Hnin Khaing*	1-12
2.	Performance Analysis of Active Noise Cancellation Using Adaptive Filter Algorithms for Transportation Environment *Myat Min Khant ,Atar Mon,Soe Nandar Lin, and Aye Theingi Oo*	13-21
3.	Autonomous Mobile Robot Based on Robot Operating System2 (ROS2) with Gazebo *Thet Hsu Wai , Atar Mon , Myo Su Su Theint, and Soe Nandar Lin*	22-31
4.	Exploring Transformer Models for Machine Translation of Myanmar and Dawei Languages *Sint Sint Shein*	32-37
5.	Characterization of Dragon Fruit Dye as a Light-Harvesting Material in Natural Dye-Sensitized Solar Cells *Chit Myo Naing, and Zan Mo Mo Aung*	38-43
6.	Speaker Recognition System using Spectrogram Image Feature and Convolutional Neural Networks *Thein Htay Zaw, and San Lin Aung*	44-50
7.	Deep-Learning Based Grape Leaf Disease Detection Using EfficientNet-B7 Enhanced with Convolutional Block Attention Module *Moe Phyu Phyu Khaing, and Phyu Pyar Moe*	51-60
8.	A CNN-Based Framework for Detecting and Classifying Student Emotions via Facial Expressions to Enhance Classroom Interaction *Phyu Pyar Moe, and Moe Phyu Phyu Khaing*	61-66
9.	Myanmar Language Question Answering System Using GRU, LSTM and mBART-large Model *Naing Naing Khin, Yi Mon Shwe Sin, and Win Lei Kay Khine*	67-74

10.	Predicting Terrorist Activities Around The World Applying Convolutional Neural Network And Shallow Neural Network *Htet Htet Wah Lwin, and Thandar Aung*	75-83
11.	Image Captioning Using DenseNet Feature Extraction and LSTM-based Sequence Generation *Hsu Myat Htet, and Yu Ya Win*	84-92
12.	Fuzzy Decision Support System for Evaluating the Student's Performance in Online Learning Platform *Phyu Pyar Moe, and Sergey Andreevich Lupin*	93-101
13.	Weather Forecasting Using Long Short-Term Memory (LSTM) *May Muyar Han, and Thin Thin Naing*	102-107
14.	Grammar Error Correction System Using a Sequence-to-Sequence Model with An Attention Mechanism *Nandar Win Khin, and Thandar Aung*	108-113
15.	Fake News Detection Using GloVe-LSTM Model *Sandar Win, and Yu Ya Win*	114-120
16.	Intelligent Human Detection and Alert System with Deep Learning Model *Kyal Sin Lin, and Thein Htay Zaw*	121-128
17.	Image Reflection Removal Using Deep Convolutional Encoder-Decoder Network *Aye Su Khaing and Mya Thidar Kyaw*	129-136
18.	Enhancing Lossless Data Compression Efficiency for DNA Sequences and Audio *Zaw Win Myat, and Thin Thin Naing*	137-145
19.	Detection of Chest X Rays for Pneumonia Using Resnet50 *Myint Zu Hmwe, and Thein Htay Zaw*	146-153

20. Color Following Robot Using Kinematic Theory and PID Control 154-161
Htet Wai Lin, and May Mya Aung

BIOMETRIC IRIS RECOGNITION SYSTEM USING DEEP CONVOLUTIONAL NEURAL NETWORKS FOR IMPROVED SECURITY

Yu Thin Zar Myo[1], Atar Mon[2] , Kyi Pyar Zaw[3] , and Hnin Hnin Khaing[4]

[1,2,3,4]Faculty of Electronics Engineering, University of Technology

(Yatanarpon Cyber City)

yuthinzarmyo16797@gmail.com, atarmon@gmail.com,kyipyarzaw89@gmail.com, hninhninkhaing.ec@gmail.com

ABSTRACT

Iris recognition is a very safe biometric identification technique which makes unique and complex patterns of the iris that don't change over a person's life time. To address these challenge, this paper propsed iris recognition system based on Deep CNNs, VGG16 and Inception V3 for feature extraction and Support Vector Machine (SVM) for classification are used, respectively. To boost the effectiveness of recogntion with image processing methods, Histogram Equalization and Hough Circle Transform. The proposed technique has been effectively implemented on iris images from the MMU dataset which includes totalling 460 images of 46 persons. The dataset is split into 80% for taining and 20% for testing. The system's models achieve accuracy values of 96% from VGG16 and 89% from InceptionV3. The proposed system demonstrates an advantage of the accuries and realibilites for various apllications such as immagration, law enformancement and access control. However, the system's performance may be impacted by subjects wearing eyeglasses due to reflections and obstructions. This limitation highlights potential for improvement using techniques like polarization filtering.

Keywords

Bimortric, Iris Recognition, Histogram Equalization, Hough Circle Transform, CNN models and SVM

1. INTRODUCTION

A term 'Biometric', a body measurement and calculation of human characteristics and features which can accomplish security for authentication and recognition each individual has traits and attributes. Biometric systems can be divided into two major categories according to their measured: physical and behavioural characteristic of a living being. e.g., fingerprint, voice, face, signature, key strokes, speech and so on.

- Physical characteristics contains DNA, face, fingerprint, finger vein, scent, iris and so on.
- Behavioural characteristics of biometrics are signature, gait, voice, handwriting and keystrokes.

In general, physical biometrics systems are more accurate and have a greater price than behavioural biometrics systems. Figure.1 shows the commonly used biometrics characteristics [1]. Biometrics can now be utilized in applications were establishing or verifying an individual's identity is needed because to the rapid advancements in information technology. Now a day, the incresement of security demand in the daily life toward the digitization of a biometric-based, intelligent, and trustworthy person identification system. Keys, PIN codes, ID cards passwords are used in traditional systems. These systems are often susceptible to being broken down, forgotten or stolen. Unlike traditional methods, biometric

system offers the convenience of eliminating the need to carry or remember these things. Therefore, there is an urgent need for biometric identification systems to identify people independing on their possessions or memories.

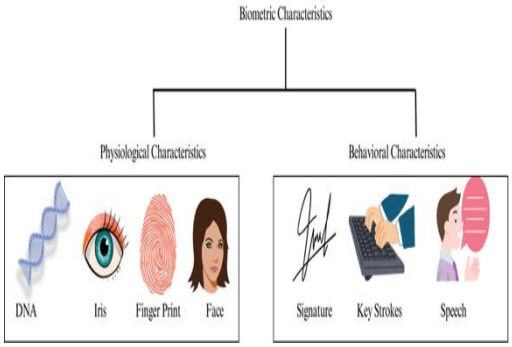


Figure 1. Commonly used Biometric Characteristics

Biometric technology has been used for accurate recognition and verification in a variety of applications, including border controls, airport passenger control, database access, restricted area access, health care and financial and medical services. The current banking services' credit and debit cards, as well as biometric payments, will become significantly safer, quicker, and simpler. As a result, biometric systems have slowly substituted the traditional authentication methods by utilizing users' unique traits in high security fields[2]. Table 1 shows the comparison of qualities of difference biometric identifiers [4].

Table 1. Comparison of qualities of Difference Biometric Identifiers.

Biometric identifier	Uniqueness	Permanence	Universality	Measurability	Collectability	Performance	Acceptability
DNA	H	H	H	L	L	H	H
Iris	H	H	H	M	H	H	M
Finger Print	H	H	M	H	M	M	H
Face	M	M	H	M	H	L	H
Signature	H	L	L	M	H	M	H
Key Strokes	L	L	L	L	M	L	L
Voice	L	L	M	M	M	L	H

H = High, M = Medium, L = Low

Moreover, as biometric systems continue to advance, they are also being integrated into mobile devices, smart home systems, and e-commerce platforms, expanding their accessibility and practical usage. Among these, biometric systems have been essential for security systems with iris recognition being one of the most accurate method due to the iris's unique and stable texture. The human frontal eye includes iris, pupil and sclera. Figure 2. illustrates the human eye.

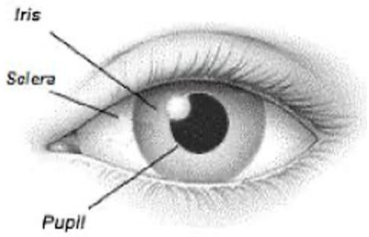


Figure 2. The Human's Eye

The human iris is the coloured, ring-shaped portion of the pupil's surroundings. It also plays a vital role in controlling how much light enters the eye. Human irises are distinctive and appropriate for biometric measurement because of their remarkable structure[3]. The pattern in the iris, including crypts, ridges, furrows, rings and freckles are randomly formed during fatal development. These patterns are unique for every individual, including identical twins, has unique iris patterns that remain consistent over their lives. When compared to other biometric technologies like face, palm print and voice, the distinctiveness of the iris pattern makes it more stable and dependable for identification. That is why iris recognition is so powerful for secure identity verification. Due to its uniqueness, stability over time, and relative ease of access and one of the best characteristics for

verification is the iris pattern when compared to other biometrics as shown in Table 1[4].

The field of computer science known as Artificial Intelligence (AI) is concerned with building devices or software that are capable of carrying out operations that normally call for human intelligence. Learning problem and solving, making decision, comprehending natural language, and identifying patterns like sounds or visuals are some of these tasks. AI works using techniques such as machine learning, deep learning, and neural network. The development of artificial intelligence has significantly improved machine learning's capacity to handle vast data sets, train large data sets effectively and enable autonomous learning in models. Deep learning is an important branch of research and development in the field of machine learning. The ability of deep learning to automatically combine basic features into more complex ones and use these feature combinations to solve issues is one of its primary challenges. Figure. 3 illustrates the artificial intelligence concerns with machine learning and deep learning.

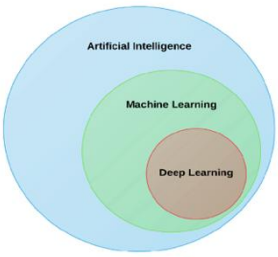


Figure 3. The Artificial Intelligence Concerns with Machine Learning and Deep Learning

Generally, the iris recognition process is complex, including iris image assessment, localization, normalization, mask estimation, segmentation, converting gray scale, feature extraction, matching and rough classification. In additional, the extraction of iris features is usually used artifical.

In this paper, an effective iris recognition system relying on Deep CNN models, VGG16 and inception v3. The iris dataset provided by Iris database for the Biometric Attendance System at Multimedia University is used to train the models and makes the model self-study to identify the iris in the iris database of the human eye. For normalization, histogram equalization is used to enhance contrast and normalizes illumination to standardize the iris image. Hough circle transform to identify and separate the iris from the image's other components. To improve the performance of iris recognition, SVM is utilized for classification. This study evaluates the robustness of iris recognition models and compare their accuracies. The structure of the paper is outline as follows: section 2 presents related work, section 3 introduces the methodology of the system, section 4 imparts the results and compares the models and section 5 concludes the paper.

2. RELATED WORK

Previously, iris recognition system had been developed by numerous methods. This section reviews some of these approaches.

Traditional techniques applied for automatic iris recognition process has been imparted by Daugman and Downing. John Daugman [6] introduced this method which consisted of 2D Gabor wavelet filters for feature extraction and generated Iris Code as a binary representation of iris texture. Hamming distance is used for matching. LiMa [7] focused on texture analysis multi-scale analysis like 1D and 2D wavelets. Segmented iris using edge detection and circular Hough transform. Alaslni et al. [8] introduced an iris recognition system utilizing Transfer Learning, where the authors utilized the pre-trained VGG-16 model for feature extraction and classification. Arora et al.[9] presented a novel CNN iris recognition model. Their technique locates the iris using the Circular Hough Transform, then extracts features from the iris region using a proprietary CNN. A Softmax classifier then does classification. This model achieved a noteworthy accuracy of 98% when tested solely on the IITD dataset. Alaslani et al.

[10] introduced a method for iris recognition that segmented iris pictures using circular Hough Transforms and Canny edge detection. The authors used a pre-trained CNN to extract features and the rubber sheet model for the normalization step. A Support Vector Machine (SVM) handled the classification phase. Mashael et al. [11] presented a hybrid system with handmade deep learning methods. Further feature extraction and iris image classification were then accomplished using a unique CNN-Fuzzy model.

3. METHODOLOGY

In this section, the methodology for iris recognition is discussed. Figure 4. illustrates the system block diagram of the proposed iris recognition system.

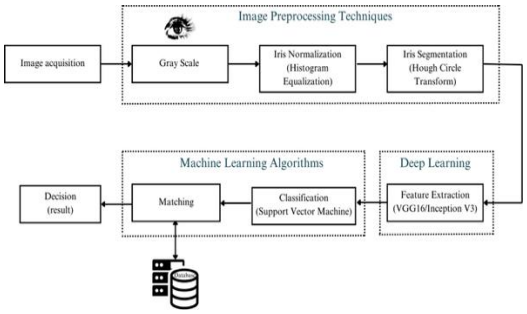


Figure 4. System Block Diagram of the Proposed Iris Recognition System

The system begins with image acquisition, where iris images are collected from the MMU iris dataset. These images undergo a series of preprocessing techniques such as histogram equalization and grayscale conversion to improve contrast and lower noise. Next, the iris region is segmented using the Hough Circle Transform to accurately isolate the iris. The segmented and normalized images are then passed through deep convolutional neural networks, VGG16 and Inception V3, for feature extraction. The extracted feature vectors are finally classified using a Support Vector Machine (SVM) to determine the identity of the individual. This modular pipeline ensures robustness, accuracy, and efficiency in iris recognition. Figure 5 and 6 are the flow charts for training and testing phases that outline a systematic approach for developing an eye image analysis system. Both begin with image acquisition. Next, image processing enhances the quality of image. CNN models will tain to extract features and classification step determines the desire accuracy. If the desire accuracy doesn't get, other CNN model will be train to extract features.

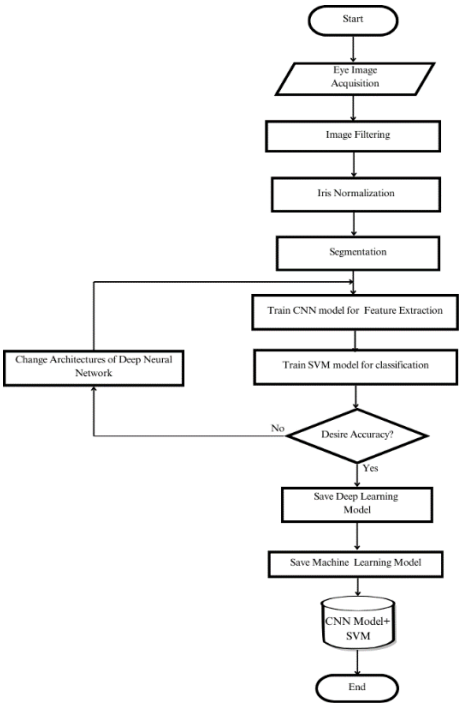


Figure 5. Flow Chart for Training Phase

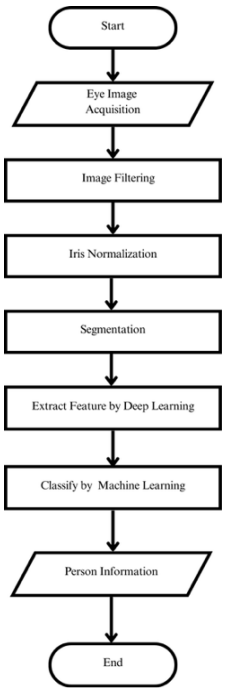


Figure 6. Flow Chart for Testing Phase

3.1. Image Acquisition

Iris identification begins with image acquisition, which uses cameras and sensors to take a series of sharp iris photos of the subject. To enhance the image's quality, certain preprocessing techniques may be used after these pictures clearly display the full eye, particularly the iris and pupil parts. Obtain photos that are sharp and have enough resolution [12]. In this paper, MMU iris dataset is utilized in place of eye image before real time data is collected. The MMU Iris Database is a publicly available dataset created by the Multimedia University (MMU), Malaysia. This Dataset consist of both 5 images each of left and right IRIS of 46 persons, totalling 460 images along with few empty files. IRIS segmentation can be performed for identifying a specific person or categorizing an IRIS image using a stored database.

3.2. Image Filtering

Applying mathematical procedures to an image is known as image filtering modify its characteristics, such as brightness, contrast, edges or color.

3.2.1. Grayscal Conversion

Converting a color image to grayscale is a form of image filtering. It is a specific type of filter that transforms the image by removing color imformation and converting it to shades of gray, based on the intensity of each pixel and it alters the image's apperance by removing color information. Instead than representing a particular color (red, green, or blue), the value of each pixel in grayscale photographs indicates the intensity of light. Grayscale images are simpler to analyze computationally because they contains less information compare to color images, other tasks are not necessary where feature extraction is done and grayscale images can save cost to store and process, especially for printing. The standard grayscale value Y from an RGB pixel is caluclated as follow:

$$Y = 0.299R + 0.587G + 0.114B \quad (1)$$

The above method is called NTCS Luminiosity method (Weighed Average) which is the most commonly use method, because it adjust with human perceiver brightness. Humans are more sensitive to green, so it gets the highest weight coefficient. And less sensitive to red and even less to blue. So it is luminance-based grayscal, which gives better result visually.

3.3. Image Normalization

Image normalization is a preprocessing step used in computer vision and image processing to standardize the pixel values of an image. It helps improve the performance of machine learning models by ensuring consistency in the input data. The iris regions, which have the same constant dimensions is produced in the normalization process, so that two photographs of the same iris under different conditions will have characteristic features at the same spatial location [13].

3.3.1. Histogram Equalization

A technique called histogram equalization modifies an image's pixel values according to its intensity histogram, producing a flat histogram and a uniform distribution of intensities. It offers a complex way to change an image's dynamic range and contrast by transforming it so that its intensity histogram takes on the shape. Step-by-Step Mathematical Process for Histogram Equalization:

Step 1: Sort the gray levels according to ascending order.

Step 2: For every gray level, count the number of pixels (n_k).

Step 3: Determine each gray level's probability density function (PDF).

$$p_k = \frac{n_k}{N} \quad (2)$$

where: N = the total number of pixels in the image

Step 4: Determine each gray level's cumulative density function (or CDF).

$$C(k)= \sum_{j=0}^{k} p_j \quad (3)$$

Step 5: Multiply C(k) with (L-1),

where: L= number of gray levels.

Step 6: Completing step 5 by rounding off the figures acquired.

Step 7: Equate new gray levels to the numbers of pixels.

3.4. Segmentation

The next step of iris recognition is iris segementation, is a process to isolate the actual iris region in a digital eye image. The circular iris region is located in the human's eye and isoloated using the Hough circle transform.

3.4.1. Hough Circle Transform

Hough Circle Transform is a perfect for detecting circular shapes like the pupil and iris in eye images. It's often used in iris recognition pipelines to locate the iris boundaries after normalization. It's a method in image processing that detects circles by transforming image points into a parameter space and looking for patterns that form circles. The gradients in an eye image are calculated in the segmentation process to produce an edge map. For every edge pixel in the edge map, the surrounding points on the circle at different radii are acquired, and votes are taken to establish the maximum values that comprise the parameters of circles in the Hough space[4]. The center coordinates and the radii of two boundaries are computed using the following equation:

$$(x-a)^2+(y-b)^2 = r^2 \quad (4)$$

where a, b are the centers of the circle and r is the radius. Hough transform does not necessitate the use of every edge pixel that defines the circle. This enhances circle localization's efficiency and accuracy.

3.5. Feature Extraction

Feature extraction is the most crucial stage in iris recognition. The technique of identifying the most distinctive information found in an iris pattern. The feature extraction technique primarily determines the recognition rate and runs duration of matching two iris templates. In this paper, the normalized iris image's features are extracted using deep CNN models, VGG16 and Inception V3. VGG16 was selected for its strong performance in capturing detailed texture features, which are critical for iris patterns and well-suited for small and medium-scale image datasets like MMU. The Inception V3 model was selected due to its efficacy and capacity to manage multi-scale features through the utilization of parallel pathways, thereby providing a more expeditious alternative.

3.5.1. Convolutional Neural Network

A specific category of deep neural networks is CNN which specially designed to work with grid-like data such as images. In addition to automatically learning picture feature representations, CNNs have also surpassed a number of traditional hand-crafted methods. Neural network models rely on layered computation. Because NN models are implemented sequentially. The input of the next layer will be the output of the previous layer. Each layer provides a single level of representation and the layers are specified by a set of weights. It is also observed that, in addition to a set of biases adjusted iris image, the input units link to output units through the weights [14]. Because CNN shares weights locally, each input site has weights that are comparable. The key idea behind CNNs is the use of convolution operations to extract features such as edges, textures, and shapes which are essential for understanding and interpreting visual data. These features are passed through layers of the network to identify more complex patterns to perform tasks such as image classification, object detection. Convolution neural network (CNN) models are a class of deep learning algorihtms primarily use for analyzing visual data. They are designed to automatically detect patterns such as edges, textures, and objects in images through layers of convolutional filters. Popular CNN

architectures like LeNet, AlexNet, VGG, ResNet, and Inception have been widely used in image recognition.

3.5.2 VGG16 models

The Visual Geometry Group at Oxford created the well-known Convolutional Neural Network (CNN) model VGG16, which is used for object identification and picture classification. It is known for its simple and uniform architecture, consisting of 16 layers with small 3x3 convolution filters and 2x2 max-pooling layers. VGG16 is used for image recognition tasks and was one of the top performers in the 2014. Its deep structure allows it to learn complex features, making it highly effective for object classification. Despite its large size and high computational cost, VGG16 remains a benchmark model for many computer vision applications.

13 convolutional layers, 5 max-pooling layers, and 3 fully linked layers composed of VGG16. 13 convolutional layers use 3x3 filters and ReLu activation functions. 5 max-pooling layers downsample the spatical dimensions of the feature maps. 3 fully connected layers are used for classification. As a result, the number of layers having tunable parameters is 16 (13 convolutional layers and 3 fully connected layers). In the first block, there are 64 filters; in subsequent blocks, this number is doubled, reaching 512. One output layer and two fully connected hidden layers complete this model. There is one completely linked output layer and two fully connected hidden layers in every VGG model.

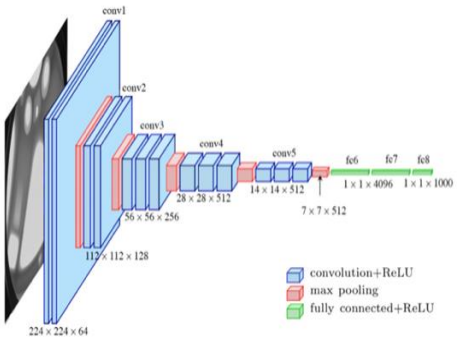


Figure 7. The Architecture of VGG16

Figure.7 describes the architecture of VGG16. It remains widely used in transfer learning and computer vision tasks to its effectiness and well structured design [15]. The model typically takes 224x224 pixel images as input. VGG16 was a popular model for image recognition tasks and has been used in various applications, including object detection, image classification, and transfer learning.

3.5.3 Inception V3 models

Inception model, also known as GoogLeNet, is a deep convolutional neural network developed by Google for image recognition tasks. Inception V3 is a part of the Inception family of models and it is an improved version of the earlier Inception models. Designed for image classification tasks, Inception V3 is known for its efficiency and high accuracy, achieved through a carefully crafted architecture that includes factorized convolutions, auxiliary classifiers, and batch normalization. One of its key innovations is the use of “Inception modules,”. This multi-branch structure makes the model both deep and computationally efficient. Inception V3 significantly reduces the number of parameters compared to earlier deep models, while maintaining strong performance on large-scale datasets like ImageNet. It is widely used in real-world applications and transfer learning due to its balanced trade-off between speed and accuracy. Figure 8. illustrates the architecture of Inception V3.

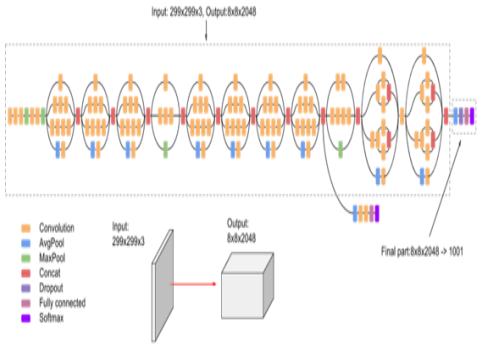


Figure 8. The Architecture of Inception V3

The Inception V3 architecture is a highly optimized deep convolutional neural network. It introduces several innovations to reduce computational cost while

maintaining high accuracy, such as factorized convolutions, auxiliary classifiers, and aggressive regularization techniques. The network starts with several convolutional and pooling layers that process the input image and reduce its spatial dimensions. It then employs multiple Inception modules, each consisting of parallel convolutional and pooling operations of different kernel sizes (1x1, 3x3, 5x5, etc.), which allows the model to extract features at multiple scales. An auxiliary classifier is included during training to help with gradient flow and regularization. Global average pooling, dropout, and a fully connected softmax layer for classification round out the network. This combination of modular design and optimization techniques makes Inception V3 both powerful and computationally efficient [16].

3.5.4 VGG16 vs Inception V3

While InceptionV3 offers a more sophisticated and optimized structure for large-scale datasets, its increased complexity can result in elevated computational demands. In contrast, VGG16 offers a compromise, exhibiting slightly lower computational cost and more stable convergence on smaller datasets.

3.6. Classification

Iris classification is a common machine learning task that involves identifying iris flower species according to characteristics including petal length, petal width, and sepal length and width. The template created during the features extraction phase required a measurement metric to evaluated the similarity between two iris templates. This metric provides one set of valuse when comparing templated from the same eye and a different set when comparing templates from various people, enabling to determine whether the two templates are the same person or different people. In this paper, support vector machine (SVM) is used to classify the i

3.6.2 Support Vector Machine

Support Vector Machine (SVM) is a powerful supervised learning algorithm used for classification and recognition tasks. [2]. Figure 9. is an example of linearly separable classification problem.

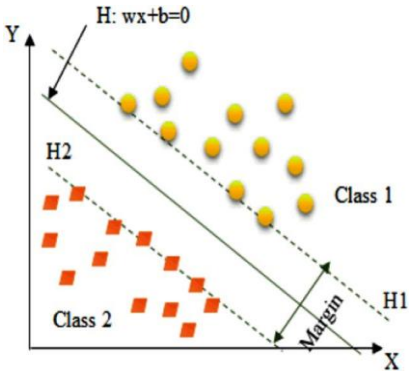


Figure 9. Linearly Separable Classification Problem

SVM performes pattern recogntion. This operates according to the structural risk minimization approach. It functions as a binary classifier, identifying the appropriate hyperplane in a high-dimensional space to divide data points of various classes. SVM seeks to optimize the support vectors, the distance between the closest data points in each class. With the use of kernel functions like the polynomial kernel or the radial basis function (RBF), due to its robustness and accuracy, SVM is used in applications such as text classification, image recognition, and bioinformatics.

The two promary components of creating an SVM classifier are: (i) identifying the best hyperplane that separartes two distinct classes of data, and (ii) converting a non-linearly separabale classification into one that is linearly separable [18].

Let x denotes a collection of input feature vectors and y indicated the corresponding class labels. The input feature vectors along with theis class labels can be expressed as $\{x_i,y_i\}$ where i = 1,2,...,N. The separting hyperplane can be represente%d as follows,

During training, this implies,

$$w.x+b=0 \quad (5)$$

$$y_i\ (w.x_i + b) \geq 1;\ i = 1,2,3,\ldots N \quad (6)$$

$\{w, b\}$ can take on many different values that form a separating hyperplane. SVM achieves this margin maximization by treating it as a quadratic optimization problem [18]. So when it computes for a new iris samples, the SVM can decide whether to accept or reject a person.

During verification, this implies,

$$f(x) = w^T \times x + b \quad (7)$$

$$\text{Prediction} = \begin{cases} \text{"Accept"}, & \text{if } f(x) \geq 0 \\ \text{"Reject"}, & \text{if } f(x) < 0 \end{cases} \quad (8)$$

3.7. Test And Result

The experimental results of the proposed techiques on MMU iris dataset are reported. The dataset includes 5 pictures of each of the left and right IRIS of 46 individuals, for a total of 460 people who were registered in the system dataset to assess the performance. The Anaconda (python distribution) tool was used as experimental platform. Anaconda Navigator is a desktop graphical user interface (GUI) that allows users to work with packages and environemts without needing to type conda commands in the terminal window. Anaconda is an open-source data science and artificial intelligencs distribution platform develoed by Anaconda,Inc. an American company. Through Navigator, users may locate the packages they desire, install them in a specific environment, run them, and update them immediately. Figure 10. shows the Anaconda Navigator.

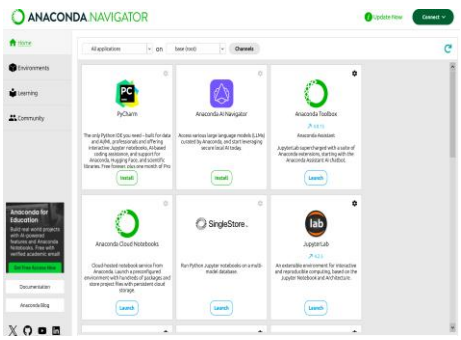


Figure 10. Anaconda Navigator

Figure 11. is the result of converting gray scale of the input image from the dataset.

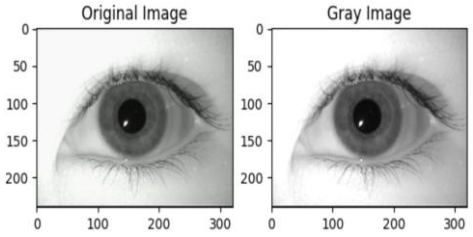


Figure 11. The Result of Converting Gray Scale

Figure. 12 shows the result of Histogram Equalization of the converting gray scale image.

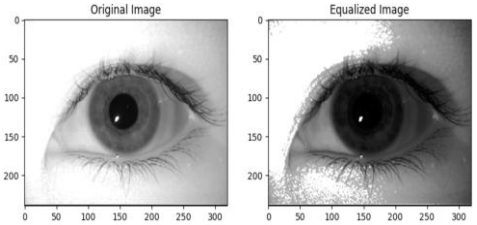


Figure 12. The Result of Histogram Equalization

Figure 13 illustrates how the Hough transformation is represented graphically.

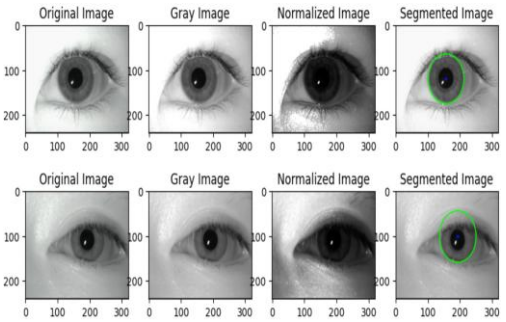


Figure 13. Segmentation using Hough cirle Transformation

CNN is used toselect the feature vector of iris images. Table 2. is the comaprison for these two feature vector from the VGG16 and Inception V3 models. Although Inception V3 provides a higher-dimensional features space (2048 features per sample) and better multiscale abstraction, it is computatuonally heavier and slower during training and inference.VGG16, with 512 features per samole, achieves a balance

between efficiency and accuracy, outperforming Inception V3 in this study (96% vs 89%). The findings indicate that VGG16's more streamlined architecture exhibits enhances efficiency and is perticularly well-suited for implmentation in setting with limited resources.

Table 2. Comparison for Two Features Vectors from VGG16 and Inception V3 Models

Model	Feature Vector Shape	Number of Feature	Remarks
VGG	(450,512)	512	VGG based encoder
Inception V3	(450, 2048)	2048	Inception based encoder

Figure 14. demonstrates the accuries of the proposed system accorss two CNN feature extractor models, 96% of VGG16 and 89% of Inception V3.

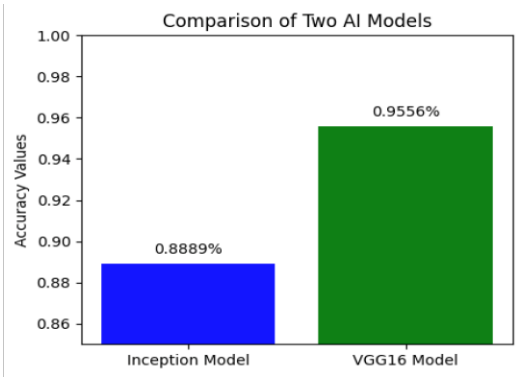


Figure 14. The Accuracies of Two Proposed Systems on the MMU Dataset

Additionally, the proposed methods performed exceptionally well on the MMU database, which was gathered in difficult circumstances like reflections and blur occlusions. This achievements highlight the remarkable ability of this proposed technology to precisely recognize iris even in difficult environmental conditions. The specific of these successes are detailed in Table 3.

Table 3. Accuries Comparison of The Proposed Models

Method	Accuracy	Precision	Recall	F1-score
Inception + SVM	89%	90%	90%	89%
VGG16 +SVM	96%	98%	98%	97%

4. CONCLUSION

This paper presented a robust biometric iris recognition system leveraging deep convolutional neural networks, specifically VGG16 and Inception V3, for enhanced security and accuracy in identity verification. By integrating advanced image preprocessing techniques such as grayscale conversion, histogram equalization, and Hough Circle Transform with CNN-based feature extraction and SVM-based classification, the proposed system demonstrated superior performance, achieving recognition accuracies of 96% for VGG16 and 89% for Inception V3 respectively on the MMU iris database. These results highlight the system's effectiveness even under challenging conditions, such as reflections and occlusions. The promising outcomes not only reinforce the reliability of deep learning in biometric applications but also suggest practical implementations in areas requiring high-security authentication, including immigration, law enforcement, and access control. Future research could explore model optimization for real-time deployment, cross-dataset validation, and integration with multimodal biometric systems for even greater security. Additionally, implementing the proposed system on hardware platforms such as a Raspberry Pi combined with a high-resolution camera could enable low-cost, portable biometric verification

solutions. A significant limitation identified relates to users wearing glasses. Eyeglasses have been shown to cause reflections, occlusions, and distortions that obscure the iris texture, leading to failed segmentation or feature extraction. The MMU dataset utilized in this study did not encompass images of subjects wearing glasses, thereby constraining the evaluation of this challenge. The proposed system can not be explicit the impact of eyeglasses and the MMU dataset doesn't utilize the images of glasses. Future work should incorporate datasets containing images with eyeglasses and investigate mitigation strategies such as: (1) Utilizing Near-Infrared (NIR) imaging which often penetrates glass reflections better than visible light; (2) Employing polarization filters on the camera to reduce specular reflections from eyeglasses; (3) Developing specialized segmentation algorithms robust to partial occlusions caused by frames; (4) Capturing multiple images from slightly different angles to avoid reflections.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Atar Mon, head of Faculty, for her continuous guidance, support and encouragement throughout this paper research. I am thankful to my co-supervisors for their patience and for provide a research enviroment that allowed for independent thought and learning. I would also like to acknowledge my parents, who always stand by my side every challenge, offering both financial and emotional support whenever needed.

REFERENCES

[1] Muhammad Waheed (2014) *Iris Recognition using Image Processing and Neural Networkz.*

[2] Abdessala HATTAB, Ali BEHLOUL, Wassila HATTAB (2024) "An Accurate Iris Recognition System Based on Deep Transfer Learning and Data Augmentation", 1st International Conference on Innovation and Intelligent Information Technology (IC3IT), DOI: 10.1109/IC3IT63743.2024.10869437

[3] Abdurahman Aminu Ghali, Sapiee Jamel, Kamaruddin Malik Mohamad, Nasir Adubakar Yakub, Mustafa Mat Deris (2017) "A Review of Iris Recognition Algorithms", International Journal on Informatics Visualization, Vol.1, No.4-2, e-ISSN: 2549-9904, ISSN: 2549-9610.

[4] Md.Shafiul Azam, Humayan Kabir Rana (2020) "Iris Recognition using Convolution Neural Network", International Journal of Computer Applications, Vol. No-175.

[5] Yuewen Tang, Xiangkui Li, Yulian Jiang (2019) "Research on Human Eye Iris Recognition Based on Inception v3", Frontiers in Signal Processing, Vol.3, No.4.

[6] John Daugman (2004) "How Iris Recognition Works", IEEE transactions on Circuit and Systems for Video Technology, Vol.14, No.1.

[7] Li Ma, Tieni Tan; Yunhong Wang, Dexin Zhang (2003) "Personal Identification based on Iris Texture Analysis", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.25, DOI.10.1109/TPAMI. 2003.1251145.

[8] M. G. Alaslni, L. A. Elrefaei et al., "Transfer learning with convolutional neural networks for iris recognition," Int. J. Artif. Intell. Appl, Vol. 10, No. 5, pp. 47–64, 2019.

[9] S. Arora and M. S. Bhatia, "A computer vision system for iris recognition based on deep learning," in 2018 IEEE 8th International Advance Computing Conference (IACC). IEEE, 2018, pp. 157–161.

[10] M. G. Alaslani, "Convolutional neural network-based feature extraction for iris recognition," International Journal of Computer Science & Information Technology (IJCSIT) Vol, vol. 10, 2018.

[11] M. M. Khayyat, N. Zamzami, L. Zhang, M. Nappi, and M. Umer, "Fuzzy-cnn: Improving personal human identification based on iris recognition using lbp features," Journal of Information Security and Applications, vol. 83, p. 103761, 2024.

[12] Rana HK, Azam MS, Akhtar MR (2017) "Iris recognition system using PCA based on DWT", SM Journal of Biometrics & Biostatistics vol. 2, no. 3, pp.: 1015, DOI 10.5281/zenodo.2580202, 2017.

[13] John G. Daugman (1993) “High condense visual recognition of persons by a test of statistical independence”, Pattern Analysis and Machine Intelligence, IEEE Transactions on, 15(11):1148-1161.

[14] D. Scherer, A. Müller, and S. Behnke (2010) "Evaluation of pooling operations in convolutional architectures for object recognition," Artificial Neural Networks–ICANN 2010, pp. 92-101.

[15] Karen Simonyan, Andrew Zisserman (2015) “Very Deep Convolution Networks for Large Scale Image Recognition”, Computer Vision and Pattern Recogntion(cs.CV), https://arxiv.org/abs/1409.1556

[16] Christain Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens (2015) “Rethinking the Interception Architecture for Computer Vision”, arXiv:1512.00567v3 [cs.CV].

[17] Rana HK, Azam MS, Akhtar MR, Quinn JMW, Moni MA (2019) “A fast iris recognition system through optimum feature extraction”, PeerJ Computer Science, 5:e184 https://doi.org/10.7717/peerj-cs.184.

[18] Jony, Mehdi Hassan, et al (2019) "Detection of Lung Cancer from CT Scan Images using GLCM and SVM." 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), IEEE.

PERFORMANCE ANALYSIS OF ACTIVE NOISE CANCELLATION USING ADAPTIVE FILTER ALGORITHMS FOR TRANSPORTATION ENVIRONMENT

Myat Min Khant[1], Atar Mon[2],Soe Nandar Lin[3], and Aye Theingi Oo[4]

[1,2,3,4] Faulty of Electronic Engineering, University of Technology (Yatanapon Cyber City)

myatminkhant999178@gmail.com, atarmon@gmail.com, soenandarlin.phyo@gmail.com, ayetheingioo777@gmail.com

ABSTRACT

Active noise cancellation (ANC) has gained significant attention due to its effectiveness in reducing unwanted sounds in various environments. This system focuses on the design and implementation of an adaptive filter-based ANC system for transportation environments by using Raspberry Pi. For initial simulation, MATLAB is employed to analyze the impact of key parameters such as step size and filter length on noise cancellation performance. The LMS, NLMS, and FxLMS adaptive-filtering algorithms are applied and compared with one another in order to create an adaptive filter that generates the best estimate of the intended signal from the noisy environment. The performance of these algorithms is examined using the Mean Square Error (MSE) and the Signal-to-Noise Ratio (SNR). Among these algorithms, FxLMS has the best performance. ANC systems for transportation environments typically operate background noise frequency range from 20 Hz to 10 kHz. MATLAB programming has been employed to simulate the algorithms. The experimental setup involved the use of dual microphones to record a range of ambient noises, including tire friction and engine hum, captured from real-world transportation contexts.

KEYWORDS

Active noise cancellation, adaptive filtering, FxLMS algorithm, MSE, SNR

1. INTRODUCTION

Today's society is rapidly becoming more technologically advanced. While these advancements have significantly improved in comfort and quality of life, they have also introduced new challenges that negatively impact the environment. With the advancement of industrial society, environmental noise has increased significantly. Humans are routinely exposed to acoustic noise in various environments, including industrial settings, residential areas, public transportation systems, hospitals, and aircraft. These noises originate from a wide array of sources such as engines, blowers, transformers, fans, compressors, MRI machines, and ventilation systems. The Active Noise Cancellation (ANC) technology is extensively utilized in a variety of industries, such as consumer electronics, industrial manufacturing processes, and audio processing [2]. It is used in applications like noise-cancelling headphones, active mufflers, and devices for reducing noise in air conditioning ducts. ANC is limited to providing localized noise cancellation around the error measurement in numerous real-world scenarios. As demonstrated in Figure 1, Active Noise Cancellation (ANC) employs a secondary speaker that generates an interference wave to eliminate undesired noise from a specific location.

There are many algorithms used in Active Noise Cancellation (ANC) system. These algorithms are the Least Mean Square (LMS) algorithm, Filtered-x Least Mean

Squares (FxLMS) and Normalized Least Mean Squares (NLMS). The FxLMS approach is one of these algorithms that is frequently utilized in active noise cancellation applications because of its low computational complexity. By generating an input signal with an adaptive filter, it lowers noise in a target area.

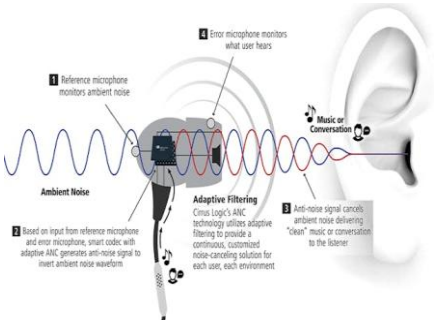


Figure 1. Active Noise Cancellation System

In this paper, the recorded audio signals from the transportation area, in which the FxLMS algorithm is filtered. Afterwards the architecture is implemented with Raspberry Pi 3B+ where the algorithm reads the mean square error to determine the state of the error signal and then filters the unwanted noise free signals.

2. System Design and Architecture

Figure 2 shows the block diagram of the system. In the purposed system, an application has been developed on Raspberry Pi 3B+ and MATLAB.

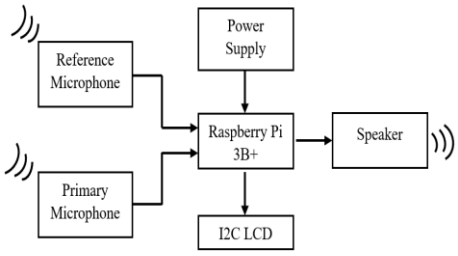


Figure 2. Block Diagram of the System

The Raspberry Pi 3B+ is used with coding adaptive filtering algorithm as shown in Figure 2. The audio signal recorded in the transportation area across the microphone inputs to Raspberry Pi 3B+. The output of the Raspberry Pi is connected with speaker to result the noise free desired audio signal. In adaptive filtering, FxLMS algorithm is used to filter the input signal to Raspberry Pi3B+. With a sample rate of 44.1 kHz and a 32-tap filter length, the Raspberry Pi 3B+ performs adaptive filtering in real time with a latency of less than 20 ms, which is adequate for speech-frequency ANC.The software aspect consists of coding a MATLAB simulation for FxLMS algorithm. The I2C LCD is used to show the reduction if noise level in dB. This system is made up of hardware with Raspberry Pi3B+ and software with MATLAB that work together to provide the intended functionally. In order to ensure that the system functions in a causal manner with minimal latency, the buffer size was optimized to be less than 512 samples. The Raspberry Pi within each frame is responsible for processing, ensuring uninterrupted real-time operation.

3. Methodology

Figure 3 displays the block diagram for noise cancelling system based on adaptive filtering algorithm.

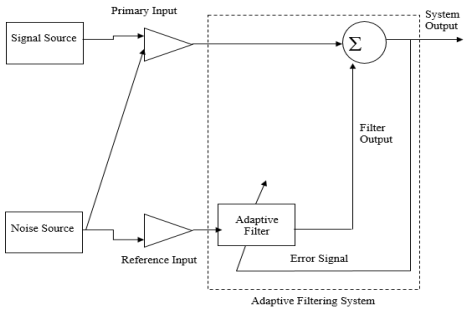


Figure 3. Adaptive Noise Cancelling System

The technique for computing a desired signal from a noise-corrupted measurement is called the adaptive filtering of noise cancellation. The technique employs a reference input with noise that is connected with the primary noise and a primary input with the distorted signal. The estimated signal is obtained by adaptively filtering the

reference input and subtracting it from the primary input. When noise is subtracted from the primary input, the signal may become distorted. This could raise the noise level if done incorrectly. The process of filtering and subtraction are managed by the adaptive strategy as the transmission routes' characteristics are imprecise and unexpected. Consequently, an adaptive filter is used, which can alter its impulse response to minimize a distorted signal dependent on the filter output. The adaptive filter is used in this work to cancel out disturbance signals. The primary signal and the reference signal are used as inputs by the program to implement the system. These are all recorded acoustic signals. The gradient vector estimates from the supplied data are used by the FxLMS algorithm. The program utilizes the reference signal and the primary signal as inputs to implement the system. A comprehensive database of these auditory signals has been meticulously compiled.

3.1. Filtered-X Least-Mean-Square (FXLMS) Algorithm

There are numerous noise cancellation algorithms that can be used and implemented with MATLAB. An algorithm that modifies its characteristics during execution based on information availability and prior procedures is known as an adaptive algorithm. The FxLMS algorithm, or Filter-X Least Mean Square algorithm, is a well-known algorithm for adaptive systems that functions as a self-adjusting algorithm. The gradient-based algorithm known as FxLMS can be used to identify the Active Noise Cancellation (ANC) system when a secondary path is present.

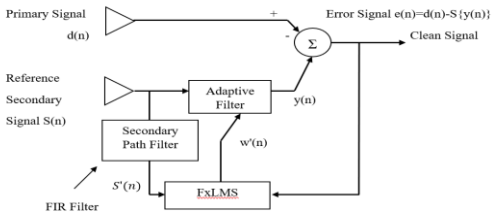


Figure 4. System Block Diagram of FxLMS Algorithm

Figure 4 illustrates the functional block diagram of an FxLMS algorithm that adjusts an ANC controller. An estimate model of the secondary path is represented by sˆ in this Figure 4 and it is evident that the conventional LMS method uses the reference signal after it has been filtered by sˆ. The secondary path compensation is the major distinction between the LMS and FxLMS algorithms.

The error signal, or remaining noise at a microphone, is minimized by adjusting the filter coefficients using the FxLMS algorithm. The secondary path filter is incorporated by the algorithm to ensure proper adaptation. In mathematical relationship, the filter output is calculated by Equation 1.

$$y(n)=w^{T}(n).x(n) \quad (1)$$

Where y(n) is the filter output, $W^{T}(n)$ is the filter weight's transverse correlation, and x(n) is the noise cancellation system's input signal. The update filter output is computed by Equation 2.

$$w(n+1)=w(n)+\mu.e(n).\hat{x}(n) \quad (2)$$

Where, w(n) is the filter weight of the signal, μ is the step size for noise cancellation, e(n) is filter error vector and $\hat{x}(n)$ is the filter reference signal. After that, the error signal is evaluated in Equation 3.

$$e(n)=d(n)-y(n) \quad (3)$$

Where d(n) is desired signal of the system. For secondary path of the algorithm, the filter reference signal is calculated in Equation 4.

$$\hat{x}(n)=S(n).x(n) \quad (4)$$

Where S(n) is the secondary path of the input signal. Through its output, y(n), the FxLMS method performs an adaptation procedure to vector w(n) in an approach that minimizes the power of the residual acoustic noise e(n).

The FxLMS algorithm's capacity to consider secondary path dynamics renders it more suitable for transportation contexts than classic LMS and NLMS algorithms. These secondary path dynamics are of particular importance in non-stationary noise circumstances, such as tire friction, engine vibration, and fluctuating road noise. FxLMS introduces a filtered reference signal that more accurately simulates real-world conditions than LMS/NLMS, which require direct signal access. This ensures more robust convergence and stable noise cancellation, even when noise characteristics vary quickly.

3.2. Normalized Least-Mean-Squares (NLMS) Algorithm

The LMS method with a time varying step-size parameter is known as the NLMS algorithm. The signals used as input and output follow the same parameters outlined for the LMS algorithm. The system block diagram for NLMS algorithm is shown in Figure 5.

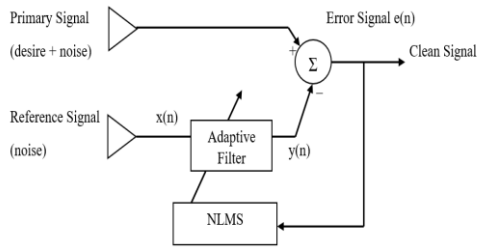


Figure 5. System Block Diagram of NLMS Algorithm

At iteration n, the weight vector w(n)'s correction from sample (n+1) is adjusted with reference to the input vector x(n). The evaluation of the tap weight vector adaptation for the NLMS algorithm is conducted through the application of Equation 5.

$$w(n+1)=w(n)+\beta\frac{\hat{x}(n)}{|x(n)|^2}e(n) \qquad (5)$$

Where β is the normalized step size parameter range from 0 to 2. The NLMS algorithm adapts better to signals with varying power and faster convergence than LMS.

3.3. Least-Mean-Squares (LMS) Algorithm

The LMS algorithm modifies the filter coefficients until the variance between the intended and actual signal is as minimal as possible in order to provide the least mean-squares of the error signal. The stochastic gradient descent approach is the foundation of the LMS algorithms since it only modifies the filter coefficients in response to the current error.

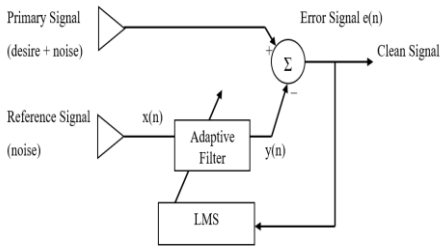


Figure 6. System Block Diagram of LMS Algorithm

Figure 6 illustrates the system block diagram of LMS algorithm. The step-by-step calculations of the filtering process for LMS are the same as FxLMS above. However, the tap weight vector adaption of LMS is calculated by Equation 6.

$$w(n+1)=w(n)+\mu.e(n).\hat{x}(n) \qquad (6)$$

where w(n) is the estimate of the weight value vector at time n, x*(n) is the input signal vector, e(n) is the filter error vector and μ is the step-size, which determines the filter convergence rate and overall behaviour.

The sequential operation of algorithm for this system is shown in Figure 7. Firstly, input audio signals from transportation area recorded in this system. And then, the primary input signal and the reference input signal are defined by the recorded audio signal. After that the process of adaptive

filter in FxLMS algorithm is operated. The input signals are filtered by the algorithm and the error signal is computed in this system. Following that, mean square error (MSE) is evaluated.

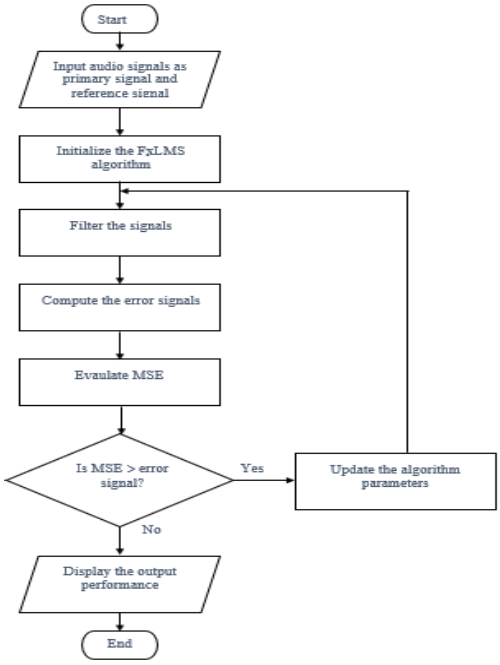


Figure 7. Flow Chart of Adaptive Process in Active Noise Cancellation System

At that point, the mean square error is less than the resulted error signal. It will check the condition is true or false. If the condition is yes, the adaptive parameters are updated by the algorithm and then return to the program. After which, the condition is satisfied and the value of mean square error is approached to zero. And then, the output result is displayed. Finally, the system will go to the end of adaptive process in active noise cancellation system.

4. Simulation Results of the System

Performance Results of algorithms are analysed with MSE, SNR and Spectrogram.

4.1. Mean Square Error (MSE)

The performance of ANC systems is frequently assessed using the Mean Square Error (MSE) statistic. It measures the discrepancy between the ANC system's actual output and the intended noise-cancelling signal. The MSE is calculated as in Equation 7.

$$MSE=\frac{1}{N}\sum_{n=1}^{N}e(n)^2 \quad (7)$$

Where e(n) is the error signal that difference between the input signal and the ANC outputs. N is the number of samples. The ANC systems use adaptive filter to minimize MSE.

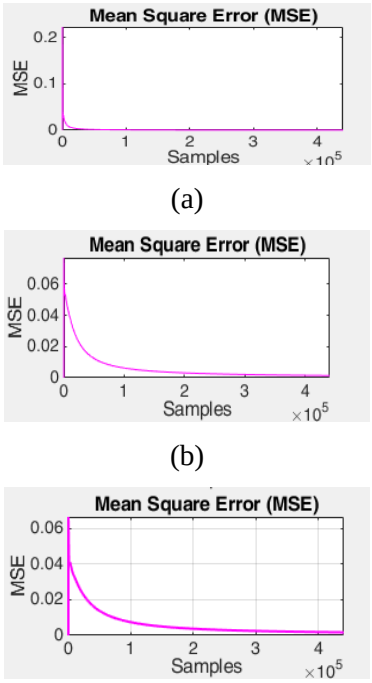


Figure 8. Comparison of MSE for (a) FxLMS (b) NLMS (c) LMS

In Figure 8 shows the MSE comparison of FXLMS, NLMS and LMS algorithms. In this Fingure, the MSE of FxLMS was found to be 4.08, which indicates the lowest MSE among the models. This was followed by LMS, which had an MSE of 5.213, and NLMS, with an MSE of 5.067. The lowest value of MSE means better noise cancellation for the system. By minimizing MSE, ANC systems achieve better noise reduction, leading to improved performance

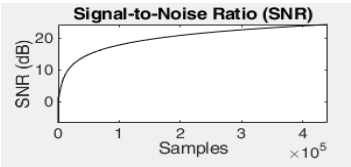
in noise cancelling applications. Within the target frequency bands, an MSE of 4.08 is indicative of a discernible noise energy reduction of approximately 20–30%.

4.2. Signal to Noise Ratio (SNR)

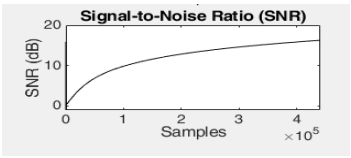
A crucial metric for assessing the work that ANC systems perform is the signal to noise ratio (SNR). It measures the ratio of the desired signal such as music or speech to the residual noise after cancellation, indicating effectively the ANC system suppresses unwanted noise. The SNR in ANC system measures the difference between the targeted signal's power and the post-cancellation residual noise power. The mathematical calculation of SNR is expressed in Equation 8.

$$SNR=10\log_{10}\frac{P_{signal}}{P_{noise}} \quad (8)$$

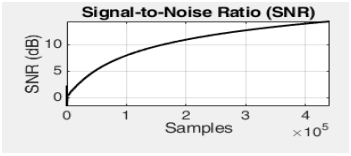
Where P_{signal} is the power of desired signal such as music or speech. P_{noise} is the power of the residual noise after ANC.



(a)



(b)



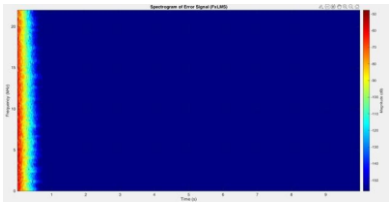
(c)

Figure 9. Comparison of SNR for (a) FxLMS (b) NLMS (c) LMS

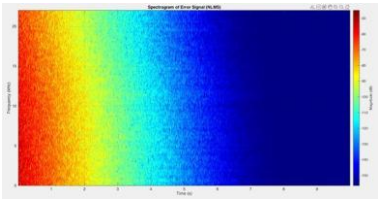
In Figure 9 shows the SNR comparison of FXLMS, NLMS and LMS algorithms. A high output signal-to-noise ratio (SNR) is preferred since it demonstrates how extremely effective the adaptive filter can eliminate a large amount of noise and produce an accurate estimate of the desired signal. The signal-to-noise ratio increases as the output noise power decreases. Reduced output power results in completely noise-free filtered signals. In comparison, FxLMS evaluates the maximum value of SNR, and it has higher performance than another algorithm.

4.3 Spectrogram of ANC System

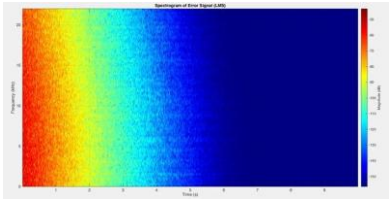
A spectrogram is a graphic representation concerning the way a signal's frequency content varies over time. It shows the signal's frequency in vertical axis, time in horizonal axis, and amplitude in colour intensity.



(a)



(b)



(c)

Figure 10. Comparison of Spectrogram for (a) FxLMS (b) NLMS (c) LMS

As demonstrated in Figure 10, spectrogram charts reveal that secondary path modeling facilitates enhanced low and mid-frequency noise removal by FxLMS. Compared to LMS and NLMS, FxLMS attenuates

prominent frequencies more sharply. However, it makes no mention of particular bands, such as 100–1000 Hz, which are particularly important for transportation. A spectrogram can effectively illustrate the different adaptive filtering algorithms perform in an ANC system under the same step size at 0.01. Due to the secondary path compensation, the FxLMS approach attenuates prominent frequencies more sharply than LMS and NLMS for ANC.With its secondary path modeling, it exhibits the fastest noise elimination.

In this part, the outcomes of the simulation of LMS, FxLMS and NLMS algorithm are compared as noise canceller by using adaptive filtering. The algorithms are implemented various step size. Utilizing filter order and processing resources, the effectiveness of the noise reduction for the various algorithms was assessed based on noise reduction. Filter order plays an important role in active noise control system. After writing the program according to the algorithm, the program is compiled to MATLAB simulation. In simulation process, the filter length, the step size and the iterations of the signal are used as an input. Through altering the step size of the input signal, the values of error signal and the signal to noise ratio are changed according to the algorithm. In these algorithms, the filter length is 32 taps, the step size for coefficient adaption is 0.01 and the iteration rate is 1000 seconds per iteration.

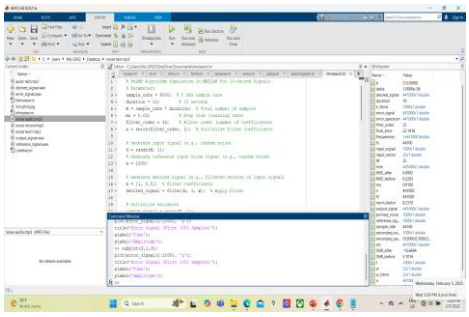


Figure 11. System Simulation with FxLMS Algorithm in MATLAB

In Figure 11, the system is simulated with FxLMS algorithm in MATLAB software. In this system, the frequency in primary input noise is used at 8000 Hz and the reference noise is 1000 Hz. The entired procedure is used in 10seconds.

Figure 12 presents the simulation results of the active noise cancelling system by utilization of FxLMS algorithm. The step size is normalized by the FxLMS algorithm at each iteration based on the power of the filtered input vector, ensuring stability and consistent convergence regardless of input amplitude fluctuations.

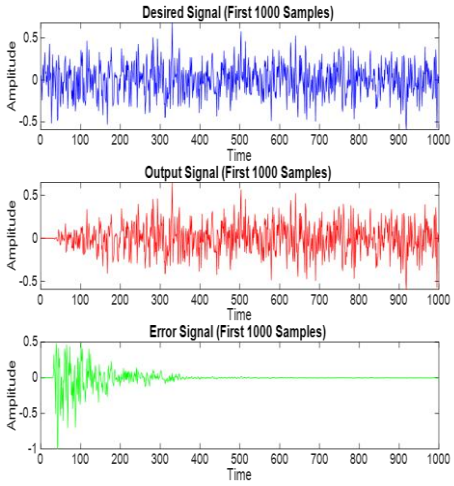


Figure 12. Simulation Results of Desired Signal, Output Signal and Error Signal in FxLMS Algorithm

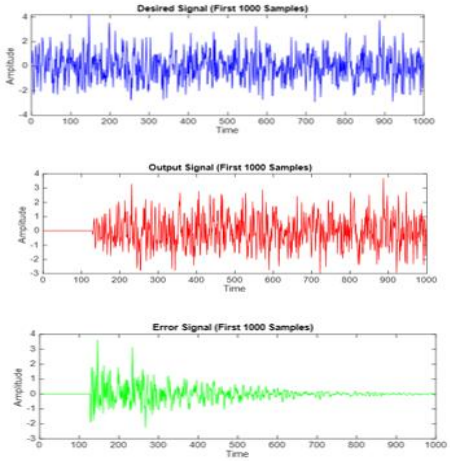


Figure 13. Simulation Results Of Desired Signal, Output Signal And Error Signal In LMS Algorithm

The results are presented with illustrative Figure 13. These results affirm the LMS algorithm's capability to effectively model linear systems and minimize error over

time. In NLMS algorithm results, Time-domain plots of the desired, output, and error signals revealed that the NLMS output increasingly approximate the Time-domain plots of the desired, output, and error signals exposed that the NLMS output increasingly resembled the desired signal, while the error signal reduced. The NLMS simulation results are presented in Figure 14.

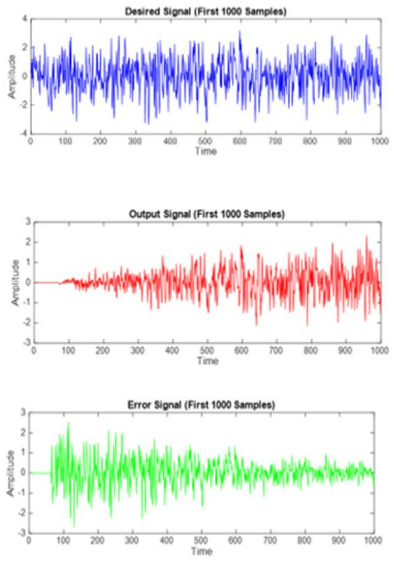
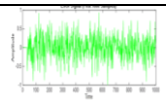
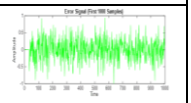
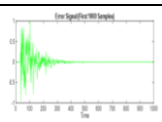


Figure 14. Simulation Results Of Desired Signal, Output Signal And Error Signal In NLMS Algorithm

Table 1. Comparison Results of Error Signal

Algorithm	e(n)	SNR	Result
LMS	5.213	8.23dB	
NLMS	5.067	8.74dB	
FxLMS	4.08	14.36dB	

Utilizing a step-size of 0.01 for the coefficient rate and a filter order of 32, the error signal can be calculated. As illustrated in Table 1, a comparative analysis of the error signal for the LMS, NLMS, and FxLMS algorithm is provided.

5. Conclusion

The paper investigated and compared the performance of various active noise cancellation system approaches in simulations. Experimental performance has been described utilizing the NLMS, FxLMS, and LMS algorithms. The comparative analysis of LMS, NLMS, and FxLMS algorithms reveals that FxLMS consistently achieves the lowest MSE and the highest SNR across all tested step sizes, indicating superior performance in adaptive filtering tasks. These findings underscore the efficacy of FxLMS in minimizing error, particularly at optimal step sizes, and suggest its suitability for applications requiring precise adaptive filtering.

Acknowledgment

The author would like to express sincere appreciation the Rector and Pro-Rector of University of Technology (Yatanarpon Cyber City) for kind permission to prepare for this paper. The author would also like to give special thanks to my supervisor and all teachers in Faculty of Electronics Engineering and all who willingly helped the author throughout the preparation of the paper.

References

1] LingYi Lu*, Fan Yang, Hu Huang, ZeZhong Liu, MingJun Xie, YuXin Xie, “Active noise reduction for traffic noise based on Filtered-X least mean square algorithm”. 2021 4th International Conference on Mechatronics and Information Technology (ICMIT 2021).

[2] Dongyuan Shi, Woon-Seng Gan, Bhan Lam, Shulin Wen, and Xiaoyi Shen, “Active Noise Control based on the Momentum Multichannel Normalized Filtered-x Least Mean Square Algorithm”. arXiv:2308.03684v1 [eess.AS] 7 Aug 2023.

[3]Desai Nilay, Pandya Utpal, and Desai Chintan, “Active Noise Control Using LMS & NLMS Algorithm”. Journal of Engineering & Technology (JET)- Volume 1- Issue 1.

[4] B A Sujathakumari, Ravi S Barki, Lokesh A R, Pavithra C M "Active Noise Cancellation using Adaptive Filter Algorithms". International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 7 Issue 02, February-2018.
[5] Zhuo Diao, Conghao Jin, "Review of ANC Algorithm". Journal of Physics: Conference Series 2386 (2022).
[6] Junyan Dong, Yupi Fu, Bomin Zheng, Kechao Li and Jichen Qu,"Overview of the application of active noise control technology in the indoor acoustic environment". IOP Conf. Series: Earth and Environmental Science 508 (2020) 012021. doi:10.1088/1755-1315/508/1/012021.
[7] T. Suman, M.Venkatanarayana "Comparison Study of Active Noise Cancellation". 2019 JETIR February 2019, Volume 6, Issue 2 www.jetir.org (ISSN-2349-5162).
[8] P. Lia, and X. Yua, "Active noise cancellation algorithms for impulsive noise". Mech Syst Signal Process. 2013 April 1; 36(2).
[9] Krishna A/L Ravinchandra, Ka Fei, Thang and Chee Yong, Lau, "Active Noise Reduction using LMS and FxLMS Algorithms". International conference on computer vision and machine learning IOP Conf. Series: Journal of Physics: Conf. Series 1228 (2019) 012064 IOP Publishing doi:10.1088/1742-6596/1228/1/012064.

Author's

Biography

Myat Min Khant graduated from Technological University (Monywa) in Myanmar in 2023 with a Bachelor of Engineering in Electronics and Communication Engineering. He is currently conducting his M.E. thesis research in the field of Electronics and Communication Engineering at the University of Technology (Yatanarpon Cyber City) in Myanmar. His thesis focuses on using adaptive filtering methods to design and analyze active noise cancellation systems. His areas of interest include digital signal processing, embedded systems, and real-time audio signal augmentation with Raspberry Pi and MATLAB.

AUTONOMOUS MOBILE ROBOT BASED ON ROBOT OPERATING SYSTEM2 (ROS2) WITH GAZEBO

Thet Hsu Wai [1], Atar Mon [2], Myo Su Su Theint [3], and Soe Nandar Lin [4]

[1,2,3,4] Faculty of Electronic Engineering, University of Technology (Yatanarpon Cyber City)

thethsuwai95@gmail.com, atarmaon@gmail.com, myosumyomyo1987@gmailcom, soenandarlin.phyo@gmail.com

ABSTRACT

*The system proposes a mobile, autonomous robot that operates in a Gazebo environment using a Lidar sensor and ROS2. The three components are mapping, localization, and navigation. The SLAM technique uses the occupancy grid mapping algorithm to create a map of an unknown environment, and the recursive Bayes filter is used to simultaneously determine the robot's position within the created map. For the stability control system, ROS2 control is used for the robot. In ROS2, the robot uses Navigation2 to plot its route and avoid obstacles. By avoiding the barriers on the recorded map, the proposed system will also show how the environment maps and how the intended goal is reached. RViz displays these findings. This paper presents a prototype robot and modeling for robot state estimation. Gazebo is used to create an environment model consisting of four rooms within a 5.8 m * 8.9 m area, and the obstacle density is low.*

KEYWORDS

ROS2, SLAM, Occupancy Grid Mapping, Recursive Bayes Filter, ROS2 Control, Navigation2

1. INTRODUCTION

Robots are designed to assist humans in a variety of tasks, from industrial applications to daily tasks. Autonomous mobile robots are being employed extensively in a variety of industries, restaurants, hotels, hospitals, and distribution and warehousing businesses. An autonomous mobile robot is an automated system that chooses its own location, steers clear of obstructions, and plots a course to reach its objective. For a mobile robot to perform planning and localization and to comprehend the appearance of its surroundings, mapping is necessary.

In order to avoid obstacles and choose the optimal path from its current location to the desired location, autonomous mobile robots must perform mapping, localization, and autonomous navigation. Robots must take environmental uncertainty into account when performing these jobs. Probability theory is used because robot motion and observations are unpredictable. Robots must also do state estimation, localization, mapping, navigation, and motion planning in a recursive approach.

This paper presents a ROS2-based autonomous mobile robot in a Gazebo environment. This system uses the Navigation2 stack planning server for autonomous navigation and the SLAM technique to create a map of the Gazebo environment. In this paper, a 2D map of the environment is created using a lidar sensor. The occupancy grid mapping algorithm is used to build maps. The world is divided into discrete cells on grid maps. The occupancy probability determines whether a cell is occupied or not via occupancy grid mapping. This system then uses a recursive Bayes filter to determine the robot's position based on its mobility and lidar sensor. Localization and mapping must be done at the same time if the robot is moving in an unfamiliar environment. The two steps are interdependent. A proper map is necessary for accurate localization, and vice versa. The simultaneous localization and

mapping (SLAM) technique is the name given to this process.

The two components of the SLAM technique are posture graph optimization at the back end and sensor-dependent front-end processing. Visual SLAM, Lidar-based SLAM, Radar SLAM, Acoustic SLAM, and Multi-Sensor SLAM are the front-end processing-based SLAM types. For this system, Lidar-based SLAM will be utilized due to its stability, ease of use, precise map data, and reduced computation. The Lidar-based SLAM approach uses a laser beam to hit barriers and measure the reflection in order to estimate the distance.

Features-based SLAM, direct-based SLAM, Kalman Filter SLAM (EKF-SLAM), particle filter SLAM (Fast SLAM), graph-based SLAM, and deep learning-based SLAM are all back-end processing-based SLAM techniques. Because it is flexible in situations with high levels of uncertainty, graph-based SLAM is used in this system. To estimate the robot's trajectory, the graph-based SLAM approach creates a pose graph with nodes. The SLAM algorithms will be implemented using Ceres Solver on an Intel Core i7 laptop running Robot Operating System (ROS).

The robotic platforms for robotic development are Microsoft Robotics Studio, the robot operating system (ROS), and the player. The open-source platform ROS is used to create sophisticated robotic systems that can perform functions including navigation, control, motion planning, and sensing. Additionally, it offers motion planning, communications, package management, simulation, visualization, motion drivers from sensors and actuators, controller, robot model definition, frame transformations, and more.

Nodes are computation-performing processes in ROS, and a system consists of several nodes. Topic, service, and action are the three categories of communication techniques that a node uses to interact with other nodes. ROS uses topics and messages to connect sensors, actuators, and the control system. ROS1 and ROS2 are the two versions of ROS. This system uses ROS2 Humble on Linux Ubuntu 22.04 due to its multi-threading capabilities, reliability, and real-time performance.

2. System Overview

The mapping and autonomous navigation processes are the two primary divisions of the system. The system block diagram is described in Figure 1. The user has to decide between mapping and navigation, as seen in the block diagram below.

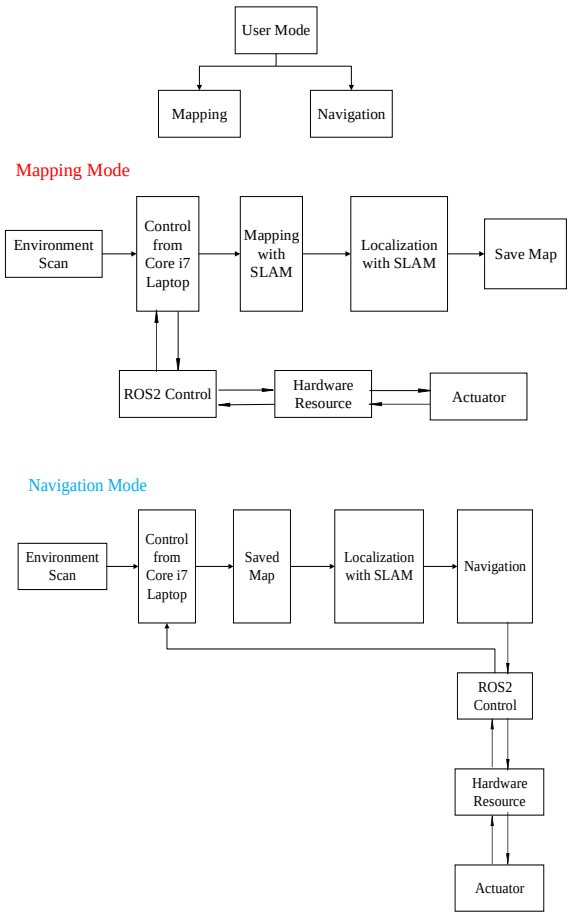


Figure 1. System Block Diagram

The mapping process requires the user to provide the actuator from hardware resources with the command velocity via ROS2 control. Lidar is used to sense the environment. Next, a recursive Bayes filter is used to estimate the robot's position based on Lidar data (observation) and motion. Lidar and robot pose are used to generate a 2D map of the surroundings using the occupancy grid mapping algorithm.

This algorithm uses log odds form to determine the likelihood that a cell is occupied or not. Since mapping and localization are done concurrently when a robot is moving in an unfamiliar area, the simultaneous localization and mapping (SLAM) technique is

used to find motion models and observation models. Once localization and mapping are complete, the user stores the map for self-navigating.

Using a map that was saved throughout the mapping process, the robot initially locates its starting location for the navigation phase. The SLAM toolbox is utilized for the localization process. The robot uses the Navigation2 stack to carry out all navigational tasks when the user provides the appropriate goal from RViz2. The Navigation2 stack consists of a controller server to manage the robot's movement, a planner server to create a global path planner, and a map server to load a saved map. In order to do path planning and obstacle avoidance, the Navigation2 stack in the ROS2 system generates command velocity for the ROS2 control. The robot receives velocities from ROS2 control.

3. Design Structure And Methodology

3.1. Robot Model Design

The two-wheeled differential drive robot model is designed to simulate within the Gazebo environment. The simulation is carried out on a two-wheeled differential drive mobile robot as shown in Figure 2. This robot includes two caster wheels for balance and two DC motors with encoders. The motors are equipped with 0.066-meter-diameter wheels. Two wheels are separated by a base width of 0.192 meters. The first plate is equipped with a Lidar sensor. The computer utilized to operate the ROS2 system on the robot in this paper is based on Linux (Ubuntu 22.04).

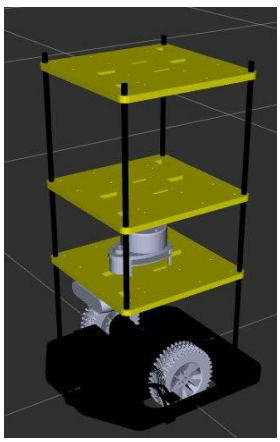


Figure 2. Robot Model

In ROS2, the robot kinematics are described using the Unified Robot Description Format (URDF). A hardware plugin called gazebo_ros2_control/GazeboSystem is used to model a closed-loop kinematic chain. Since this paper uses a differential drive two-wheel robot, control of left and right wheel joints is required. The simulation's laser ray specifications include a 10 Hz update rate, 360 samples, a minimum and maximum angle of -2.3562 and 2.3562, a minimum range of 0.05, and a maximum range of 12. The result is the "/scan" topic with the sensor_msgs/LaserScan type.

There are 3600 points per scan for 360-degree lidar resolution, with a scan frequency of 20 Hz and an angle of 0.1 between each scan point. The 360-degree lidar resolution improves mapping accuracy by capturing finer details, improving scan matching, and reducing localization drift.

Table 1. Comparison between 360-degree Lidar and Depth Camera

Feature	360-degree Lidar	Depth Camera
Field of View	360-degree (horizontal)	Limited (60 degree to 90 degree)
Range	Long (10–100 m)	Short (0.3-5 m)
Depth Accuracy	High over large distance	High only at close range
Light Sensitivity	Works in total darkness	Affected by lighting
Mapping Accuracy	Very high for large and open spaces	Good for small, cluttered indoor areas
Moving Objects	Handles them more robustly	Motion blur

Depth cameras are more expensive than 360-degree lidar. Therefore, 360-degree lidar is widely used for environment modeling.

The navigation system depends on the transformation of the robot's frame and environmental data. In ROS2, the transform (tf) tree is used to define the significance between frames. In order for the robot's algorithms to function properly, this system

uses base_footprint, base_link, and odom frame as the beginning frame for tracking the robot's position and map frame for the map's origin. TF tree of the system is shown in Figure 3.

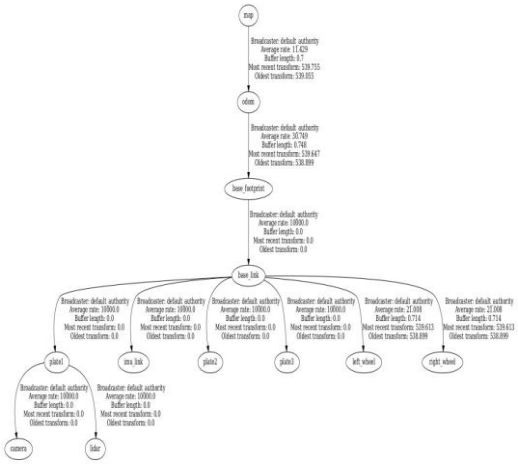


Figure 3. TF Tree of the System

3.2. ROS2 Control

The ROS2 framework contains a collection of packages called ROS2 control that are used to control robots. Figure 4 shows the block diagram of ROS2 control structure.

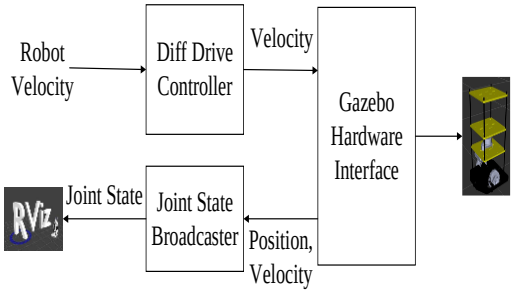


Figure 4. ROS2 Control Structure

The controller manager and resource manager are the two parts of ROS2 control. The ros2_control framework's controllers and hardware abstraction sides are connected by the controller manager. The controller type (diff_drive_controller, forwarding controller, joint_trajectory_controller, gripper controller), robot type (arm, mobile, drone, etc.), and control type (effort, position, velocity, joint_state) must be selected in the controller manager. In this paper, velocity and joint_state control types are used for a mobile robot using a differential drive controller. The robot joints' position, velocity, and effort are provided by the joint_state_controller and published into the /joint_states topic. The physical robot hardware is communicated with via the resource manager. It outlines a common set of procedures for reading and writing information to and from the robot's actuators, joints, and sensors.

3.3. Navigation2 in ROS2

The ROS2 system's navigation stack is a metapackage that allows robots to navigate. After receiving a goal posture from the user, the navigation stack gives the robot velocity commands to help it get there. Figure 5 shows the navigation2 structure in ROS2 system.

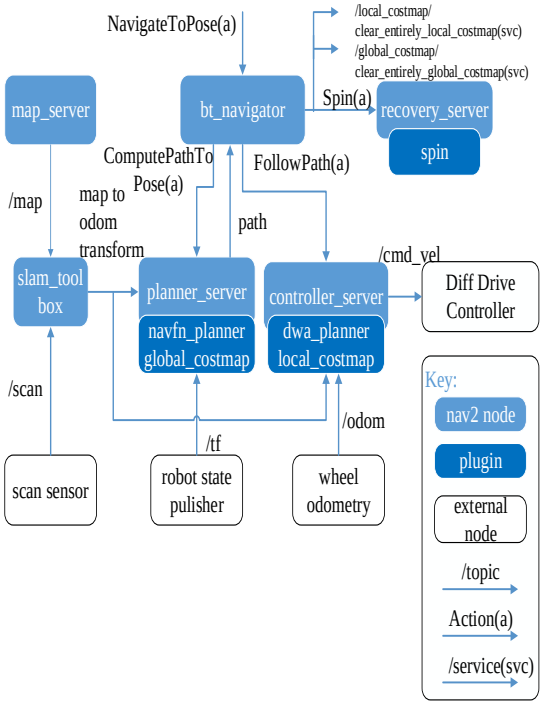


Figure 5. Navigation2 Structure in ROS2

The navigation system's output is velocity, while its inputs are map, tf, odom, and scan. The map_server loads the saved map in the navigation2 system. The "/map" topic from map_server and the "/scan" topic from the laser sensor are subscribed to by the slam_toolbox node. The map to odom transform is then published. This transformation subscribes from the planner server and controller server. Calculating the plan is the responsibility of the planner server. The robot's movements are managed by the controller server, which makes sure they adhere to the predetermined plan. The robot is assisted in recovering from a stuck state via the recovery server.

There are global and local planners in Navigation2. The global planner plans a path from the robot's current pose to the goal pose using a global map. The local planner generates velocity commands to follow that path, while avoiding obstacles in real time. The global planner and local planner work together. The global planners in Navigation2 are Nav2 Navfn and Smac planner. In this paper, Nav2 Navfn planner based on the A* algorithm is used because this algorithm is simple, reliable, and works well with grid-based costmaps, and is fast for 2D navigation tasks. The DWA planner is selected as the local planner in Navigation2 due to its adaptability, ease of adjustment, and compatibility with differential drive robot.

3.4. State Estimation

For autonomous navigation, a robot must be aware of its surroundings and position. Thus, the robot estimates the state recursively. State estimation is done using the recursive Bayes filter technique because the environment and the robot's movement are interdependent. Bayes filter algorithm calculates the belief distributions through conditional probability from measurement and control data. To estimate the robot locomotion, belief over a state variable x_t is denoted by bel (x_t) for posterior.

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \quad (1)$$

Bayes filter algorithm is obtained from Equation (1) by using Bayes rule, Markov assumption, and independence case.

$$bel(x_t) = \eta p(z_t|x_t) \int p(x_t|x_{t-1}, u_t) bel(x_{t-1}) dx_{t-1} \quad (2)$$

The two steps of the Bayes filter algorithm are the prediction step and the correction step. The prediction phase determines a belief about the state x_t by using the previous belief about the state x_{t-1} and the control u_t. The belief derived from the prediction step is multiplied by the probability of measurement z_t in the correction phase. Prediction step is described in Equation (3) and correction step is also depicted in Equation (4).

$$\overline{bel}\,(x_t) = \int p\,(x_t|x_{t-1}, u_t)\ bel\ (x_{t-1})\ dx_{t-1} \quad (3)$$

$$bel(x_t) = \eta p(z_t|x_t) \overline{bel}(x_t) \quad (4)$$

where, $p\,(x_t|x_{t-1}, u_t)$ is motion model and $p(z_t|x_t)$ is observation model

The pseudocode for basic Bayes filter is as follows:

Algorithm Bayes_filter ($bel\ (x_{t-1})$, u_t, z_t):

for all x_t do

$\overline{bel}\ (x_t) = \int p\ (x_t|x_{t-1}, u_t)\ bel\ (x_{t-1})\ dx_{t-1}$

$bel(x_t) = \eta p(z_t|x_t) \overline{bel}(x_t)$

endfor

return bel (x_t)

The Bayes filter is recursive, that is, the belief (x_t) at time t is calculated from the belief $bel\ (x_{t-1})$ at time t-1. Its inputs are the belief bel at time t-1, along with the most recent control u_t and the most recent measurement z_t. Its output is the belief bel (x_t) at time t.

3.5. Mapping

Robotic tasks like localization, planning, and recognizing the appearance of the surroundings all depend on maps. The map given the robot's pose is computed by the Equation (5).

$$m^* = \text{argmax}_m\ P(m|x_1, z_1, \dots, x_t, z_t) \quad (5)$$

where, m is map, u is control command, z is observation (sensor), x is robot's pose.

Robot creates a map by using occupancy grid map in this system. The occupancy grid mapping algorithm is represented by Equation (6) or Equation (7).

$$L(m_i \mid z_{1:t}, x_{1:t}) = L(m_i|z_t, x_t) + L(m_i|z_{1:t-1}, x_{1:t-1}) - L(m_i) \quad (6)$$

$$L_{t,i} = \text{inv\_sensor\_model}(m_i, z_t, x_t) + L_{t-1,i} - L_0 \quad (7)$$

$L_{t,i}$ is the log odd representation of occupancy grid map in current time

$L_{t-1,i}$ is the recursive term for occupancy grid map

L_0 is the prior of occupancy

3.6. Localization

Mobile robot localization is the determination of the robot's position relative to a map of the environment given the motion model and observational model. There are two types of motion model: odometry-based model and velocity-based model (dead reckoning). An odometry-based model that calculates the new pose based on the encoder values is used for robot motion in this paper. Figure 6 illustrates the odometry-based model.

The odometry information of motion is represented as u= ($\delta_{rot1}, \delta_{trans}, \delta_{rot2}$).

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2} \quad (8)$$

$$\delta_{rot1} = \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \quad (9)$$

$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1} \quad (10)$$

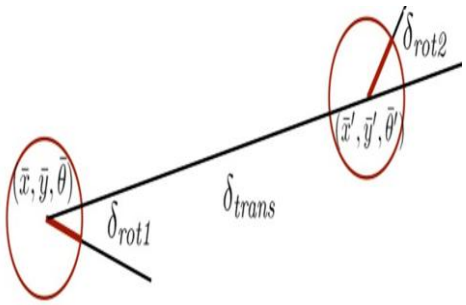


Figure 6. Odometry-based Model

Two rotations and one translation are computed to determine the rotation and distance between two poses by using an odometry-based model for a differential drive robot. Errors may appear at the front and end of the target position when the odometry model is applied. Therefore, adding a Gaussian distribution can reduce errors in one translation and two rotations.

The measured motion is given by the true motion corrupted with noise.

$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1|\delta_{rot1}|+\alpha_2|\delta_{trans}|} \quad (11)$$

$$\hat{\delta}_{trans} = \delta_{trans} + \varepsilon_{\alpha_3|\delta_{trans}|+\alpha_4(|\delta_{rot1}|+|\delta_{rot2}|)} \quad (12)$$

$$\hat{\delta}_{trans} = \delta_{trans} + \varepsilon_{\alpha_3|\delta_{trans}|+\alpha_4(|\delta_{rot1}|+|\delta_{rot2}|)} \quad (13)$$

$\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are the robot-specific error parameters, which specify the error accrued with motion.

ε is a zero-mean noise variable

$$\varepsilon_{\sigma^2}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{x^2}{\sigma^2}} \quad (14)$$

x is the query point

σ is the standard deviation

The final position possibility of robot by using odometry motion model is calculated by the following equations.

$$x' = x + \hat{\delta}_{trans} \cos(\theta + \hat{\delta}_{rot1}) \quad (15)$$

$$y' = y + \hat{\delta}_{trans} \sin(\theta + \hat{\delta}_{rot1}) \quad (16)$$

$$\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2} \quad (17)$$

The laser range finder is used for the observation model. A combination of four density models, such as measurement noise, unexpected obstacles, random measurement, and maximum range, is used by laser range finders to determine the distance to nearby objects. The observation model is defined as a conditional probability distribution $p(z_t|x_t, m)$, where x_t is the robot pose, z_t is the observation at time t and m is the map of environment.

$$p(z_t|x_t, m) = \prod_{i=1}^{k} p(z_t^i|x_t, m) \quad (18)$$

The observation model for perceiving landmarks in the map is described by using Equation (19).

$$\begin{pmatrix} r_t^i \\ \emptyset_t^i \end{pmatrix} = \begin{pmatrix} \sqrt{(m_{j,x}-x)^2+(m_{j,y}-y)^2} \\ \operatorname{atan2}(m_{j,y}-y, m_{j,x}-x)-\theta \end{pmatrix} + Q_t \quad (19)$$

In Equation (19), range bearing $z_t^i = (r_t^i, \emptyset_t^i)^T$, where r is distance and ϕ is angle. Robot pose is given by $(x, y, \theta)^T$ and Q_t is Gaussian noise with covariance.

4. SYSTEM FLOW CHART

Flow chart of the autonomous mobile robot is shown in Figure 7. Firstly, a robot model is created, and a four-room model is built in Gazebo. The robot is moved by adding ROS2 control. The user must then decide between

navigation and mapping. The mapping process uses odometry and lidar data with the SLAM technology to produce a map when the user commands the robot's velocity. This map is then stored. The map server loads this saved map for the navigation process, and every server in the navigation system is set up. The planner server determines the trajectory to pose when the user submits the goal from RViz, and the controller, smoother, and collision servers manage the robot's movement according to the plan the planner server has calculated by avoiding obstacles. After that, the robot travels to its destination.

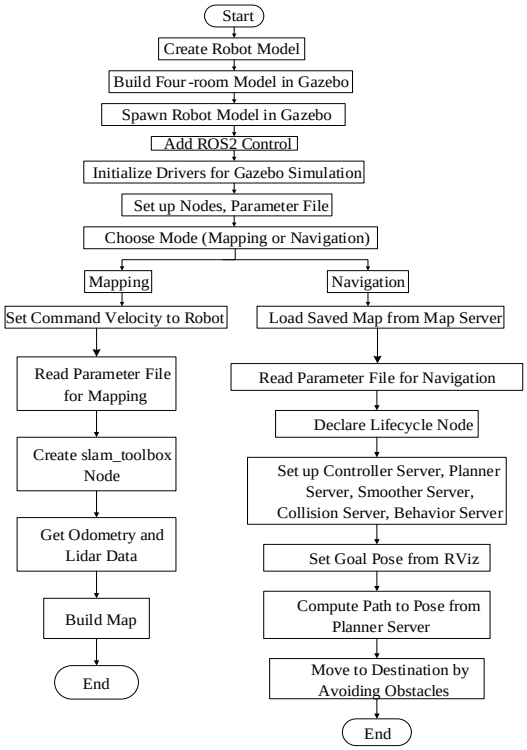


Figure 7. Flow Chart of the System

The collision monitor in Nav2 modifies to prevent dynamic obstacle collisions if the robot encounters a dynamic obstacle while it is traveling. The source_timeout in the collision monitor is five seconds. The recovery process is carried out in the event that the navigation process fails. Three recovery fallback actions are available in the recovery server. The three actions are cleaning both local and global costmaps, spinning, and waiting. The robot goes back to the navigation procedure and advances to the goal if the first action is adequate. The rotating action is triggered if the initial action is unsuccessful. The maximum spinning action is 1.57 meters. The waiting action is carried out in the event that the spinning action is unsuccessful. In this operation, the robot waits five seconds. If waiting is insufficient, the robot uses a behavior tree to inform the user that the goal was not achieved. Robot uses planner and controller servers to navigate to its destination if waiting is adequate. The robot tells the user, "Goal Succeeded," after it has arrived at its location.

5. TESTS AND RESULTS

Gazebo is utilized to simulate the autonomous mobile robot. In the Gazebo, a four-room model (5.8 m * 8.9 m) and a robot model with a laser sensor, two base-link wheels, and three plates are constructed. Figure 8 shows the environment and the robot model positioned in the middle of it.

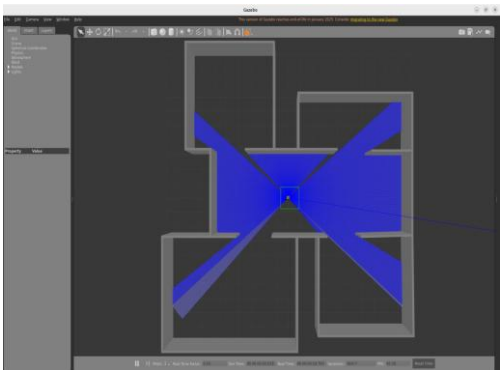


Figure 8. Working Environment of Robot in Gazebo

Firstly, the ROS2 system must broadcast the "/scan" topic from the laser_controller node and the "/tf" topic for the robot's position and orientation in order to compute the mapping process. Figure 9 shows the robot and laser rays visualization in RViz.

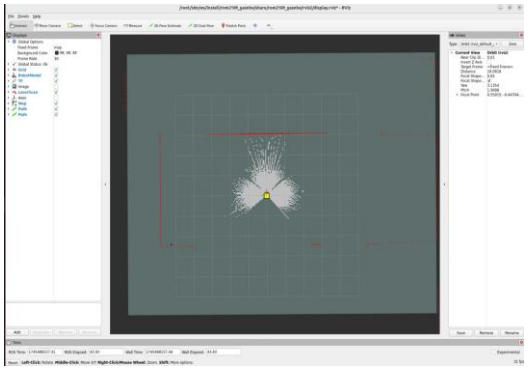


Figure 9. Visualization of Robot and Laser Rays in RViz

The rqt_graph for the mapping procedure is shown in Figure 10. The laser_controller node's "/scan" topic is subscribed to by the slam_toolbox node. The occupancy grid mapping algorithm and the recursive Bayes filter are used by the slam_toolbox node to compute the mapping and localization process. Then, a map is established.

The twist_mux node is used to move the robot. This node subscribes to any topic related to velocity, separates velocity according to priority, and generates the "/cmd_vel_out" topic when it is published. However, in this system, this topic is remapped as "/diff_cont/cmd_vel_unstamped." After the diff_cont node subscribes to the "/diff_cont/cmd_vel_unstamped" topic, ROS2 control is used to determine the robot's position. The "/joint_state" topic is published to the robot_state_publisher node by the joint_broad node in the controller manager. The position and velocity of the left and right wheel joints are included in the "/joint_state" topic. After listening to the URDF file, the robot_state_publisher node broadcasts the "/robot_description" and "/tf" topics. The left and right wheel transforms from base_link are included in the "/tf" topic. This transformation allows RViz to visualize the movement of the robot.

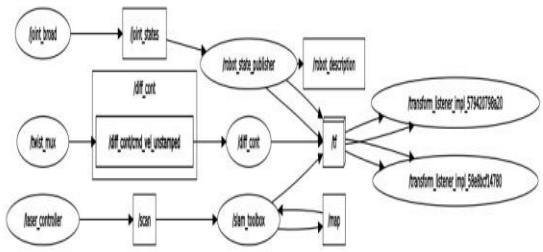


Figure 10. Rqt_graph for Mapping Procedure

The angular velocity (z) is 0.5 for left rotation and -0.5 for right rotation, whereas the linear velocity (x) is 0.8 for forward motion and -0.8 for backward motion. The system determines the robot's state by using motion and laser observation when the user enters linear or angular velocity for robot movement using rqt_publisher. Based on the robot's state and the available laser data, it then generates a 2D occupancy grid map. Creating a map is illustrated in Figure 11.

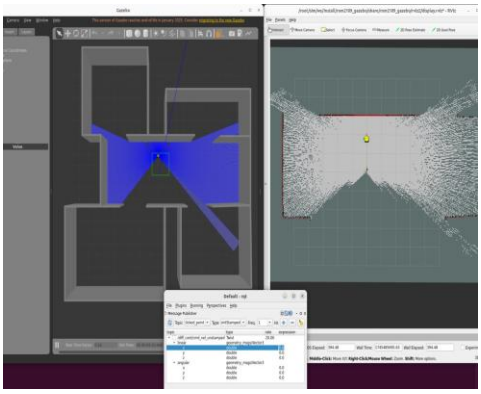


Figure 11. Creating a Map

Figure 12 depicts the final map result. White areas indicate free space, black areas represent obstacles, and grey areas show unknown space. This map is saved for autonomous navigation.

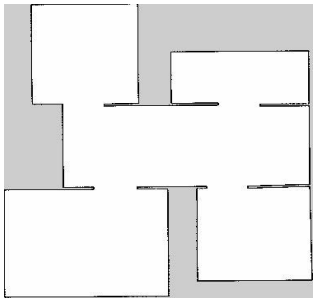


Figure 12. Final Map Result

The map server loads the saved map for the navigation procedure. The planner server computes a path to pose when the user provides a 2D goal pose via RViz, and the controller server controls the robot's movement based on the plan that the planner server has computed. The target pose is set in Figure 13. Figure 14 shows an illustration of the path planning procedure.

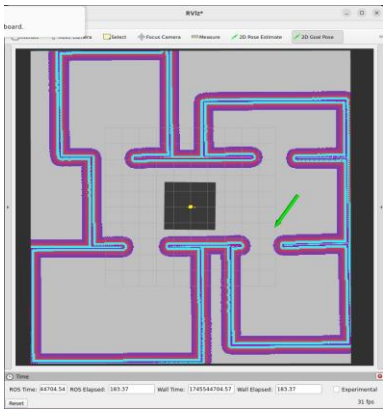


Figure 13. Setting Target Pose

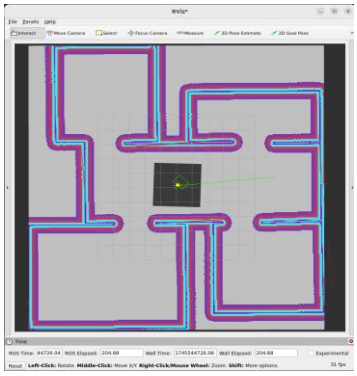


Figure 14. Path Planning to the Target Pose

The planner server determines a new path by avoiding obstacles as they are added to the Gazebo environment while the robot is moving to the desired position. Figure 15 illustrates creating a new path by avoiding obstacles to reach the desired position.

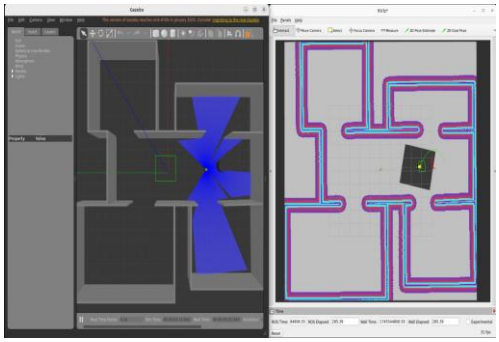


Figure 15. Creating a New Path by Avoiding Obstacle to Target Pose

6. CONCLUSION

A two-wheel differential drive mobile robot positioning and mapping system utilizing ROS2 is presented in this paper. The Linux Ubuntu 22.04-based i7 laptop runs ROS2, and Gazebo is used to create an environment within a 5.8 m * 8.9 m area. This environment has no furniture, so the obstacle density is low. However, the proposed system is partially scalable to dynamic environments if moving obstacles are sparse. This system also has a robot model with two motors, two wheels, and two casters.

For robot localization and mapping, a Lidar-based SLAM approach with a Bayes filter and an occupancy grid mapping algorithm is used. Both procedures are dependent on each other. Every computation is carried out in a C++ node that is part of the ROS network. Robot movement is controlled using ROS2. Nodes communicate with one another on topics.

The Intel Core i7 (quad-core) with 16 GB of RAM is used in this paper. This laptop's 2D Lidar-based SLAM system significantly utilizes two cores for pose graph and scan matching. On a quad-core i7 laptop, the CPU utilization is roughly 50% and the RAM usage begins at 300 MB and increases to 1 GB as the map size increases. Even though 3D SLAM requires more CPU and RAM than 2D SLAM, an i7 laptop can support both 2D and 3D SLAM. Nevertheless, 2D/3D SLAM is not supported by the Raspberry Pi 4. For 2D SLAM, embedded systems like the Intel NCU (i5/i7) or Jetson Xavier are used.

The planner server in the navigation2 package of the ROS2 system is used to carry out path planning. Navigation2 has both a global planner and local planner. Using a global map, the global planner determines a route from the robot's present stance to the desired pose. In real time, the local planner avoids barriers by generating velocity commands to follow that course. Since the A* algorithm is quick for 2D navigation tasks and performs well with grid-based costmaps, the Nav2 Navfn planner based on it was selected as the global planner. DWA is used as the local planner in Navigation2 since it is simple to tune, and a differential drive robot is employed in this proposed system. The mapping and autonomous mobile robot navigation capabilities of this system are demonstrated. In this paper, the robot can successfully navigate autonomously to the intended target when it is set from RViz.

There are many challenges for SLAM and navigation system in real-world environments. Lidar sensors misread reflective surfaces, such as polished floors, glass, and mirrors. Using RGB cameras in dimly lit areas is not advised. If large, wide areas have few characteristics, mobile robot using visual SLAM or 2D Lidar-based SLAM will have mapping errors. The above problems are solved with multi-sensor fusion. Robot aids by the combination of lidar and RGB cameras in areas with no features, dark rooms, and reflecting surfaces. Moreover, IMU and RGB camera fusion, as well as lidar and IMU fusion, are also employed depending on the situation.

ACKNOWLEDGEMENT

The author would like to express my respectful gratitude to the Rector and Pro-Rector of University of Technology (Yatanarpon Cyber City). I am special thanks to Dr. Atar Mon, Professor and Head of the Faculty of Electronic Engineering, University of Technology Yatanarpon Cyber City for her guidance and encouragement. Also, The author would also like to give special thanks to my supervisor and all teachers in Faculty of Electronic Engineering and all who willingly helped the author throughout the preparation of the paper.

REFERENCES

[1] L. Lia, L. Schulzeb, K. Kalavadia, (2024), "Promising SLAM Methods for Automated Guided Vehicles and Autonomous Mobile Robots", 5th International Conference on Industry 4.0 and S-mart Manufacturing, Procedia Computer Science 232, 2867–2874

[2] Tran Hoang Viet, Truong Xuan Nghiem, Nguyen Minh Huy, Truong Gia Binh, Vo Thanh Ha (2024), "Research of Autonomous Navigation for Mobile Robots Using Karto SLAM Algorithm Under ROS", International Journal of Research in Engineering and Science (IJRES), Vol. 12, No. 5, PP. 173-179

[3] Luiz Gustavo de Lima, Vicente Idalberto Becerra Sablón (2024), "Simulation and Modeling of Control and Sensorization Strategies for Autonomous Navigation with Differential Drive Robots:Unknown Environments Focused Approach",22nd L-ACCEI International Multi-Conference for Engine-ering, Education, and Technology, ISBN: 978-628-95207-8-1.ISSN:2414-6390. Digital Object Identifi-er:10.18687/LACCEI2024.1.1.196

[4] Dr. Rahul S. Pol , Dr. Vijaya N. Aher , Dr. Sandeep V. Gaikwad , Dr. Daulappa G. Bhalke, Dr. Ajay Yadaorao Borkar and Dr. Mahesh T. Kolte (2023), "Autonomous Differential Drive Mobile Robot Navigation with SLAM, AMCL using ROS", International Journal of Intelligent Systems and A -pplications in Engineering, IJISAE, 2024,12(5s), 46–53

[5] Bin Li, Zhengguang Ma, Yongguo Zhao (2022), "2D Mapping of Mobile Robot Based on micro-ROS", Journal of Physics: Conference Series 2402 (2022) 012030 IOP Publishing doi:10.1088/1742-6596/2402/1/012030

[6] Aniruddha Gaikwad, Mihir Kulkarni, Anuja Agnihotri , Samruddhi Purohit, Dr.Shantipal Ohol (2021), Proceedings of 6thInternational conference on Intelligen Technologies (ICIT-2021), Singapore, 17-19 December, 2021

[7] Ruchik Mishra, C. Vineel (2020), "Indoor Navigation of a Service Robot Platform Using Multiple Localization Techniques Using Sensor Fusion ", 6th International Conference on Control, Automation and Robotics, 978-1-7281-6139-6/20/$31.00, 2020 IEEE

[8] Denis Chikurtev and Sofia (2020), "Mobile Robot Simulation and Navigation in ROS and Gazebo", 978-1-7281-9308-3/20/$31.00, 2020 IEEE

[9] Duc-Tuan Ngo1 and Hoang-Anh Pham (2020), "Towards a Framework for SLAM Performance Investigation on Mobile Robots", 978-1-7281-6758-9/20/$31.00, 2020 IEEE

[10]Yuhai Wei, Hui Zhang, Guang Deng, Hang Zhong and LiLiu (2020), "Research on the SLAM of Mobile Robot Based on Particle Filter Method", Proceedings of 9th IEEE International Conference on CYBER Technology in Automation Control and Intelligent Systems, Suzhou, China

[11] Rajesh Subramanian (2023), Build Auton-omous Mobile Robot from Scratch using ROS, Simulation and Hardware Books.

[12] Francisco Martín Rico (2023), A Concise Introduction to Robot Programming with ROS2, ISBN: 978-1-003-28962-3 (ebk)

Author

Biography

Thet Hsu Wai is a Master of Engineering (Electronics) student at the University of Technology (Yatanarpon Cyber City), Pyin Oo Lwin. She had enrolled at the University of Technology (Yatanarpon Cyber City) in December, 2012 and received the Bachelor of Engineering (Electronics) in November, 2018. Her research interests include robotics, artificial intelligence and machine learning.

EXPLORING TRANSFORMER MODELS FOR MACHINE TRANSLATION OF MYANMAR AND DAWEI LANGUAGES

Sint Sint Shein[1]

[1]Department of Computer Science, Polytechnic University (Dawei)
sintsintshein85@gmail.com

ABSTRACT

This study introduces an innovative method for Neural Machine Translation (NMT) between Burmese and Dawei languages using a transformer-based architecture. At the moment, a total of 26,000 sentences have been gathered from textbook data and the UCSY corpus related to the Myanmar Language. To address these challenges expanded a transformer model specifically tailored to the complex structure of both languages, and use a bilingual corpora derived from various regional texts. The goal of this research is to enhance machine translation for Dawei and Burmese. By using sophisticated Transformer network designs, the goal is to increase the precision and fluency of translation across these diverse yet underrepresented languages. The study assesses different transformer-based methods and how well they work to address the particular difficulties presented by the linguistic structures of Dawei and Burma. In order to maximize hyper-parameters that are separate to the semantic and semantic aspects of the Burmese and Dawei languages, our methodology incorporates data augmentation techniques to improve the training data. This system use BLEU standard metrics. This analysis sets the foundation for additional improvements in NMT for low-resource languages and paves the process for the insertion of additional dialects in future studies.

KEYWORDS

Myanmar Language, Dawei Language Corpus, Transformer, Neural Machine Translation

1. INTRODUCTION

Newly developments in machine translation have made it much easier to communicate across linguistic boundaries, enabling previously unheard-of levels of engagement. Progression in this field has been noticeably facilitated by the creation of transformer models.

Burmese and Dawei, both vital languages in Myanmar, impact remarkable cultural and economic advantage; nevertheless, the absence of high-quality machine translation systems for these languages checks effective communication and the interchange of knowledge among their speakers.

At presents, there are not enough resources available, and there is also a lack of machine translation systems using transformers for Myanmar-Dawei language pairs. The Dawei Language is one of the ethnic languages of Myanmar and serves as the native tongue of the Dawei people, functioning as their primary means of the communication.

This paper proposes word-level segmentation in both translation directions to enhance transformer-based neural machine translation for the Myanmar-Dawei language pair. Myanmar's official language is Burmese, which is spoken by about 33 million people. With about 120,000 speakers, Dawei is the most common language in the Tanintharyi region. Despite the fact that these languages are important, there are not many resources accessible for machine translation.

2. RELATED WORK

In many studies, it can be observed that the evaluation of Neural Machine Translation systems includes assessments of the BLEU and RIBES scores [1]. The authors introduce a novel network architecture, known as the Transformer (large model),

which relies on attention mechanisms and the Standard WMT 2014 English-German dataset consisting of 4.5 million sentences, encoded through byte-pair encoding and trained on the more extensive WMT 2014 English-French dataset containing 36 million sentences and 300,000 NVIDIA P100 GPUs (3.5 days).

The findings indicate that the translation performance from English to German was 28.4 BLEU, whereas the translation performance from English to French was 41.0 BLEU [2]. The writer examines The Indian Centre for Language Technology, or IIT Bombay CFILT, has created and evaluated a 1.5 million parallel record multi-domain English-Hindi dataset that includes news domains.

The three primary processes of data pre-processing are vocabulary generation, duplicate elimination, and data cleansing. Ten distinct configurations five for word and sub-word level tokenization are used to train the Transformer model and back-translation procedures. The batch size has been configured to match the effective batch size of 64 records. The runtime of the NVIDIA Tesla K80 GPU available in Google Colab estimates a training duration of 20-24 hours for each configuration. The BLEU and RIBES scores for the Transformer model utilizing word level tokenization are 21.22 and 0.728683, while the BLEU and RIBES scores for the Transformer model using word level are 24.53 and 0.735781, respectively.

3. DAWEI LANGUAGE (Tavoyan)

The Union of Myanmar is the home country of the Dawei people. The pronunciations of people in the country and the city are a little different. The text provides a synopsis of the history of Dawei and the ancient city.

The Andaman Sea borders the Tanintharyi Region, formerly known as the Tenasserim, to the west, Thailand to the east and south, and Mon State to the north. Its capital is Dawei in southeast Myanmar. Words and sounds in the Dawei language are different from those in Myanmar.

For example, Dawei Language is **“ကေ့ကေ့”** is Myanmar Language is **“ကြည့်လိုက်”** and **“ကွိုးဂန်း”- ကျိုးကန်း”**.

Unlike Standard Burmese, the Tavoyan dialect makes sense when it comes to family phrases and expressions of regard. Myanmar sentences were initially gathered before to the word segmentation phase.

Table 1. Some Myanmar ⇨Dawei Language pair

Myanmar Language	Dawei Language
ခဏလေး	ရှစ်သား (a moment)
ခဏလာပါအုန်း	ရှစ်သားလာအူး
အုန်း	အူး
လာပါအုန်း	လာအူး (please come)
သလား	လော်
နေကောင်းလား	နေကောန်းလော် (How are you?)
အမလေး	မိဝေး

အမလေး၊ ချမ်းလိုက်တာ	မိုးေ ၊ ရွမ်းေမား (Oh! very cold!)
ထမင်းစားကြစို့	မှန်းစားကေ့ဖို့
မင်းဘယ်နေရာမှာလဲ	နန်ဖယ်နေရာမာနူး

4. Proposed System Architecture

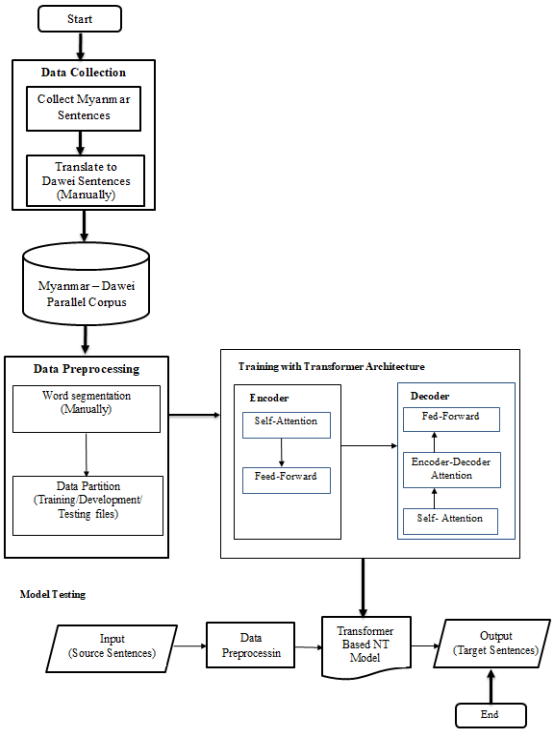


Figure 1: The Proposed System Architecture

5. Methodology

Transformer Based Neural Machine Translation has recently begun employing the Myanmar-Dawei language pair, and there is no recorded research available. Based on the stated issues, advancements in the area of Myanmar NLP research still require the development of neural machine translation systems [1].

5.1 Transformer Model

Elaborate recurrent or convolutional neural networks that incorporate an encoder and a decoder form the basis of the most widely-used sequence transduction models. The best-performing models also employ an attention mechanism to connect the encoder and decoder. The Transformer model is a novel, straightforward network design that relies entirely on attention mechanisms, eliminating the need for recurrence for convolutions.

Additionally, the Transformer based NMT model is a neural architecture that uses feed-forward neural networks and self-attention mechanisms in an encoder-decoder framework to translate text.

This enables parallel processing and efficient management of long-range dependencies to generate translations of high quality [10].

5.2 Encoder and Decoder Model

Each encoder comprises two primary components: a feed-forward neural network and a self-attention mechanism. To generate a collection of output encodings, the self-attention mechanism assesses the importance of the input encodings from the prior encoder. Each output encoding is subsequently handled individually by the feed-forward neural network.

Ultimately, the following encoder takes these output encodings as its input and decoder. Each decoder consists of three main components: a feed-forward neural network, an attention mechanism applied to the encodings, and a self-attention mechanism. The decoder operates similarly to the encoder, but it includes an extra attention mechanism that extracts pertinent information from the encodings produced by the encoders.

Similar to the first encoder, the first decoder accepts as inputs the output sequence's embedding and positional information instead of encodings. As the transformer must not utilize the present or upcoming output sequence, it must be

partially concealed to stop this reverse information transmissions. The last decoder is followed by a final linear transformation and soft-max layer to produce the output probabilities across the vocabulary.

5.3 Attention Mechanisms

Attention mechanisms are the core of the Transformer architecture, allow the model to pivot on relevant sections of the sequence:

Self-Attention (Encoder and Decoder): Focuses on relationships within the sequence being processed.

Cross-Attention (Decoder): connects the output of the encoder to the processing of the decoder.

Scaled Dot-Product Attention

The following is a description of the Transformer's attention mechanism, which serves as its primary computation:

Attention (Q,K,V) $= softmax\left(\frac{Q K^{T}}{\sqrt{d_k}}\right)$ V

Multi-Head Attention

Multi-head attention allows the model to focus on various sections of the input at the same time. It is computed as:

Multi-Head (Q,K,V)
$= Concat\ (head1, head2, \ldots\ldots, headh)$ W^{o}

Position-Wise Feed-Forward Network (FFN)

A feed-forward network is individually traversed by each place in the sequence:

FFN(x) $= ReLU\ (xW1 + b1)$ W_2+b_2

Where each attention head is computed as:

$head_i = Attention\ (QW_i^Q,\ KW_i^K, VW_i^V)$

Encoder-Decoder Attention

The attention mechanism in the decoder that connects the processing of the decoder to the output of the encoder is:

CrossAttention(Q,$K_{encoder}$,$V_{encoder}$)=softmax $\left(\frac{QK_{encoder}^{T}}{\sqrt{d_k}}\right)$ $V_{encoder}$

Output Probabilities

The decoder produces the final output by projecting its embedding's to the vocabulary size and applying a soft-max:

$$P(y_t|y < t, x) = \text{softmax}(zW^p + b^P)$$

Combined Objective Function

In order to minimize the cross-entropy loss, the model is trained in the middle of the predicted and true target sequences:

$$l = -\sum_{t=1}^{T} \log P(y_t|y < t,\ x)$$

6. Dataset and Pre-processing

A key necessity for a machine translation system is the presence of parallel corpora in both the source and target languages. At this stage, Myanmar-Dawei parallel corpus are manually collected by translating the Myanmar sentences from the text books [4] [5][6] and UCSY corpus into Dawei language. Firstly, the proposed system collects Myanmar-Dawei parallel sentences manually by translating the Myanmar sentences from the text books into Dawei language.

At the present, 26K Myanmar-Dawei parallel corpus has been collected. Data preparation, model selection, training, and evaluation are the phases involved in creating a transformer model for machine translation using Dawei data. And transformer-based Myanmar-Dawei NMT model are proposed at the word level in both translation directions.

Table 2. Statistics of the parallel sentences

Domain	No. of parallel sentences
Text books	20160
UCSY-corpus	6000
Total	26160

7. Evaluation Metrics

Translation output is assessed using Bilingual Evaluation Understudy (BLEU), a de facto standard automated evaluation matrix. BLEU n-gram precision pn is calculated by aggregating the n-gram counts for each hypothesis sentence S in the test corpus C. Mathematical Formulation: Precision,

$$p_n = \frac{\sum_{S \in C} \sum_{ngram \in S} Count_{matched}(ngram)}{\sum_{S \in C} \sum_{ngram \in S} Count(ngram)}$$

This is how the final BLEU score is determined:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Here, n denotes the order of n-grams taken into account for pn, while wn indicates the weights allocated for the precisions of n-grams; these weights are evenly distributed. A BLEU score can vary between 0 and 1, with values nearer to 1 suggesting that a proposed translation is more similar to the reference translation [9].

N-Gram is a word prediction algorithm using probabilistic method to predict next word after observing N-1 words. The computing the probability of the next word is closely related to computing the probability of a sequence of words. N-Gram of sentence probabilities:

Reference: ကျနော်နဲ့ လေ သူ့ အိမ် ဟိုဟာ ပတ်တိုင်း သွား ဟုယ် ။

Predicted: သူတို့ ကျနော်နဲ့ လေ အိမ် ဟိုဟာ ပတ်တိုင်း လာ ဟုယ် ။

Bi-gram for References:
(ကျနော်နဲ့,လေ),(လေ,သူ့),(သူ့,အိမ်),(အိမ်,ဟိုဟာ),(ဟိုဟာ,ပတ်တိုင်း),(ပတ်တိုင်း,သွား),(သွား,ဟုယ်)

Bigram for Predicted:
(သူတို့,ကျနော်နဲ့),(ကျနော်နဲ့,လေ),(လေ,အိမ်),(အိမ်,ဟိုဟာ),(ဟိုဟာ,ပတ်တိုင်း),(ပတ်တိုင်း,လာ),(လာ,ဟုယ်)

Precision:
=0/7+1/7+0/7+1/7+1/7+0/7+0/7
=0.42

8. Experimental Results

The following outcomes can be regarded as experimental findings based on the Myanmar-Dawei Machine Translation research. The training data (26160), test data (1300), and validation data (2600) utilized in this study of the suggested Myanmar-Dawei Machine Translation are shown in Figure

The batch size (4096) is being used for the BLEU score analysis. The experimental results that have been described are assessed using a small sample dataset. Table 3 compares the neural machine translation models from Myanmar and Dawei. In the future, the increased volumes of the Myanmar-Dawei parallel corpus will be utilized for the proposed system.

Table 3: Evaluation Results (BLEU) of Myanmar-Dawei NMT System

Model	Myanmar → Dawei (BLEU)
Model.Myan-Dawei-1000.pt	**28.40**
Model.Myan-Dawei-2000.pt	**10.45**
Model.Myan-Dawei-3000.pt	**13.25**

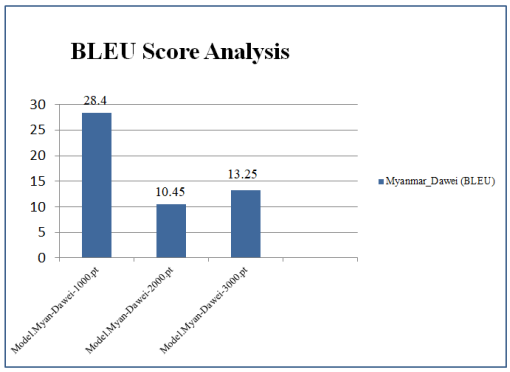


Figure 2. BLEU Score Analysis

9. CONCLUSIONS

Machine translation refers to the method of employing a computer-based system to automatically convert text from one language to another. Neural Machine Translation is still needed for the development of Natural Language Processing research field in Myanmar. Dawei Language is classified as one of the ethnic languages in Myanmar and is also considered a low-resource language. Furthermore, there are not only insufficient resources, but there is also no existing machine translation system using neural methods for the Myanmar-Dawei language pair.

Currently, the construction of the Myanmar-Dawei parallel data is an important task in machine translation, so the construction of the Myanmar-Dawei dataset will continue. For that reason, firstly, Myanmar- Dawei parallel corpus (around 26K) is generated and manual segmented exactly in this research. And this research proposes a Transformer model to translate text for Myanmar-Dawei (Tavoyan) language pair.

REFERENCES

[1].Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

[2].Hindi to English: Transformer-Based Neural Machine Translation Kavit Gangar, Hardik Ruparel, and Shreyas Lele Veermata Jijabai Technological Institute, Mumbai, India arXiv:2309.13222v1 [cs.CL] 23 Sep 2023.

[3].Amit Kumar and Anil Kumar Singh. 2019. NLPRL at WAT2019: Transformer-based Tamil-English Indic Task Neural Machine Translation System. In *Proceedings of the 6th Workshop on Asian Translation*, pages 171–174, Hong Kong, China association for Computational Linguistics.

[4].https://www.citymall.com.mm/citymall/Modern-900-English-Conversation-Audi Cd/p/cmhl_1000000076702_1

[5].https://www.shop.com.mm/products/how-why-english-grammar-i116547190.html

[6].https://www.myanmaronlinesales.com/MyanmarBooks/BookDetails/31898/

[7].Hide Ki Suzuki, Tsutomu Hire, KevinDuh, KatsuhitoSudoh, and Hajime Tsukada, 2010. "Automatic Evaluation of Translation Quality for distant language pairs". In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 944-952, Cambridge, Association for Computational Linguistics.

[8].Attention-based syllable level neural machine translation system for Myanmar to English Language pair, Yi Mon Shwe Sin1 and Khin Mar Soe2, International Journal on Natural Language Computing (IJNLC) Vol.8, No.2, April 2019.

[9].https://github.com/bangoc123/BLEU

[10].https://www.tutorialspoint.com/gen-ai/multi-head-attention-in-transformers.htm

Biography

Sint Sint Shein received the B.C.Sc(Hons.) and M.C.Sc. degrees from the University of Computer Studies, Dawei, in 2006 and 2010, respectively. Currently, she is working in the Department of Computer Science at Polytechnic University (Dawei). Current research publications include three national papers and one IEEE.

CHARACTERIZATION OF DRAGON FRUIT DYE AS A LIGHT-HARVESTING MATERIAL IN NATURAL DYE-SENSITIZED SOLAR CELLS

Chit Myo Naing[1], and Zan Mo Mo Aung[2]

[1] Department of Engineering Physics, Myanmar Aerospace Engineering University
[2] Faculty of Computer Science, University of Computer Studies, Mandalay
chitmyonaing1992@gmail.com,zanmomoaung@gmail.com

ABSTRACT

The growing environmental concerns and health risks associated with synthetic dyes have led to increased interest in natural dye alternatives derived from renewable resources. Natural dyes may be obtained from a variety of natural sources, including plants, fruits, flowers, and leaves. These offer a cost-effective and eco-friendly alternative to synthetic dyes. Moreover, studies have found that synthetic dyes may release harmful substances that can cause allergies, cancer, and other health problems. Fruits are used to extract natural dyes because they contain high levels of pigments such as anthocyanins and betalains, which produce vibrant and appealing colors. In this study, a natural dye is extracted from the peel of red-fleshed dragon fruit (Hylocereus polyrhizus), which is known to contain high concentrations of betacyanin pigments. The optical properties of the extracted dye solution are analyzed to evaluate its potential application in dye-sensitized solar cells (DSSCs). Ultervilolet-visible (UV-Vis) spectroscopic measuremet are conducted to investigate the optical properties of the natural dye. Moreover, two different approaches are employed to determine the values of the band gap energy.

KEYWORDS

Dragon fruit, Hylocereus polyrhizus, Natrual Dye,DSSCs

1. INTRODUCTION

Petroleum, natural gas, and coal are fossil fuels that are widely used as the primary energy sources around the world. In addition, burning fossil fuels releases large amounts of carbon dioxide, which pollutes the environment and alters climate conditions. Solar, wind, hydropower, biomass, and geothermal are the main sources of renewable energy, each with its own advantages and disadvantages [1]. Solar energy is a fundamental and easily accessible renewable energy source today. It provides energy for the growth and development of all living things on Earth through the process of photosynthesis. However, it also differs by the geographical location. The key advantage of solar energy is its ease of use at both household and industrial levels. Solar energy can be directly converted into useful heat or used to generate electricity [2].

A photovoltaic energy conversion is transformed sunlight directly into electrical power. In Brazil, the researchers examined the Energy Payback Time (EPBT) and carbon dioxide emission rates of rooftop crystalline silicon photovoltaic systems across various geographical regions with differing levels of solar irradiation. According to their analysis results, they showed that a 1.2 kW residential PV system in Brazil can compensate for its energy consumption over its lifetime while generating relatively low levels of CO_2 emissions [3]. The development of solar cells is categorized into three generations. First-generation solar cells include

monocrystalline and polycrystalline silicon cells, known for high efficiency and long-term stability but at relatively high production costs [4]. Second-generation solar cells involve thin-film technologies, such as amorphous silicon (a-Si), cadmium telluride (CdTe), and copper indium gallium selenide (CIGS), which offer reduced material usage and lower manufacturing costs [5]. The first dye-sensitized solar cell was made by O'Regan and Grätzel in 1991. Third-generation solar cells aim to enhance efficiency and reduce costs through advanced technologies, including dye-sensitized solar cells (DSSC), perovskite solar cells, and organic photovoltaics, often using novel materials and innovative designs [6]. In this study, dye-sensitized solar cells (DSSCs) are fabricated using natural dyes extracted from *Hylocereus polyrhizus* (Pitaya), commonly known as dragon fruit, which served as the light-harvesting sensitizer.

2. MATERIALS AND METHODS

2.1. Raw Material Collection

The selection of dragon fruit-based dye was influenced by its local availability and rich pigment content, especially in red-fleshed varieties [7]. Originally native to Central America, dragon fruit is now extensively cultivated in Asian countries and other tropical and subtropical regions around the world [8]. Its successful commercial cultivation depends on warm, humid climates. There are three main commercially grown varieties of dragon fruit:

1. Red skin with red flesh (*Hylocereus polyrhizus*)
2. Red skin with white flesh (*Hylocereus undatus*)
3. Yellow skin with white flesh (*Selenicereus megalanthus*)

All varieties are characterized by pulp containing numerous tiny black seeds, which are edible along with the flesh. Among these, the red-fleshed type is especially promising for solar cell applications due to its high concentration of betacyanin pigments, which exhibit strong light absorption and antioxidant properties, making them effective for use in DSSC sensitization.

2.2. Preparation of Liquid Dye Extract from Red-Fleshed Dragon Fruit

The dragon fruit was thoroughly washed with clean water to remove any dirt and surface impurities. The peel and pulp were then separated using a clean stainless steel knife. In this study, only the peel was used for dye extraction, as it contains a higher concentration of natural pigments.

2.3. Dye Extraction Process

10 ml of red-fleshed dragon fruit liquid is mixed with 100 ml of ethanol and methanol at three different temperatures: room temperature, 45 °C, and 60 °C. These mixtures were stirred for one hour with magnetic stirrer to get homogeneous in color. The filtrates were extracted using filter paper. These filtrates can be used as a natural dye sensitizer. The extraction of dye solution from red-fleshed dragon fruit is shown in Figure 1.

Figure 1. Extraction of dye from red-fleshed dragon fruit

2.4. Light Absorption Regions Relevant to DSSCs

To evaluate the light-harvesting capabilities of natural dyes in dye-sensitized solar cells (DSSCs), it is important to understand the different regions of the electromagnetic

spectrum. Natural dyes, such as those extracted from red-fleshed dragon fruit, primarily absorb light in the visible range, which is critical for generating photo-induced electrons in solar cells. Light Wavelengths such as UV, visible, and infrared light are often measured in nanometers (nm). As shown in Table 1, the electromagnetic spectrum can be categorized based on wavelength [9].

Table 1. Classification of Electromagnetic Spectrum by Wavelength

Region	Wavelength Range (nm)
Far ultraviolet	10 - 200
Near ultraviolet	200 - 380
Visible	380 - 780
Near infrared	780 - 3,000
Middle infrared	3,000 - 30,000
Far infrared	30,000 - 300,000
Microwave	300,000 - 1,000,000,000

2.5. Ultraviolet-visible (UV-vis) Spectrophotometer

UV-Vis spectrophotometer is a method used to analyze the absorbance or transmittance of a sample within the ultraviolet and visible light ranges. It records light intensity relative to the wavelength of the light source. Spectrophotometers are widely used in various disciplines such as physics, molecular biology, chemistry and biochemistry. To measure the optical properties of natural dyes, UV-Vis absorption spectrophotometer (GENESYS 10S) is used in this study, as shown in Figure 2. These experiments were conducted at the Material Science Laboratory, University of Mandalay [10].

Figure 2. UV-Vis Spectrophotometer (GENESYS 10S)

2.6 Characterization by Extracted Dye

The red-fleshed dragon fruit dye's optical characteristics were investigated using a UV-Visible (UV-Vis) spectrophotometer. Measurements were taken within the 350 nm to 800 nm wavelength range, showing distinct absorption peaks associated with chromophoric groups in the extract. This examination is essential for evaluating the dye's effectiveness as a sensitizer in dye-sensitized solar cells (DSSCs), as it highlights its ability to absorb visible light efficiently.

3. EXPERIMENTAL RESULTS

3.1. Impact of Temperature on Light Absorption Peel Extract

The absorption spectra of temperature controlled dye solutions extracted from dragon fruit were shown in Figure 3 and Figure 4.

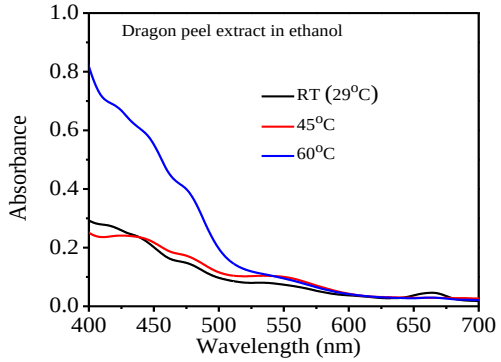


Figure 3. Effect of Temperature Variation on Dragon Fruit Peel Extract in Ethanol

Figure 2 displays the UV-Vis absorbance spectra of dragon peel extract in ethanol measured at three different temperatures: Room Temperature (RT, 29 °C), 45 °C and 60 °C. It exhibits strong absorbance primarily in the 400 nm to 500 nm range, which corresponds to the visible region. This result shows that the higher extraction temperatures (up to 60 °C) enhance the dye's light-absorbing capability. The absorption band ranges from 400 nm to 700 nm and peaks at the λ_{max} is the absorption maximum of 550 nm.

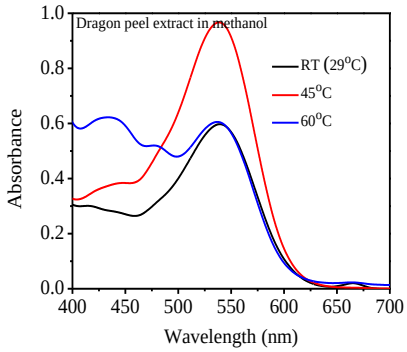


Figure 4. Absorption Spectrum of the Different Temperatures in Methanol for Red-fleshed Dragon fruit

In Figure 3, it shows the UV-vis spectra of dragon fruit peel extract at room temperature (29°C), 45°C, and 60°C. The extract heated at 45°C exhibits a maximum absorption peak at around 540 nm to 550 nm, which falls within the visible light range.

This suggests that 45°C is the optimal temperature for dye extraction, as it leads to stronger light absorption. At room temperature, absorption is relatively low, while at 60°C, the dye begins to decompose, resulting in reduced absorption. Therefore, 45°C is the best temperature for preparing the dye for use in solar cells. The absorption band ranges from 400 nm to 700 nm and peaks at the λ_{max} is the absorption maximum of 550 nm.

Anthocyanin is a group of natural phenolic compound found in plant, flowers and fruits at 45ºC in methanol, mixing of ethanol and methanol is the optimum extracting temperature for dragon fruit dye solution.

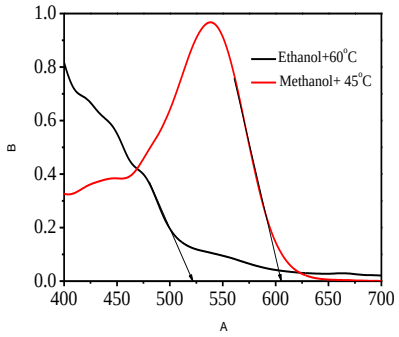


Figure 5. Absorption Spectrum of the different Temperatures in Methanol and Ethanol for Red-fleshed Dragon Fruit

Figure 5 exhibit that the efficiency of extracting natural dyes from dragon fruit peel is tested using methanol at 45 °C and ethanol at 60 °C in the range of 400 nm to 700 nm. These temperatures are chosen to facilitate efficient extraction without degrading the heat-sensitive pigments. Interestingly, methanol at the lower temperature of 45 °C yielded a higher concentration of pigment compounds than ethanol at 60 °C. This improved extraction is likely due to methanol's greater polarity, which enhances its ability to dissolve and extract the hydrophilic betacyanin pigments effectively, even at a lower temperature. Conversely, although ethanol is used at a higher temperature, its relatively lower polarity may have limited its extraction efficiency. These findings highlight the importance of selecting an appropriate solvent and temperature combination to maximize pigment yield while minimizing thermal degradation of sensitive compounds.

3.2. Band Gap Energy

A UV-Vis spectrophotometer was used to measure the wavelength of light absorbed, allowing for the analysis of absorbance levels within the visible spectrum and the evaluation of the dye's color intensity components [11]. The band gap of dye determines by using the formula in equation (1). The numerical value of the symbols

are: h=6.63 × 10^{-34} Js, c =3.0 × 10^{8} m/s, 1 eV = 1.60 × 10^{-19} J.

$$E = h\nu = hc / \lambda \quad (1)$$

where h is the Planck's constant,

ν = the frequency

c = the speed

λ = the wavelength

E = photon energy

Table 2. Band Gap Energy of Red-Fleshed Dragon Fruit Dye in Various Solvents at Different Temperatures

Dye	Extract Solvent	Extract Temperature	E (eV)
Red-fleshed Dragon Fruit	Ethanol	RT(29°C)	3.11
		45°C	3.11
		60°C	2.3
	Methanol	RT(29°C)	2.28
		45°C	2.28
		60°C	2.86

The band energy values of Red-Fleshed Dragon Fruit Dye solution was calculated by using equation (1). From this result, it was observed that the value of band gap of dye for ethanol is 3.11 eV, 3.11 eV and 2.3 eV at RT (29°C), 45°C and 60°C respectively. Besides the band gap values of dyes for methanol and ethanol are 3.11eV and 2.39eV at temperature 60°C and 45°C in Table 2.

3.3. Absorption Coefficient

The absorption coefficient indicates how deeply light of a specific wavelength can penetrate a material before being absorbed. The absorption coefficient for each wavelength is calculated by dividing the absorbance by the corresponding wavelength, as shown in equation (2).

$$\text{Absorption coefficient, } \alpha = 4\pi k/\lambda \quad (2)$$

where λ (nm) = taken from the cutoff wavelength of the dyes and k = the Boltzman constant with value of 8.617 × 10^{-5} eV/K.

Table 3. Peak Absorption Wavelength and Absorption Coefficient of Red-Fleshed Dragon Fruit Dye

Extract Solvent	Extract Temperature	Peak absorption wavelength	α (km^{-1})
Ethanol	29°C	540	2.004 × 10^{-6}
	45°C	400	2.705 × 10^{-6}
	60°C	400	2.705 × 10^{-6}
Methanol	29°C	545	1.985 ×10^{-6}
	45°C	545	1.985 ×10^{-6}
	60°C	435	2.487 ×10^{-6}

Optical properties of Red-Fleshed Dragon Fruit Dye showed narrow band of absorbance from 400 nm to 700 nm. The values of absorption coefficient of dye in ethanol were obtained 2.004×10^{-6}, 2.705×10^{-6} and 2.705×10^{-6} at different temperatures (29°C, 45°C and 60°C) respectively in Table 3. In methanol solvent, the absorption coefficient values were 1.985×10^{-6}, 1.985×10^{-6} and 2.487×10^{-6} at different temperatures. This result showed strong absorbance in 400 nm to 700 nm wavelength and above 700 nm with little reduced absorbance.

4. CONCLUSIONS

In this study, the red-fleshed dragon fruit (Hylocereus polyrhizus) is identified as an effective natural sensitizer for DSSCs due to its rich concentration of betacyanin pigments, which absorb visible light efficiently. Dye extracted from the fruit's peel is not only environmentally safe and

affordable but also binds well to TiO_2 surfaces, enhancing its suitability for solar energy applications. Temperature affects the stability and structure of anthocyanins in the natural pigments in dragon fruit peel responsible for light absorption. At moderate temperatures (like 45°C), these pigments are more effectively extracted and remain relatively stable, resulting in stronger light absorption. However, at higher temperatures (60°C), anthocyanins begin to degrade or decompose, leading to a reduction in absorbance. At lower temperatures (room temperature), the extraction efficiency is lower, so fewer pigments are available to absorb light.

ACKNOWLEDGEMENT

The authors would like to acknowledge the Myanmar Aerospace Engineering University's ERC committee for granting permission to submit this paper.

REFERENCES

[1] G. Richhariya, A. Kumar, P. Tekasakul, and B. Gupta, "Natural dyes for dye sensitized solar cell: Areview," *Renewable and Sustainable Energy Reviews*, vol. 69, pp. 705–718, Mar. 2017

[2] EnergySage, "What Are The Environmental Benefits of Solar Energy–And What Are Its Drawbacks?" [Online]. Available:https://www.energysage.com/solar/health-environmental-benefits-of-solar-energy/.

[3] S. H. Fukurozaki, R. Zilles, and I. L. Sauer, "Energy Payback Time and CO2 Emissions of 1.2 kWp Photovoltaic Roof-Top System in Brazil," *International Journal of Smart Grid and Clean Energy*, vol. 2, no. 2, pp. 164–169, 2013

[4] M. A. Green, "Third generation photovoltaics: Ultra-high conversion efficiency at low cost," *Progress in Photovoltaics: Research and Applications*, vol. 9, no. 2, pp. 123–135, 2001

[5] A. Shah, "Photovoltaic Technology: The Case for Thin-Film Solar Cells," *Science*, vol. 285, no. 5428, pp. 692 –698, Jul. 1999

[6] B. O'Regan and M. Grätzel, "A low-cost, high-efficiency solar cell based on dye-sensitized colloidal TiO2 films," Nature, vol. 353, no. 6346, pp.737–740, Oct. 1991.

[7] A. Mohamed and Nafarizal Nayan, "Fabrication and analysis of dye-sensitized solar cell using natural dye extracted from dragon fruit," *International Journal of Integrated Engineering*, vol. 2, no. 3, pp. 7–13, Jan. 2010.

[8] T. K. Lim, *Edible medicinal and non-medicinal plants. Volume 4, Fruits*. Dordrecht: Springer

[9] A. R. Jha, *Solar Cell Technology and Applications*. Boca Raton, FL, USA: CRC Press, 2010

[10] C. M. Naing and Z. M. M. Aung, "Optical properties of natural dyes extracted from *Rhinacanthus nasutus* leaves for dye-sensitized solar cells," *J. Inf. Technol. Res. Innov.*, vol. 4, no. 2, pp. 97–103, Dec. 2024.

[11] N. K. K. Thein, A. Htet, and T. T. Aye, "Optical Properties of Natural Dye from Strawberry Fruits as Photosensitizer for Dye-Sensitized Solar Cells Application," University of Mandalay, Research Journal, Vol.11, 2020

Authors

Biography

Chit Myo Naing received the M.Sc. degree in Physics from Banmaw University in 2016. Moreover, he also obtained D.C.Sc. degree from Computer University, Banmaw. He is also a Lecturer. He is currently with the Department of Engineering Physics, Myanmar Aerospace Engineering University (MAEU). His research interests include in embedded systems, machine learning, artificial intelligence, and security.

Zan Mo Mo Aung received the Ph.D. degree in Information Technology from University of Computer Studies, Mandalay (UCSM). She is also a Lecturer. She is currently with the Faculty of Computer Science, UCSM. Her research interests include in natural language processing, machine learning, artificial intelligence, network engineering and data mining.

SPEAKER RECOGNITION SYSTEM USING SPECTROGRAM IMAGE FEATURE AND CONVOLUTIONAL NEURAL NETWORKS

Thein Htay Zaw[1], and San Lin Aung[2]

[1,2] Department of Information Technologies Supporting and Maintenance, University of Computer Studies (Mandalay)

theinhtayzaw.cu@gmail.com, sanlinaung08@gmail.com

ABSTRACT

Various biometric security systems, such as face recognition, fingerprint, voice, hand geometry, and iris, have been developed. Apart from being a communication medium, the human voice is also a form of biometrics that can be used for identification. The voice recognition technology market continues to involve rapidly as almost all smart home devices are providing speaker recognition capability today. The system extracted spectrogram image, which is a pictorial representation of speech in terms of raw features, to identify speakers. These features were inputted into a convolutional neural network (CNN), and used to recognize the speakers. This method applied to the speech dataset available from the kaggle.com. As a result, the performance of the designed and constructed model is presented in this study. This paper aims to find a suitable self-deep learning model for speaker recognition and to provide improved speaker recognition models with observed data using CNN.

KEYWORDS

Speaker Recognition, Deep Learning, Pre-processing, Convolutional Neural Network (CNN), Matlab

1. INTRODUCTION

The need for digital data security is growing along with technological advances. Since unsecured systems are prone to risks, such as robbery, demolition, and misuse, security plays a major role in the lives of people. Typically, security systems implement methods that employ pattern, pin, or password locks. However, they exhibit certain flaws as they can be easily hacked. Therefore, security systems based on the identification of human physiological characteristics, namely biometrics, are preferred over the aforementioned methods. Speech is a common biometric used to identify a person. In comparison with other biometrics, speech-based biometrics are more cost-effective as they do not require specific hardware; moreover, the file size requirement is comparatively small [1].

Speaker identification and verification are two crucial tasks carried out by recognition using a biometric system. A person's identity is to be ascertained via an identifying system. The verification system seeks to accept or reject the asserted identification of someone [2]. Generally, biometric technology works by using pattern recognition techniques or pattern recognition. The patterns learned typically are fingerprint patterns, palm lines, faces, irises, and voice recognition [3]. Apart from being a communication medium, the human voice is also a form of biometrics that can be used for identification [4], [5]. Voice has unique characteristics that can be used as a differentiator between one person and another. A sound speaker recognition system must be able to pick up the features that characterize a person's voice. These features represent and describe the voice signal [6]. Atypical speaker recognizer consists of two main components: a feature extraction module and a speaker modelling part. The feature extraction module transforms raw audio signals into feature vectors that capture speaker-specific properties. Based on these vectors, the speaker model is trained. Which features to extract and what technique to use for the modelling are

strongly dependent on the intended application, computing resources, and amount of speech data available [7].

Based on MFCC and LPC characteristics for the Azerbaijani DataSet, Support Vector Machines were applied to the acoustic model of the Speech Recognition System in [8]. Demonstrate that SVM with a Radial Bases kernel on LPC features performs better than SVM with a polynomial kernel on MFCC features.

In [9], surveys a comparison of LPCC, MFCC and BFCC as feature extraction approaches and ANN as a classifier. Hindi Isolated, Paired and Hybrid words are used for database purpose. The results of the series of studies show that MFCC surpasses the traditional LPCC and BFCC approaches.

SVM was used to complete the connected digit recognition problem in [8]. Show that SVM-based voice recognition performs somewhat better overall than ANN on the similar data set.

To address the aforementioned drawbacks, the system implemented a simple feature, namely a spectrogram, which is an image that represents speech and is obtained from the short-time Fourier transform (STFT). STFT provides more information in terms of both frequency and time, particularly from speech input.

Raw features can be obtained directly from speech signals. In this study, we used a convolutional neural network (CNN) rather than an HMM to identify the speaker from the input spectrogram. Although CNNs are more commonly used as classifiers for image recognition, a few studies have utilized CNNs for speech recognition.

Section 2 of this article involves pre-processing. Features from this research are described in Section 3. Section 4 provides details on the suggested categorization model and the experimental implementation. Section 5 presents the results of the experiment and then assesses the CNN model using various evaluation metrics. Finally, Section 6 concludes this paper and discusses the outlook of future work.

2. PRE-PROCESSING

The first and most crucial phase in the automatic speech recognition process is speech signal pre-processing. Sampling, framing, and windowing steps are included in the pre-processing stage. In audio signals, frame length or window length segmentation is crucial because it enhances the classification performance. After pre-processing is complete, the audio output moves on to the feature extraction stage.

2.1. Sampling

Analog signals must first be transformed to sampled signals before being processed for classification. A time- and discrete-signal is the end outcome. The audio file was sampled by the system using 16000 Hertz.

2.2. Framing and Windowing

Create fixed-length frames from the audio signal segments. Each frame is multiplied using a Hamming window in order to reduce spectral distortions when the speech signal is blocked [8].

The number of frame of the signal is calculated as the following equations,

$$no\ of\ \ frame = floor\left(\frac{s(\ fLength - wLength + 1)}{wLength}\right) + 1 \quad (1)$$

$$s = \frac{1}{1 - overlaprate} \quad (2)$$

Where,

- *fLength* = the length of signal
- *wLength* = the length of window for signal
- *overlaprate* = the length of overlap rate on window for signal

2.3. Image Resizing

Image resizing is necessary to increase or decrease the total number of pixels. Whenever an image resizes, its pixels are updated. It is very important to extract features. In this system, the images were resized 150*250*3 pixel.

3. FEATURE EXTRACTION

An emphasis is made on extracting elements in speaker independent speech recognition that are somewhat resistant to changes in the

speaker. Therefore, feature extraction involves speech signal analysis [9]. Spectrograms image is extract to train and test using CNN.

3.1. Mel-spectrograms

The speech signal was extracted into a spectrogram using the STFT algorithm. Herein, the speech signals were initially blocked into frames of 25 ms with an overlap of 15 ms. As frame blocking can result in discontinuous signals, we used a Hamming window to prevent this discontinuity. Subsequently, the product of the framed signal and Hamming window was used as the input to the Fourier transform process. This STFT was used to convert the speech signal from the time domain to a frequency domain.

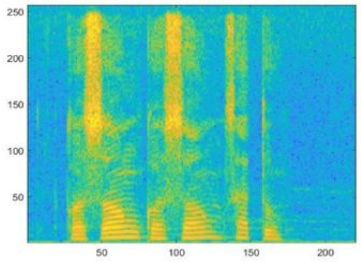


Figure1. Spectrogram

$$X[n,w] = \sum_{m=-\infty}^{\infty} x[m]w[n-m]e^{-jwm} \quad (3)$$

For real-valued inputs, positive and negative frequency components are complex conjugates of each other, such that we retain N unique units of information. However, since spectra are complex-valued vectors, it is difficult to visualize them as such. A first solution would be to plot the magnitude spectrum |Xk| or power spectrum |Xk|2. Due to large differences in the range of different frequencies, unfortunately these representations do not easily show relevant information.

The log-spectrum is the most common visualization of spectra and it gives the spectrum in decibels. It is useful because again, it gives a visualization where sounds can be easily interpreted.

To compute the periodogram estimate of the power spectrum, we apply:

$$20\log 10|X[n,w]| \quad (4)$$

The speech signals extracted from the spectrogram were saved in the image format with a size of 150*250*3 pixels as the input of the CNN. Figure.1 depicts the spectrogram sample used for the training process.

The output of the spectrum is a spectrogram, which is a 2D image representation of the speech signal, as depicted in Figure.1. This image was inputted to the CNN in the JPEG format for the training process. The model obtained in the training stage was used to identify the speaker of the tested data. The training and validation data were in the ratio 82:18.

4. CONVOLUTIONAL NEURAL NETWORK

CNN is a type of artificial neural network that is typically used to identify objects. It comprises several layers, including a convolution layer, rectified linear unit layer and pooling layer, which are used for feature learning. Furthermore, the classification is performed in the subsequent layer after flattening the input to the fully connected layer and the softmax layer. Figure.2 illustrates the architecture of CNN used for speech processing in this study.

4.1. Train Step

The system needs to build the network that trains the image dataset. In this step, features are extracted from the spectrogram image of LibriSpeech (train-clean-100) corpus. Features are detected by filters to obtain the important features for classification. The system uses Convolutional Neural Network not only in train step but also in test step.

Convolutional Neural Network (CNN) method automatically extracts the important features from the image by itself. CNN is a sequence of layers and every layer performs some unique functions on the dataset [12]. The input, hidden, and output are the layers of CNN.

In general, image is constructed as a matrix of pixels and these pixel values are given as input to input layer [13]. The output layer is a fully connected layer usually to classify the image to which class it belongs to the hidden layers are convolutional and pooling layer.

Convolutional layer condenses the input by extracting features of interest from it and produces feature maps in response to different feature detectors such as filter [11]. Outputting volume of convolutional layer (W2*H2*D2) can be computed as following formula

$$W_2 = (n+2p-f)/s+1 \quad (5)$$

$$H_2 = (n+2p-f)/s+1 \quad (6)$$

$$D_2 = k \quad (7)$$

When the system gets the features from the convolutional calculation, they are included a various pixel values. And these signal matrixes are calculated in an activation function called Rectified Linear Units (ReLUs) function to prevent the exponential growth in the computation required to operate the neural network [14]. Pooling layer helps to reduce noise feature and increase processing speed. And these layers can perform their processes repeatedly according to the author recommend to get good result. ReLU Activation Function is calculated with the following equation

$$f(x) = max(0,x) \quad (8)$$

The latest layer of CNN is fully connected layer that is the top layer of a network that collects the final convoluted feature. In this layer, the data from the pooling layer is reduced to a single dimension and each neuron is connected [10]. And the classification process is carried out by using classifier called activation functions such as Softmax function. Softmax function tests the result by the trained data for the classification. Softmax function can be defined as

$$S(y)_i = \frac{\exp(y_i)}{\sum_{j=1}^{n} \exp(y_i)} \quad (9)$$

The result of softmax function is passed into loss function to detect the difference between actual value and prediction values. The cross-entropy loss function is generated by

$$L = -\sum_{j=1}^{M} y_j \log\left(\hat{y_j}\right) \quad (10)$$

Then, it performs iterative backward passes which attempt to minimize the difference between the known correct label and the model prediction. In each backward pass, the weights move towards an optimum that minimizes the loss value and results in the most accurate prediction. Finally, the system obtains the trained model to test the result.

4.2. Test Step

This step is the final step of the system. It performs classification for the system. The tested data is tested by the trained data for the classification. Classification is the process of classify image into speaker and give accuracy percentage to the user. The classifier displays the highest probability for the predicted class. The output of the picture input is the class with the highest probability. End of the all process, it produces the output that the input image to which class it belongs to.

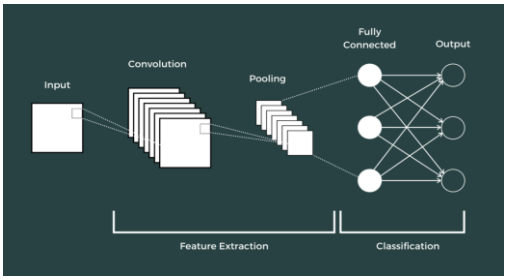


Figure2. Architecture of CNN [15]

5. EXPERIMENTS

5.1. Experimental Setup

The System recognize by three steps. They are Pre-processing step, Feature extraction step and Classification step (CNN). In Pre-processing step. The system create fixed-length frames from the audio signal segments. Frame size is 25ms. Overlapping size is 15ms. Each frame is multiplied using a Hamming window in order to reduce spectral distortions. the product of the framed signal and Hamming window was used to produced spectrogram image. The speech signals extracted from the spectrogram were saved in the image format with a size of 150 × 250 pixels as the input of the CNN.

The dataset is divided training part for 820 and test for 180. The first subset is used for optimizing/ fitting CNN model and the second subset is used for validation. The classifier was trained and tested on images of each speaker, which are speaker1, speaker2, speaker3, ……, speaker10.

The system used Conv-3, Conv-4, Conv-5 and Conv-6. Each result of the convolutional layer performs ReLUs function and pooling layer

function. So, Conv-3 calculates ReLUs function and pooling layer function three times. After convolution, the system trained by using five Fully Connected Layer. After performing the different runs of the experiment, the mean Loss function was calculated for each of the models trained with the results for each of the categories.

5.2. Corpus

The LibriSpeech (train-clean-100) corpus of read speech was created to offer speech data for the development and testing of automatic speech recognition systems as well as for the acquisition of acoustic- phonetic knowledge. LibriSpeech contains a total of approximately 100 hours long by each of 125 female speakers and 126 male speakers. This system will use 5 female speakers, 5 male speakers and (820+180) audio files to train and test.

5.3. Results and Discussion

The performance of the proposed system is evaluated by measuring the evaluation criteria on the speaker recognition. It is a great performing model. The System recognize by three steps. They are Pre-processing step, Feature extraction step and Classification step (CNN). Create fixed-length frames from the audio signal segments. Frame size is 25ms.Overlapping size is 15ms. Each frame is multiplied using a Hamming window in order to reduce spectral distortions.

These tables showed maximum, average and minimum accuracy results of ten speakers in test set.

In the first part of experiments, the system used Conv3, five Fully Connected Layers, Epochs 12, training 820 images and testing 180 images to classify speaker. The result for experiments is shown in table 1.

Table 1. The performance of CNN (Conv3)

Cov3	Sp1	Sp2	Sp3	Sp4	Sp5	Sp6	Sp7	Sp8	Sp9	Sp10
Sp1	16	0	0	0	0	0	0	1	1	0
Sp2	0	15	1	0	1	0	0	0	0	0
Sp3	1	1	14	0	0	0	1	0	0	1
Sp4	0	0	0	16	1	0	0	0	0	0
Sp5	1	0	1	1	13	1	1	0	0	0
Sp6	0	0	0	0	0	15	0	1	0	0
Sp7	0	1	0	0	0	0	15	0	0	0
Sp8	0	0	0	0	0	1	0	14	1	0
Sp9	0	0	1	0	0	1	0	0	16	0
Sp10	0	0	1	1	0	0	0	0	1	15
Accuracy	88.89	83.33	77.78	88.89	72.22	83.33	83.33	77.78	88.89	83.33
Average Accuracy										82.77

In the second part of experiments, the system used Conv4, five Fully Connected Layers, Epochs12, training 820 images and testing 180 images to classify speaker. Table 2 displays the experiment results.

Table 2. The performance of CNN (Conv4)

Cov4	Sp1	Sp2	Sp3	Sp4	Sp5	Sp6	Sp7	Sp8	Sp9	Sp10
Sp1	16	0	0	0	0	0	0	1	1	0
Sp2	0	15	1	0	1	0	0	0	0	0
Sp3	1	1	14	0	0	0	1	0	0	1
Sp4	0	0	0	15	1	0	0	0	0	0
Sp5	1	0	1	1	13	1	1	0	0	0
Sp6	0	0	0	0	0	17	0	1	0	0
Sp7	0	1	0	0	0	0	16	0	0	0
Sp8	0	0	0	0	0	1	0	15	1	0
Sp9	0	0	1	0	0	1	0	0	16	0
Sp10	0	0	1	1	0	0	0	0	1	15
Accuracy	88.89	83.33	77.78	83.33	72.22	94.44	88.89	83.33	88.89	83.33
Average Accuracy										84.44

In the third part of experiments, the system used Conv5, five Fully Connected Layers, Epochs 12, training 820 images and testing 180 images to classify speaker. Table 3 presents the experiment results.

Table 3. The performance of CNN (Conv5)

Cov5	Sp1	Sp2	Sp3	Sp4	Sp5	Sp6	Sp7	Sp8	Sp9	Sp10
Sp1	16	0	0	0	0	0	0	1	1	0
Sp2	0	16	1	0	1	0	0	0	0	0
Sp3	1	1	14	0	0	0	1	0	0	1
Sp4	0	0	0	17	1	0	0	0	0	0
Sp5	1	0	1	1	14	1	1	0	0	0
Sp6	0	0	0	0	0	16	0	1	0	0
Sp7	0	1	0	0	0	0	17	0	0	0
Sp8	0	0	0	0	0	1	0	16	1	0
Sp9	0	0	1	0	0	1	0	0	16	0
Sp10	0	0	1	1	0	0	0	0	1	15
Accuracy	88.89	88.89	77.78	94.44	77.78	88.89	94.44	88.89	88.89	83.33
Average Accuracy										87.22

In the last part of experiments, the system used Conv6, five Fully Connected Layers, Epochs 12, training 820 images and testing 180 images to classify speaker. Table 4 states the experiment results

Table 4. The performance of CNN (Conv6)

Cov6	Sp1	Sp2	Sp3	Sp4	Sp5	Sp6	Sp7	Sp8	Sp9	Sp10
Sp1	16	0	0	0	0	0	0	1	1	0
Sp2	0	16	1	0	1	0	0	0	0	0
Sp3	1	1	15	0	0	0	1	0	0	1
Sp4	0	0	0	17	1	0	0	0	0	0
Sp5	1	0	1	1	15	1	1	0	0	0
Sp6	0	0	0	0	0	17	0	1	0	0
Sp7	0	1	0	0	0	0	17	0	0	0
Sp8	0	0	0	0	0	1	0	16	1	0
Sp9	0	0	1	0	0	1	0	0	16	0

Sp10	0	0	1	1	0	0	0	0	1	16
Accuracy	88.89	88.89	83.33	94.44	83.33	94.44	94.44	88.89	88.89	88.89
Average Accuracy										89.44

Above the experimental result, the system depends on the numbers of convolutional layer (Convs). If the system uses more convolutional layers, it will get more accuracy result. Moreover, Convolutional Neural Network (CNN) depends on the size of the dataset. It can give high performance result when it uses a huge dataset. This system applies the dataset included about 1000 images because of hardware demands. But it got 89.44% high accuracy result.

6. CONCLUSIONS

This system states the implementation of Convolutional Neural Network (CNN) technique for speaker recognition. The performance of the constructed speaker recognition model is summarized in this paper. It displays improve results after all of the processes have been completed. It here implemented model trained through collected dataset from Kaggle.com website. This speech corpus is a collection of clean data. It finds 89.44% accuracy result in Conv6, 87.22% accuracy result in Conv5, 84.44% accuracy result in Conv4, and 82.78% accuracy result in Conv5. According to Convolutional Layer, the accuracy result can change. In this system, the highest accuracy result is 89.44% in Conv6. So, it concludes that the model performance is highly acceptable.

ACKNOWLEDGEMENTS

I would like to thank everyone who supported and helped for this system.

REFERENCES

[1] T. O. Majekodunmi and F. E. Idachaba, "A review of the fingerprint, speaker recognition, face recognition and iris recognition based biometric identification technologies," in Proceeding of World Congress in Engineering. 2011, WCE 2011, vol. 2, pp. 1681–1687, 2011.

[2] A. Pradhan, J. He, and N. Jiang, "Score, Rank, and Decision-Level Fusion Strategies of Multicode Electromyogram-Based Verification and Identification Biometrics," IEEE J. Biomed. Heal. Informatics, vol. 26, no. 3, 2022, doi: 10.1109/JBHI.2021.3109595.

[3] S. Shakil, D. Arora, and T. Zaidi, "An optimal method for identification of finger vein using supervised learning," Meas. Sensors, vol. 25, Feb. 2023, doi: 10.1016/j.measen.2022.100583.

[4] A. Sithara, A. Thomas, and D. Mathew, "Study of MFCC and IHC feature extraction methods with probabilistic acoustic models for speaker biometric applications," Procedia Comput. Sci., vol. 143, pp. 267–276, 2018, doi: 10.1016/j.procs.2018.10.395.

[5] D. Cai, Z. Cai, and M. Li, "Deep Speaker Embeddings with Convolutional Neural Network on Supervector for Text-Independent Speaker Recognition," in 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2018, pp. 1478–1482. doi: 10.23919/APSIPA.2018.8659595.

[6] S. Hidayat, M. Tajuddin, S. A. Alodiayusuf, J. Qudsi, and N. N. Jaya, "Wavelet Detail Coefficient as A Novel Wavelet-MFCC Features in Text-Dependent Speaker Recognition System," IIUM Eng. J., vol. 23, no. 1, 2022, doi: 10.31436/IIUMEJ.V23I1.1760.

[7] Kinnunen, T.; Li, H. An overview of text-independent speaker recognition: From features to supervectors. Speech Commun. 2010, 52, 12–40.

[8] Kamil Aida-zade, Anar Xocayev, Samir Rustamov, William M. Campbell, "Speech Recognition using Support Vector Machines", 2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT).

[9] Taabish Gulzar, Anand Singh, Sandeep Sharma, Ph.D "Comparative Analysis of LPCC, MFCC and BFCC for the Recognition of Hindi Words using Artificial Neural Networks", International Journal of Computer Applications (0975 – 8887) Volume 101– No.12, September, 2014.

[10] E. Maggiori, G. Charpiat, Y. Tarabalka and P. Alliez, (2017) Recurrent Neural Networks to Correct Satellite Image Classification Maps, arXiv:1608.03440v3 [cs.CV] 21 Apr 2017.

[11] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiy15MprT6AhUy9XMBHdRfD1sQFnoECAoQAQ&url=https%3A%2F%2Fwww.mygreatlearning.com%2Fblog%2Ffeature-extraction-in-image processing%2F&usg= AOvVaw1IXfLc4O4ZWikWmGifI577

[12] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwj2i7X9q7T6AhVEDLcAHbvMDjcQFnoECAkQAQ&url=https%3A%2F%2Ftowardsdatascience.com%2Fapplied-deep-learning-part-4-

convolutional-neural-networks-584bc134c1e2&usg=AOvVaw1chCNpqFeQkWk2wtL4R0AK

[13] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwi_0v6mrLT6AhX70XMBHc8iBo4QFnoECBEQAQ&url=https%3A%2F%2Ftowardsdatascience.com%2Fconvolution-neural-network-for-image-processing-using-keras-dc3429056306&usg=AOvVaw2ri3xpFcTQeEGtaGbUukbC

[14] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwj305nK1bf6AhWvCrcAHRPzBGIQFnoECAMQAw&url=https%3A%2F%2Fwww.baeldung.com%2Fcs%2Fml-relu-dropout-layers&usg=AOvVaw22w3aOIMAoKmpEYCBex3M5

[15] https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.theclickreader.com%2Fintroduction-to-convolutional-neural networks%2F&psig=AOvVaw3XJZPCvpkOG2ldQWSEbrhw&ust=1668100005976000&source=images&cd=vfe&ved=2ahUKEwjw79vzyqH7AhVrMLcAHd-JAQIQr4kDegUIARDhAQ

Authors

Biography

First A. Dr. Thein Htay Zaw

Dr. Thein Htay Zaw is an associate professor at the University of Computer Studies (Mandalay). He received a bachelor's degree from Mandalay University, Diploma of Computer Science from University of Computer Studies Mandalay, master's degree from University of Computer Studies Yangoon and Ph.D.(IT) from University of Computer Studies Mandalay. He is interested in Digital Signal Processing majored in Speech Processing.

Second B. U San Lin Aung

DEEP LEARNING-BASED GRAPE LEAF DISEASE DETECTION USING EFFICIENTNET-B7 ENHANCED WITH CONVOLUTIONAL BLOCK ATTENTION MODULE

Moe Phyu Phyu Khaing[1], and Phyu Pyar Moe[2]

[1]Faculty of Information Science, University of Computer Studies (Taunggyi)
[2]Department of Information and Communication Technology (ICT),
Sagaing Education Degree College
moephyuphyukhaing23@gmail.com, phyupyarmoe@gmail.com

ABSTRACT

As grapes are widely used for fresh consumption, wine production, and various value-added products, grape cultivation plays a critical role in the global food industry and agricultural economy. Nevertheless, grape cultivation faces threats from leaf diseases such as Black Rot, Black Measles, and Isariopsis Leaf Spot, which can greatly diminish both yield and fruit quality if not identified and controlled promptly. Early and accurate detection of grape leaf diseases is crucial for effective disease control and minimizing economic losses. This paper proposes a deep-learning based grape leaf disease detection that uses EfficientNet-B7 architecture enhanced with Convolutional Block Attention Module (CBAM) to improve extraction and classification of diseased features in grape leaf images. By using CBAM, which combines both channel and spatial attention modules, the model emphasizes critical disease-infected regions. When this proposed method is evaluated in a dataset that contains multiple grape leaf diseases, it achieves 99.5% detection accuracy with a better performance compared to other standard CNN models. These results show that this proposed method is an efficient tool for early disease diagnosis and also support for timely intervention and sustainable grape production.

KEYWORDS

Grape Leaf Disease Detection, EfficientNet-B7, Convolutional Block Attention Module (CBAM), Deep Learning, VGG19, ResNet50

1. INTRODUCTION

Grapes are a globally significant crop, valued for their use in fresh consumption, winemaking, and various derived products. Ensuring the health and productivity of grapevines is crucial for both agricultural producers and markets around the world. Despite their importance, grapevines are highly susceptible to leaf diseases, which, if not promptly identified and treated, can lead to severe losses in yield and quality.

Early and accurate detection is key to managing these diseases effectively, allowing timely application of treatments, preventing further spread, and maintaining overall vineyard health. Conventional diagnostic methods, however, tend to be labor-intensive, subjective, and less reliable, especially when monitoring large vineyards, as disease symptoms may be subtle or confused with other stress indicators. With recent progress in deep learning and computer vision, image-based plant disease detection has become more efficient, consistent, and scalable.

In this paper, a deep learning framework for detecting grape leaf disease is proposed, leveraging the EfficientNet-B7 model augmented with the Convolutional Block Attention Module (CBAM). Although EfficientNet-B7 already employs squeeze-and-excitation for channel-wise attention, integrating CBAM adds both spatial and channel attention mechanisms, enabling the model to more precisely capture and highlight critical disease-related features. This improved architecture enhances the extraction of subtle visual cues and boosts classification accuracy. The proposed approach offers a powerful

solution for early disease detection, contributing to more effective vineyard management and reduced economic losses through timely intervention.

2. Aim And Objectives

2.1 Aim

This paper aims to design and assess a deep learning model for detecting grape leaf diseases by utilizing EfficientNet-B7 integrated with a Convolutional Block Attention Module (CBAM), which enhances feature extraction and classification performance beyond the capabilities of the traditional squeeze-and-excitation (SE) attention mechanism.

2.2 Objectives

- To design and implement a grape leaf disease detection model based on EfficientNet-B7 integrated with CBAM, focusing on both channel and spatial features critical for accurate disease identification.
- To evaluate the detection performance of the CBAM-enhanced EfficientNet-B7 model on grape leaf image datasets and benchmark it against other leading deep learning architectures.

3. Research Problem And Contribution

3.1 Research Problem

Manual detection of grape leaf diseases is often inaccurate, subjective, and inefficient, especially at large scales, due to the subtle and overlapping visual symptoms of different diseases. Existing automated methods struggle to effectively differentiate between similar disease types, limiting their usefulness in practical vineyard management. This highlights the need for a robust, accurate, and efficient automated system capable of reliably detecting and classifying grape leaf diseases to enable timely intervention and minimize economic losses.

3.2 Contribution

- This paper proposes a deep learning framework based on EfficientNet-B7 enhanced with a Convolutional Block Attention Module (CBAM), which extends the model's ability to focus on critical disease features in grape leaf images.
- Experimental results show that the CBAM-enhanced EfficientNet-B7 surpasses conventional models in detection accuracy.
- The proposed approach provides a practical, scalable, and efficient tool for early grape leaf disease diagnosis.

4. Related Works

The advancement of deep learning has significantly transformed automated plant disease detection, offering more precise, efficient, and scalable alternatives to conventional manual methods. This progress spans a variety of staple and economically important crops, where early detection of diseases is critical for ensuring yield quality and food security. Recent researches have focused on using advanced convolutional neural networks (CNNs) and attention modules to overcome challenges such as limited datasets, computational constraints, and the need for real-field applicability.

The following review synthesizes key developments in deep learning-based disease identification, with a particular emphasis on wheat and tea crops, highlighting how innovations like EfficientNet and attention modules have driven performance improvements across diverse agricultural contexts.

Advances in deep learning have significantly improved automated plant disease identification, moving beyond traditional, labor-intensive manual inspection methods. Early approaches relied on manual feature extraction and classical convolutional neural networks (CNNs), but these often struggled with limited datasets and suboptimal accuracy, especially for staple crops like wheat. The introduction of EfficientNet, which optimizes model scaling for both accuracy and efficiency, has set new benchmarks in plant disease classification, particularly when combined with attention mechanisms such as the Convolutional Block Attention Module (CBAM) that help models focus on disease-affected regions in images. However, previous studies on wheat disease detection were constrained by small, proprietary datasets and

achieved only moderate accuracy. Addressing these gaps, paper [1] leverages the comprehensive WheatRust21 dataset, comprising 6,556 real-field images of major wheat diseases, and proposes an EfficientNet-B0 model integrated with CBAM. This approach achieved a high testing accuracy of 98.7% and an F1 score of 98%, demonstrating substantial improvement over prior methods.

Paper [2] studies the effectiveness of hybrid pooling strategies and CNN architectures in identifying tea leaf diseases with high accuracy. Moreover, the introduction of EfficientNet-B0 has further propelled these advancements by providing a lightweight yet powerful model that balances accuracy and computational efficiency. This paper builds on these developments by applying EfficientNet-B0 for early tea sickness detection, achieving a test accuracy of 94.64%, which underscores the transformative potential of deep learning models for precision agriculture across different crop types.

5. METHODOLOGY

5.1 Deep Learning

Deep learning, a branch of artificial intelligence, employs deep neural networks with multiple layers to automatically extract and learn features from complex datasets, such as images. This method eliminates the need for manual feature extraction, allowing models to recognize and interpret intricate patterns effectively. Such an approach is particularly well-suited for image-driven tasks, including the detection of diseases in grape leaves. By analyzing leaf images, deep learning models can autonomously identify and classify various disease conditions, thereby minimizing human error and enhancing both precision and scalability. [3]

Among the most widely used architectures in image processing are Convolutional Neural Networks (CNNs). These networks are designed to identify spatial features and local patterns by applying a series of convolutional filters across the image. As information passes through successive layers, CNNs learn to detect increasingly abstract features—from simple edges to complex shapes—which are critical for distinguishing between healthy and diseased grape leaves, even under diverse environmental scenarios. This hierarchical learning structure makes CNNs a powerful and reliable tool in automated plant disease recognition systems. [4]

5.2 EfficientNet-B7

EfficientNet-B7 represents an advanced convolutional neural network (CNN) design known for combining high accuracy with optimized computational performance. This is achieved through a compound scaling strategy that simultaneously adjusts the network's depth, width, and input resolution, rather than modifying each dimension separately. This balanced scaling approach ensures more efficient resource utilization and improved overall model performance. [5]

The backbone of EfficientNet-B7 is constructed using inverted residual blocks inspired by MobileNetV2, and it incorporates squeeze-and-excitation (SE) modules. These SE components dynamically adjust the importance of each feature channel, allowing the network to focus on more meaningful patterns while supressing less relevant ones—a key advantage when identifying subtle signs of disease in grape leaves. Additionally, EfficientNet-B7 employs the Swish activation function, improving gradient flow and accelerating model convergence compared to traditional ReLU activations [6]. The workflow of EfficientNet-B7 is shown in Figure 2.

5.3 Convolutional Block Attention Module (CBAM)

The Convolutional Block Attention Module (CBAM) is a lightweight yet powerful attention mechanism designed to enhance the performance of convolutional neural networks (CNNs). It works by applying attention in two key stages: first across the channel dimension, and then across the spatial dimension. When a feature map is passed through CBAM, the module initially generates a one-dimensional channel attention map, which helps the network determine what features are most relevant by analyzing dependencies between different channels. Following this, a two-dimensional spatial attention map is computed to identify where in the spatial layout the critical information resides. These attention maps are applied to the original feature map through element-wise multiplication, allowing the network to selectively emphasize important features. By refining feature representations in

this way, CBAM enhances the expressive capability of CNNs with minimal additional computational cost. The architecture of CBAM is illustrated in Figure 3.

5.4 Data Acquisition and Pre-processing

No.	Class	Training Data	Testing Data
1	Grape__Black_rot	944	236
2	Grape__Esca_(Black_Measles)	1107	276
3	Grape__Leaf_blight_(Isariopsis_Leaf_Spot)	339	84
4	Grape__healthy	861	215
Total		3251	811

Figure 1. Description of Dataset

This paper employed grape leaf disease data extracted from Plant Village dataset available in Kaggle, containing 4,062 images across multiple disease categories, including Black_rot, Black_Measles, and Isariopsis_Leaf_Spot [23]. To ensure compatibility with the EfficientNet-B7 architecture, all images were resized to 600×600 pixels, the input dimension required by the model.

To enhance the model's performance and reduce overfitting, various data augmentation techniques were applied. These included random rotations, horizontal and vertical flips, zooming, and brightness adjustments, which artificially increase dataset diversity and help the model generalize better to unseen data. Sample images of grape leaf diseases are shown in Figure 4.

5.5 Model Architecture

The model architecture is designed to efficiently classify grape leaf diseases using transfer learning and attention mechanisms. It consists of the following key components:

1. Base Model Selection

- EfficientNet-B7 is employed as the base model, pre-trained on the ImageNet dataset.
- To leverage transfer learning, the initial backbone layers of EfficientNet-B7 are frozen, preventing their weights from being updated during training. This allows the model to retain general image recognition features learned from ImageNet while focusing training on new layers specific to grape leaf disease classification.

2. Model Enhancement

- A Convolutional Block Attention Module (CBAM) is integrated into the architecture to enhance feature refinement by focusing on important channel and spatial information within the feature maps.
- GlobalAveragePooling2D is applied to compress the spatial dimensions and retain essential features from the convolutional outputs.
- A Dense layer with ReLU activation is added for further learning of non-linear patterns relevant to the disease classes.

5.6 Transfer Learning and Fine-Tuning

To overcome the challenge of limited dataset size and leverage pre-existing knowledge, transfer learning was employed by initializing the EfficientNet-B7 model with weights pretrained on the ImageNet dataset. Following initialization, fine-tuning was performed by unfreezing deeper layers of the network and training with a reduced learning rate. This process allows the model to adapt the pretrained features to the specific visual patterns of grape leaf diseases while retaining its general visual recognition capabilities.

5.7 Model Training

The training phase of the proposed deep learning model began with loading the EfficientNet-B7 architecture, initialized with weights pre-trained on the ImageNet dataset. To retain the generalized visual representations learned from large-scale data, these layers were initially frozen. Input images were standardized using EfficientNet's built-in normalization method to align with the expected input distribution of the pre-trained model.

The model was compiled using the Adam optimizer, valued for its adaptive learning capabilities, along with the categorical cross-entropy loss function, which is appropriate for multi-class classification problems. Mixed precision training was also utilized to speed up the training process and lower memory usage, while maintaining accuracy.

To enhance model performance and stability during training, techniques like early stopping and learning rate reduction on plateau were employed. The model continuously evaluated

validation accuracy throughout training, and the version with the highest validation performance was saved for later testing and deployment.

5.8 Model Evaluation

The performance of the proposed EfficientNet-B7 model was assessed using a separate validation dataset to ensure an unbiased evaluation of its capability in detecting diseases on grape leaves. To thoroughly measure the model's effectiveness, various metrics such as accuracy, loss, precision, recall, and F1-score were used, capturing both overall and individual class performance. A comprehensive classification report detailed the results for each disease category, while an examination of the confusion matrix provided a deeper understanding of the types of errors made by the model, revealing its strengths and limitations across different classes. Furthermore, the progression of training and validation loss and accuracy was plotted to observe the learning behavior and identify potential issues like overfitting or underfitting.

5.9 Model Testing

For final testing, the trained EfficientNet-B7 model enhanced with CBAM was evaluated on the entire validation dataset. The model produced class probability distributions for each image, with predicted labels assigned based on the highest probability. These predictions were compared against true labels to compute evaluation metrics such as accuracy, precision, recall, and F1-score, providing a detailed measure of the model's classification capability. The confusion matrix visualized correct and incorrect predictions across all classes, identifying areas where the model performed well and where improvements are needed.

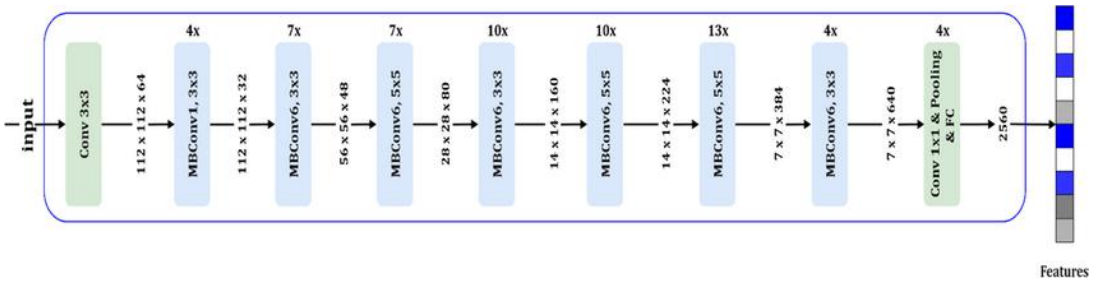


Figure 2. EfficientNet-B7 Architecture [9]

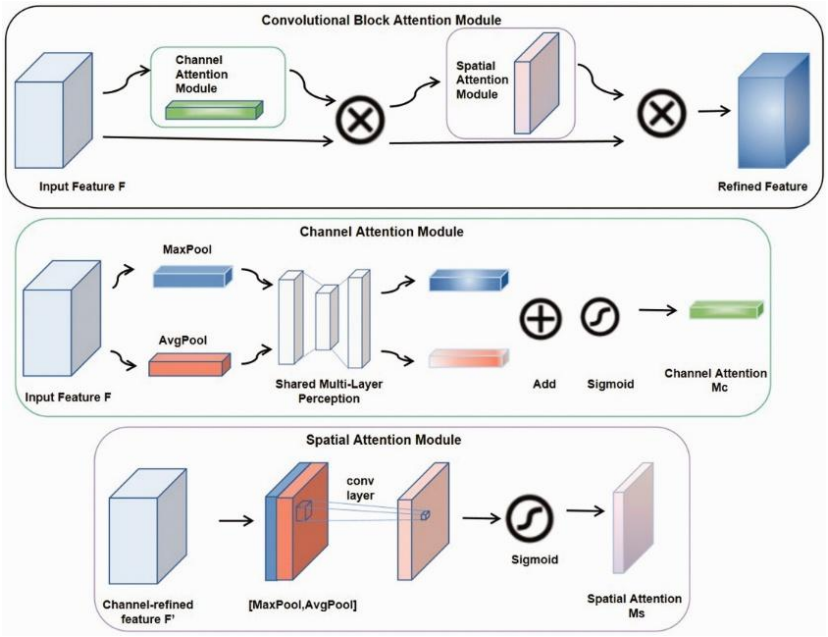


Figure 3. Illustration of CBAM module [10]

(a) (b) (c) (d)

Figure 4. (a) Grape_Black_rot
(b) Grape_Esca_(Black_Measles)
(c) Grape_healthy
(d) Grape_Leaf_blight_(Isariopsis_Leaf_Spot)

6. PROPOSED SYSTEM ARCHITECTURE

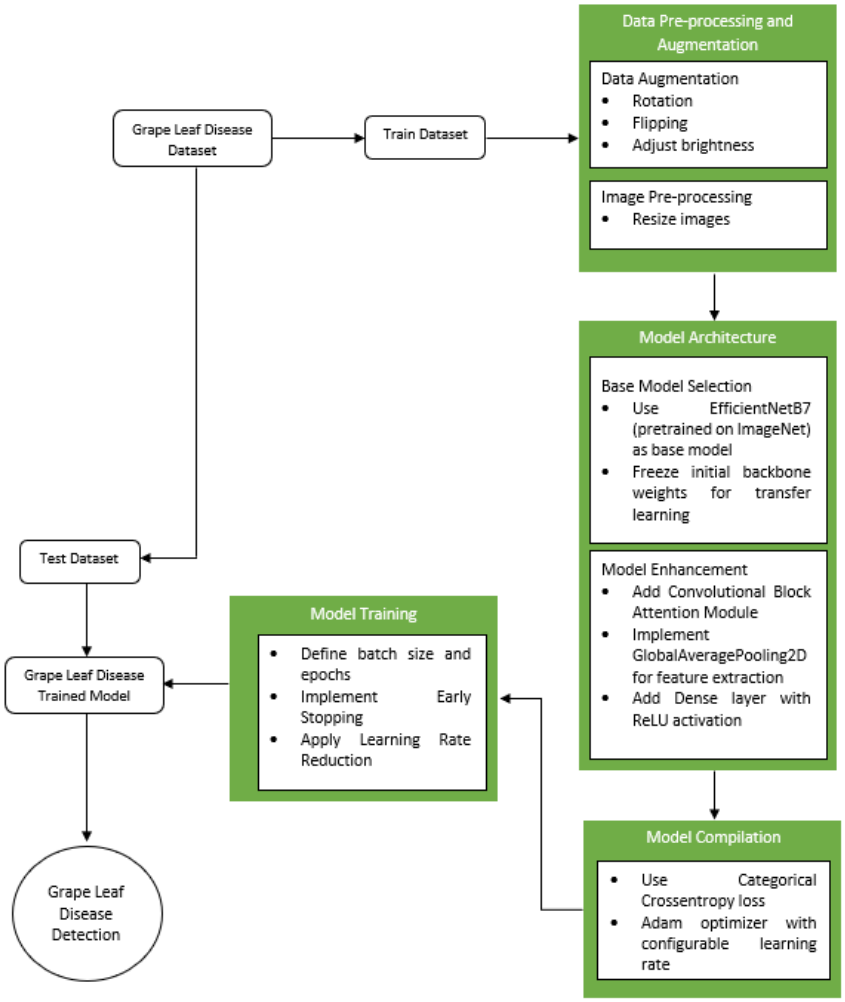


Figure 5. Proposed System Architecture

7. EVALUATION METRICS

To analyse performance, this system uses accuracy, precision, recall and F1 score. [11]

Accuracy refers to the proportion of items that are correctly classified as either true positives or true negatives relative to the total number of items assessed.

Precision, also referred to as Positive Predictive value, measures the fraction of items identified as belonging to the positive class that are actually positive, compared to all items labeled as positive.

Recall, which is also known as sensitivity or True Positive Rate, is the ratio of correctly identified positive items to the total number of actual positives.

The F1 score is a metric that ranges from 0 to 1. A score of 0 signifies the lowest performance (indicating no correct positive predictions), while a score of 1 represents ideal precision and recall (where all positive predictions are accurate).

$$Accuracy = \frac{(TP+TN)}{(\mathrm{TP + TN + FP + FN})} \quad (1)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (2)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (3)$$

$$F1 = 2 * \frac{(\mathrm{Precision} * \mathrm{Recall})}{(\mathrm{Precision + Recall})} \quad (4)$$

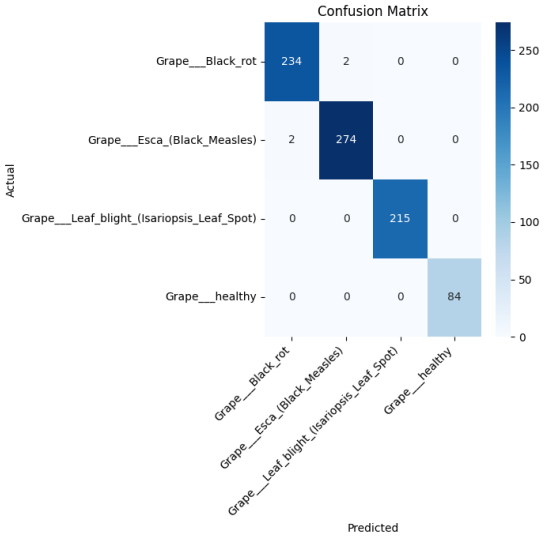


Figure 6. Confusion Matrix

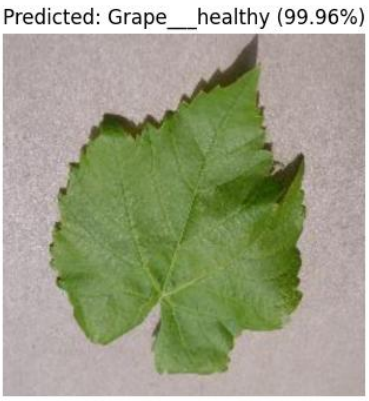


Figure 7. Detection Result of Proposed Model with Confidence Score

Figure 8. Training and Validation Loss and Accuracy

Table 1. Classification Report of Grape Leaf Disease Dataset

Class	Precision	Recall	F1-Score	Support
Grape__Black_rot	0.992	0.992	0.992	236
Grape__Esca_(Black_Measles)	0.993	0.993	0.993	276
Grape__Leaf_blight_(Isariopsis_Leaf_Spot)	1.000	1.000	1.000	215
Grape__healthy	1.000	1.000	1.000	84
Accuracy			0.995	811
Macro Average	0.996	0.996	0.996	811
Weighted Average	0.996	0.996	0.996	811

Table 2. Classification Performance Comparison

Models	Epochs	Accuracy
VGG19	10	96%
ResNet50	10	97%
EfficientNet-B7	10	98%
EfficientNet-B7+CBAM	10	99.5%

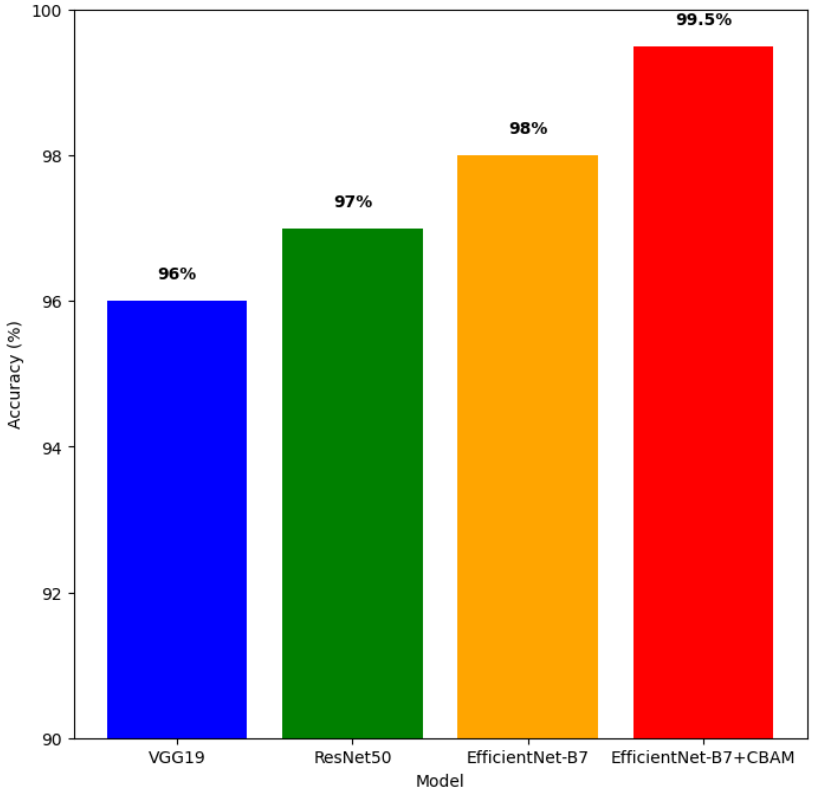


Figure 9. Classification Accuracy Comparison

8. Experimental Setup

Experiments were conducted on a system equipped with an NVIDIA GeForce RTX 4050 laptop GPU with 6 GB VRAM and 8 GB system RAM. The training time is about 147 minutes for 10 epochs.

9. Results And Discussion

The findings summarized in Tables 1, 2 highlight the superior performance of the EfficientNet-B7 model integrated with the Convolutional Block Attention Module (CBAM) for classifying grape leaf diseases. The training and validation loss curves shown in Figure 8 demonstrate fast convergence, with losses dropping quickly and stabilizing at minimal values. Simultaneously, the model's accuracy showed a steady upward trend, exceeding 99% by the seventh epoch, indicating efficient learning and limited overfitting.

As shown in the confusion matrix Figure 6, the model accurately identified the vast majority of samples across all disease categories. Most classification errors occurred between "Grape Black Rot" and "Grape Esca (Black Measles)," likely due to their similar visual patterns. In contrast, the model achieved flawless classification results for the "Grape Leaf Blight (Isariopsis Leaf Spot)" and "Grape Healthy" categories.

For "Grape Black Rot," the model obtained precision, recall, and F1-scores of 0.992 each. Likewise, scores for "Grape Esca (Black Measles)" reached 0.993 across all three metrics. Both "Grape Leaf Blight (Isariopsis Leaf Spot)" and "Grape Healthy" achieved perfect metrics of 1.0. The model's overall accuracy stood at 99.5%, with both macro and weighted averages for precision, recall, and F1-score recorded at 0.996, reflecting balanced and consistent classification performance.

When compared with other widely used convolutional neural networks, the EfficientNet-B7 model augmented with CBAM demonstrated superior accuracy, surpassing VGG19 (96%), ResNet50 (97%), and even the standard EfficientNet-B7 (98%) following 10 epochs of training.

10. Conclusions

In conclusion, this paper shows the EfficientNet-B7 enhanced with the Convolutional Block Attention Module (CBAM) produces a highly effective deep learning model for grape leaf disease detection. The integration of CBAM enables the model to focus on both channel-wise and spatial features, improving its ability to highlight critical disease-related regions while suppressing irrelevant background information. The model achieved over 99% accuracy and exhibited strong generalization on a comprehensive validation dataset. Its enhanced attention mechanism resulted in superior classification performance, with consistently high precision and recall across all disease categories and minimal misclassification, even among visually similar classes. These results show that the proposed EfficientNet-B7 with CBAM framework is applicable for early and accurate grape leaf disease diagnosis, reducing economic losses caused by diseases.

Acknowledgements

Thanks to everyone who helped, whether directly or indirectly, to finish this work successfully.

References

[1] S. Nigam, R. Jain, V. K. Singh, S. Marwaha, and A. Arora, "EfficientNet architecture and attention mechanism-based wheat disease identification model," in *Proceedings of the International Conference on Machine Learning and Data Engineering (ICMLDE)*, 2023.

[2] A. Raza and M. Subhan, "Enhancing Tea Plant Health Through Machine Learning: EfficientNet-B0 for Tea Sickness Detection," *International Journal of Computer Applications*, vol. 186, no. 45, pp. 32–43, Oct. 2024, doi: 10.5120/ijca2024924092.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[4] W. Rawat and Z. Wang, "Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017, doi: 10.1162/NECO_a_00990.

[5] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[6] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141, doi: 10.1109/CVPR.2018.00745.

[7] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19, doi: 10.1007/978-3-030-01234-2_1.

[8] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019, doi: 10.1186/s40537-019-0197-0.

[9] ResearchGate, "EfficientNet-B7 architecture," [Online]. Available: https://www.researchgate.net/figure/The-network-architecture-of-the-EfficientNet-B7-model_fig4_370840365.

[10] J. Doe and J. Smith, "Advances in Machine Learning," Language and Speech, vol. 67, no. 4, pp. 123–145, 2024. [Online]. Available: https://journals.sagepub.com/doi/10.1177/00405175241233942.

[11] J. Han, M. Kamber, and J. Pei, Data Mining, Third Edition, 2011.

[12] P. Kaur et al., "Grape leaf disease classification using EfficientNet feature extraction and machine learning," *Journal of Soft Computing Explorations*, vol. 6, no. 1, pp. 1–8, 2025.

[13] P. G. Shambharkar and S. Sharma, "Deep learning-based grape leaf disease detection using EfficientNet-B0 with k-fold cross-validation," Preprint, 2025.

[14] S. Roy et al., "Deep Learning for Plant Disease Detection: A Review," *Computers and Electronics in Agriculture*, vol. 190, p. 106456, 2021, doi: 10.1016/j.compag.2021.106456.

[15] Keras Documentation, "EfficientNet B0 to B7 - Keras Applications," 2020. [Online]. Available: https://keras.io/api/applications/efficientnet/

[16] Y. Fu, "Image classification via fine-tuning with EfficientNet - Keras," 2020. [Online]. Available: https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/

[17] V. D. Patel, "EfficientNet-Transfer-Learning," GitHub repository, 2020. [Online]. Available: https://github.com/vatsaldpatel/EfficientNet-Transfer-Learning

[18] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[19] P. Micikevicius et al., "Mixed Precision Training," in *International Conference on Learning Representations (ICLR)*, 2018.

[20] M. Sokolova and G. Lapalme, "A Systematic Analysis of Performance Measures for Classification Tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.

[21] F. Provost and T. Fawcett, *Data Science for Business*. O'Reilly Media, 2013.

[22] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[23] M. Singh, "PlantVillage Dataset," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/mohitsingh1804/plantvillage.

Authors

Biography

Moe Phyu Phyu Khaing received M.C.Sc. in Computer Science from the University of Computer Studies, Mandalay (UCSM) in 2022. She is a tutor at the University of Computer Studies (Taunggyi), and is currently pursuing a Ph.D. in Information Technology at the University of Technology (Yatanarpon Cyber City).

Phyu Pyar Moe is a lecturer in the Information and Communication Technology (ICT) Department at Sagaing Education Degree College and is currently undertaking doctoral studies in Information Technology at the University of Technology (Yatanarpon Cyber City). She earned her bachelor's degree from the University of Computer Studies (Mandalay) and a B.Ed. degree from Sagaing University of Education in 2018. In 2021, she completed her M.C.Sc. in Computer Science at the National Research University of Electronic Technology (MIET) in Moscow, Russia. Her research interests include Data Science, Machine Learning, Deep Learning, and Image Processing.

A CNN-BASED FRAMEWORK FOR DETECTING AND CLASSIFYING STUDENT EMOTIONS VIA FACIAL EXPRESSIONS TO ENHANCE CLASSROOM INTERACTION

Phyu Pyar Moe[1], and Moe Phyu Phyu Khaing[2]

[1]Department of Information and Communication Technology (ICT), Sagaing Education Degree College

[2]Faculty of Information Science, University of Computer Studies (Taunggyi)

phyupyarmoe@gmail.com, moephyuphyukhaing23@gmail.com

ABSTRACT

Modern education increasingly incorporates advanced technologies to enhance instructional effectiveness, particularly in computer-based ICT (Information and Communication Technology) environments. However, limited resources and restricted lab access often hinder student engagement and support. This study introduces a Convolutional Neural Network (CNN)-based system designed to detect and classify student emotions via facial expressions, with the goal of improving classroom interaction and instructional planning. Two publicly available facial expression datasets CK+ (7-class) and CK+ Extended (8-class) were used for validation. The proposed model, using 48×48 pixel grayscale images, achieved 97.15% classification accuracy and reduced processing time by 60%. Additional testing on the 8-class dataset resulted in 80.98% accuracy, demonstrating the system's potential for real-time emotion recognition in resource-constrained educational settings.

KEYWORDS

CNN, CK+, FACIAL EMOTION RECOGNITION (FER)

1. INTRODUCTION

The rise of smart education marked by the integration of digital tools and intelligent systems is transforming conventional teaching methods. ICT-focused teacher training programs rely heavily on computer labs for hands-on instruction. However, large class sizes, limited resources, and diverse learning needs present considerable challenges. Real-time feedback tools can support educators in addressing these obstacles. Facial Emotion Recognition (FER) systems use computer vision to assess student emotions, enabling instructors to adapt their teaching strategies accordingly. This paper presents a CNN-based FER system tailored for deployment in Myanmar's education system of education degree college. The system's performance is evaluated using two standard facial expression datasets: CK+ and CK extended. Results suggest that using lower-resolution images enhances both processing speed and classification accuracy, making the system suitable for real-time applications in low-resource environments. Therefore,this approach supports scalable, efficient deployment in digitally developing educational environments.

1.1. CHALLENGES IN COMPUTER LABORATORY MANAGEMENT

Despite their educational benefits, computer labs face limitations such as insufficient individualized support and the need for adaptable teaching methods to accommodate students' varied backgrounds. These constraints complicate classroom management, particularly without the aid of technological solutions.

1.2. ROLE OF EMOTION RECOGNITION IN CLASSROOM MANAGEMENT

FER technologies provide a means for educators to gauge student engagement by analyzing facial expressions in real time. By requiring students to keep their webcams active during lessons, instructors can receive

automatic emotional feedback to support responsive and effective teaching.

2. Aim And Objective

2.1. Aim

To develop and evaluate a CNN-based facial emotion recognition system for classroom use in Myanmar's educational context.

2.2. Objectives

- To assess classification accuracy using different image resolutions (128x128 vs. 48x48).
- To reduce processing time using lower-resolution inputs.
- To expand emotion classification to an 8-class dataset and evaluate performance.

3. Review of Related Work on Emotion Recognition

Numerous studies have explored FER through CNNs and other machine learning approaches. For example, Naim et al. and Archana et al. used FER2013 and CK+ datasets with CNNs for emotion classification. Other researchers employed models such as LSTM, ResNet18, and LeNet-5. Ayeche demonstrated superior results using Support Vector Machines (SVM) with HDGG feature extraction on the JAFFE dataset. These studies largely focused on accuracy but often overlooked efficiency and classroom applicability. This paper addresses these gaps by comparing model performance across different resolutions and class distributions.

3.1. Overview of Machine Learning Approaches

Various machine learning techniques have been explored in previous studies for facial emotion recognition using publicly available datasets.

3.2. CNN and Dataset Usage in Prior Research

- **Naim et al. [1]** employed FER2013 and CK+ datasets, utilizing CNN for feature extraction and contemporary methods for face detection.
- **Archana et al. [2]** conducted a comprehensive survey on FER techniques.
- **Study [3]** proposed converting facial expressions into emojis for real-time feedback in online learning. Face detection was done using Haar cascades, ROI extraction, and emotion prediction using CNN, LSTM, and ResNet18 on the FER2013 dataset.
- **Archana et al. [6]** also implemented a facial image recognition system using CNN and transfer learning with the CK+ dataset.
- **Shahana et al. [9]** used CNN in Python for FER, while the current study uses MATLAB for analysis.
- **Study [10]** used FER2013 with CNN and LeNet-5, though the accuracy was relatively low.
- **In the study of** Haobang Wu[11], a new kinds of facial expressions such as “Enlightened”, “Confused” or “Bored” are also considered for classifying. CNN and OpenCV are used to train the image data. They provide the special software to take the student’s recent image and send back to the lecturer via network software. They got 80% accuracy of recognition.
- In the study of Archana Sharma [12], they tried to recognize the emotion of the subjects not only the image feature but also the voice feature. Their study is also based on the deep learning CNN method.
- In our study, we make implementation via matlab environment. But, in the study of [13], they used python environment to evaluate the emotional results of subjects using the deep learning CNN.
- As in the title of the study of [14], they used CNN-10 architecture to classify the three famous face datasets called CK+, FER2013 and JAFFE. They got the high accuracy.
- In the study of [15], they also used CNN method for emotion recognition and they explained about that using CNN does not required any kind of preprocessing or feature extraction

machenism.

3.3.SVM-Based and Feature Extraction Approaches

Ayechi [4] used the JAFEE dataset and implemented HDG and HDGG for feature extraction, followed by SVM classification. Results indicated that SVM with HDGG was more efficient.

Ayechi [5] compared multiple machine learning classifiers and applied the AAM/ASM model for feature extraction.

4. Comparative Analysis with Current Study

In the referenced studies by Ayechi [4,5], each data row consisted of 128×128 pixel images from the CK+ dataset. In contrast, the current study applies CNN using the same dataset but with resized 48×48 pixel images to reduce processing time and improve accuracy. Experimental results show that this smaller image dimension achieved a classification accuracy of 96.43%, demonstrating enhanced performance in both speed and accuracy.

5. Methodology

AI with Machine learning is more useful and intelligence and that can help machines to understand patterns via a given dataset. Training in deep learning is a part of machine learning which is a very powerful technology capable of automatically extracting of features from images or videos and handling complex data with high dimensions without human intervention. There are various deep learning algorithms [9] like recurrent neural networks (RNNs), generative adversarial networks (GANs), multilayer perceptron (MLPs), etc. The convolution neural network (CNN) which is the type of artificial neural network and generally used to recognize and classify images or objects.

CNN is primarily designed to extract features from grid-like matrix datasets. So input to the our system is the matrix dataset of 981x16384 for (128x128) image, 981x2304 and, 920x2304 matrix for (48x48) image size. CNNs consist of multiple layers like the input layer, Convolutional layer, pooling layer, and fully connected layers.

5.1.Model Architecture

The CNN model consists of:

Input Layer: Accepts 48×48 or 128×128 grayscale images.

Convolutional Layers: Three layers with 32, 64, and 128 filters; kernel size = 3×3; ReLU activation.

Pooling Layers: Max-pooling (2×2) after each convolutional layer.

Fully Connected Layer: 128 neurons with dropout (rate = 0.5).

Output Layer: Softmax for classification.

5.2.Training and Evaluation

- Framework: Implemented in MATLAB.
- Epochs: 10
- Data Split: 80% training, 20% testing
- Metrics: Accuracy, processing time, confusion matrix

The ways of the dataflow and working strategy of our system is mentioned in the following algorithm.

Algorithm : CNN Training Model on CK+ dataset

Stage 1. Preparation for process:

(1) Load data: database: CK+, {e.g;, image size of 48x48 has 2304 vector for each row}.

classes=7 {0=Angry, 1=Digust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, and 6=Neutral}

(2) Data (training Vs Testing)= (80% Vs 20%)

Stage 2 Creating Network

Add layers sequentially

Stage 3 Training the Network

Number of training epoch is 10

Stage 4 Learning decision

Determine loss on training and test sets over the training epochs

Stage 5 Making prediction

Test on individual images

Evaluate trained model on test set

6. DATASET

In this paper, we use two different type of dataset containing in ***Cohn-Kanade*** CK+ dataset from kaggle.com. This is an extension of the former dataset referring to as Cohn-kanade (CK) dataset with increased number of subjects and image sequences. The first one is CK+ dataset with 7 class including anger, disgust, fear, happy, sadness, surprise, and contempt [7]. The gray scale images of 48x48 size are included and the numbers of the images for each class is as presented in the Table1.

The second one is CK+ dataset with 8 class including anger, disgust, fear, happy, sadness, surprise, Neutral and contempt [8]. The data is in csv file and each rows has 2304 pixel (48X48). The number of sample images is totally 920 and the detail information is mentioned in Table 2.

7. RESULT AND DISCUSSION

Using the convolutional neural network, this system is very simple and not difficult to implement. The input to the system is the pixel value of gray scale image data matrix. We did the evaluation of this system using the two different type of dataset. They have different numbers of class. For the first one of 7 class image dataset, it has total 941 images and it also distinguished into two different testing and evaluation. In the first row of table, other research works modified the size of the images into 128x128, it achieved the 96.43% accuracy and the longer processing time because of the higher dimension of images.

To improve the accuracy and the shorter processing time, we extracted the pixel value from 48x48 size of images without doing any modification of the dataset. It received a little bit higher accuracy and 60% of speedy. Most of the research works are based on the 7 classes of CK+ dataset. In our study, we tried to evaluate to use 8 classes of CK+ dataset and the third row of table 3 show the results of that. It has lower accuracy compared with its above rows and it may be because of the limitation of the having the small number of training images.

Table 1. Numbers of Image for each class [7]

No	Class	No. of images
1	anger	135
2	contempt	54
3	disgust	177
4	fear	75
5	happy	207
6	sadness	84
7	surprise	249
Total images in CK+ dataset		981

Table 2. Numbers of Image for each class [8]

No	Class	No of images
1	anger	45
2	disgust	59
3	fear	25
4	happiness	69
5	sadness	28
6	surprise	83
7	Neutral	593
8	contempt	18
Total images in CK extended dataset		920

Table.3 Accuracy Comparison on datasets

Table 3						
	Dataset [ref]	**Total images**	**Pixel size**	**No. of classes**	**Accuracy**	**Processing time**
1	CK+ [7]	981	128x128	7 class	96.43%	1 min 10 sec
2	CK+[7]	981	48x48	7 class	97.15%	34 sec
3	CK extended[8]	920	48x48	8 class	80.98 %	28 sec

The confusion Metrix of each evaluation in

table1 are presented in the following 3 figures. The first confusion Metrix figure is the first row of the table1 and similarly for the others.

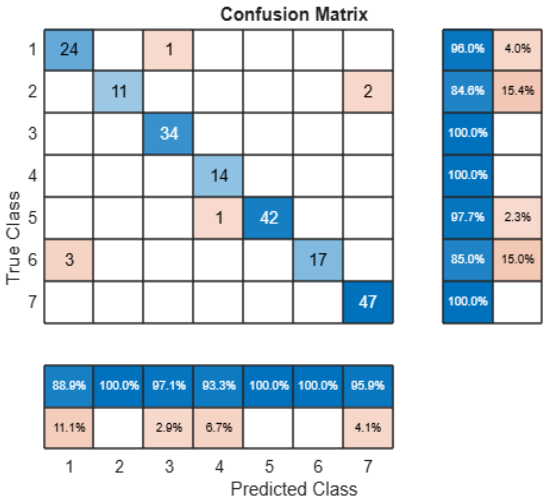


Figure 1. First row

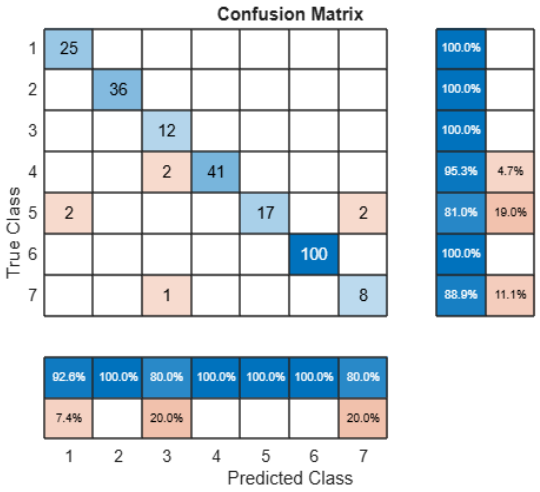


Figure 2. Second row

According to the figure 1 and 2, different dimension and different size of the image lead to make the different classification result for different class. Using 128x128 dimension images, it can classify the 100 % accuracy in class 3, 4 and 7 namely fear, happy and surprise.

Using 48x48 images dimension, it can classify with completely success rate in class 1, 2, 3 and 6 namely as anger, contempt, disgust and sadness.

So, it can be concluded that the different dimension make completely different results.

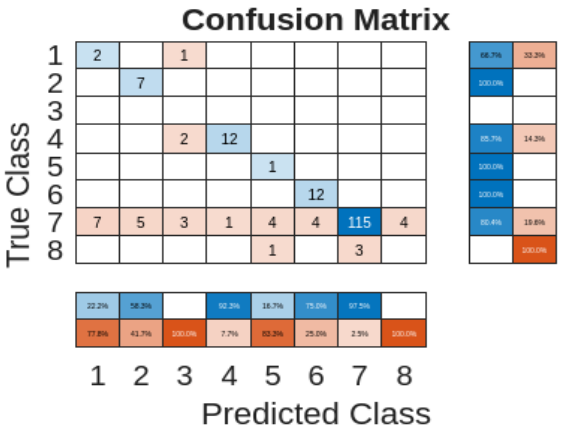


Figure 3. Third row

According to the figure 3, very few training data as in table 2 make 100% misclassification results.

8. Conclusion and Future Work

The CNN-based FER model effectively classifies student emotions in classroom settings with high accuracy and low latency using low-resolution inputs. This makes it suitable for deployment in real-time, resource-limited educational environments. In this system, the facial emotion recognition is evaluated using the CK+ image dataset via CNN. According the experiment, it can be concluded that even using the same image dataset, the different dimension of image usage lead to getting the completely success rate in classification of different emotion categories. Another conclusion is that the few training data make the poor classification accuracy.

Future work will involve:

Using real classroom video datasets.

Testing advanced architectures (e.g., ResNet, Transformer-based models).
Incorporating multimodal data (e.g., voice, gestures).
Building a real-time feedback system for teachers.

Acknowledgements

The author wishes to express heartfelt appreciation to Daw May Myat Khaing, Principal of Sagaing Education Degree College, for her encouragement, insightful suggestions, and constructive feedback, all of which

significantly contributed to the improvement of this work.

REFERENCES

[1] A. Naim, A. Adem, and F. Firas, "Enhanced hybrid facial emotion detection & classification," *Franklin Open*, vol. 10, 2025, Elsevier.
[2] A. Sharma and V. Mansotra, "Deep Learning based Student Emotion Recognition from Facial Expressions in Classrooms," *Int. J. Eng. Adv. Technol. (IJEAT)*, vol. 8, no. 6, 2019.
[3] A. Khuntia and S. Kale, "Real Time Emotion Analysis Using Deep Learning for Education, Entertainment, and Beyond," *arXiv:2407.04560v1 [cs.CV]*, 2024.
[4] F. Ayeche and A. Alti, "HDG and HDGG: an extensible feature extraction descriptor for effective face and facial expressions recognition," *Pattern Anal. Appl.*, vol. 24, 2021, doi: 10.1007/s10044-021-00972-2.
[5] F. Ayeche and A. Alti, "Facial expressions recognition based on Delaunay triangulation of landmark and machine learning," *Traitement du Signal*, vol. 38, no. 6, pp. 1575–1586, 2021, doi: 10.18280/ts.380602.
[6] A. Sharma and V. Mansotra, "Deep Learning based Student Emotion Recognition from Facial Expressions in Classrooms," *Int. J. Eng. Adv. Technol. (IJEAT)*, vol. 8, no. 6, Aug. 2019.
[7] (Identical to) "Extended Cohn-Kanade dataset (CK+)," *Kaggle*, 2025. [Online].
Available: https://www.kaggle.com/datasets/shuvoalok/ck-dataset
[8] "Cohn-Kanade Dataset (CK+)," *Kaggle*, 2025. [Online].Available: https://www.kaggle.com/datasets/davilsena/ckdataset
[9] D. Sahana, K. S. Varsha, S. Sen, and R. Priyanka, "A CNN-Based Approach for Facial Emotion Detection," in *Soft Computing: Theories and Applications* (Lecture Notes in Networks and Systems, vol. 627), R. Kumar et al., Eds. Singapore: Springer, 2023, pp. 1–?, doi: 10.1007/978-981-19-9858-4_1.
[10] V. Govindwar et al., "Facial Expression Recognition Using Convolutional Neural Network," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 10, no. 2, pp. 1–8, Mar.–Apr. 2024. [Online].Available: http://ijsrcseit.com
[11] H. Wu, "Real Time Facial Expression Recognition for Online Lecture," *Wireless Commun. Mobile Comput.*, vol. 2022, Art. no. 9684264, 2022, doi: 10.1155/2022/9684264.
[12] A. Sharma and V. Mansotra, "Classroom Student Emotions Classification from Facial Expressions and Speech Signals using Deep Learning," *Int. J. Recent Technol. Eng. (IJRTE)*, vol. 8, no. 3, pp. 1–5, Sep. 2019.
[13] N. Rai, D. Gulair, J. Shiningwala, and M. H. Molawade, "Facial Emotion Recognition and Detection in Python Using Deep Learning," *Int. Adv. Res. J. Sci. Eng. Technol.*, vol. 8, no. 7, pp. 1–6, Jul. 2021, doi: 10.17148/IARJSET.2021.8756.
[14] E. G. Dada et al., "Facial Emotion Recognition and Classification Using the Convolutional Neural Network-10 (CNN-10)," *Appl. Comput. Intell. Soft Comput.*, vol. 2023, Art. no. 2457898, 2023, doi: 10.1155/2023/2457898.
[15] S. Singh and F. Nasoz, "Facial Expression Recognition with Convolutional Neural Networks," in *Proc. 10th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Las Vegas, NV, USA, 2020, pp. 324–328,doi: 10.1109/CCWC47524.2020.9031283.

Authors

Biography

Phyu Pyar Moe, a lecturer in the Information and Communication Technology (ICT) Department at Sagaing Education Degree College, began her academic journey by earning a bachelor's degree from the University of Computer Studies (Mandalay) and a B.Ed. degree from Sagaing University of Education in 2018. In August 2019, she published a paper in the International Journal of Trend in Scientific Research and Development (IJTSRD), Volume 3, Issue 5. She then completed her M.C.Sc. in Computer Science at the National Research University of Electronic Technology (MIET-Moscow), Russia, in 2021. More recently, in 2023, she contributed a publication to the International Journal of Tuijin Jishu/Journal of Propulsion Technology, Volume 44, Number 6. In March 2024, she was honoured as a keynote speaker at a webinar hosted by Japan on the topic "The Role of Teacher Educators in the Era of Digital Transformation of Education," organised by The Asia Pacific and Africa Teacher Education Cooperation Center (APATEC²) and the HRD4E Research Lab of Hiroshima University, Japan. Currently, she is pursuing doctoral studies in Information Technology at the University of Technology (Yatanarpon Cyber City). Her research focuses primarily on Data Science, Machine Learning, Deep Learning, and Image Processing.

Moe Phyu Phyu Khaing received M.C.Sc. in Computer Science from the University of Computer Studies, Mandalay (UCSM) in 2022. She currently works as a tutor at the University of Computer Studies (Taunggyi) and is pursuing a Ph.D. in Information Technology at the University of Technology (Yatanarpon Cyber City) .

MYANMAR LANGUAGE QUESTION ANSWERING SYSTEM USING GRU, LSTM AND MBART-LARGE MODEL

Naing Naing Khin[1], Yi Mon Shwe Sin[2], and Win Lei Kay Khine[3]

[1,2,3] Faculties of Computer Science, Computer University
naingk86@gmail.com, yimonshwesinucsy15@gmail.com,
winleikaykhineucsy@gmail.com

ABSTRACT

Natural Language Processing (NLP)-based Question Answering (QA) system is designed to automatically respond to questions posed in natural human language by finding and delivering relevant answers. The farmers in Myanmar face challenges accessing expert guidance about agriculture due to language barriers and lack of technical support. By developing Myanmar-language QA system trained on agricultural texts, manuals and expert knowledge, researchers can provide farmers with accurate answers to practical questions. This study has contributed the development of a Myanmar Language Agricultural QA system by leveraging the Gated Recurrent Unit (GRU) model, a Long Short-Term Memory (LSTM) model and a fine-tuned mBART-large-50-one-to-many-mmt model on a custom-built Myanmar agricultural QA corpus. The corpus comprises 12,600 non-contextual Myanmar agricultural question-answer pairs designed to capture both linguistic nuances and domain-specific knowledge. Experimental results show that the fine-tuned mBART model outperforms both GRU and LSTM baselines, achieving superior accuracy and semantic understanding. The models are evaluated using Precision, Recall and F1 scores to assess model performance and gets the highest F1 score 73%. This work emphasizes for QA research in Myanmar language comprehension and future multilingual question answering systems.

Keywords

GRU, LSTM, mBART-large, QA

1. INTRODUCTION

Agriculture plays in many developing countries including Myanmar where a significant portion of the population depends on farming for their livelihood. This sector contributes around 60% to the country's GDP and employs approximately 65% of the labour force [24]. However, access to timely and practical agricultural knowledge remains a challenge especially for rural communities. Recent advancements in Natural Language Processing (NLP) and Deep Learning (DL) methods have introduced new possibilities for addressing this gap through intelligent QA systems. These systems enable farmers and agricultural professionals to ask questions in their native language and receive accurate, relevant answers. As Myanmar moves toward greater technological integration in its development goals, implementing an agricultural QA system built on local language data and user needs holds the potential to transform rural livelihoods, enhance productivity and contribute to sustainable agricultural growth. Therefore, QA systems serve as valuable tools in research and policy, facilitating better dissemination of findings and ensuring that innovations reach the grassroots level where they can have the most impact [10]. The question answering system can be on the Close Domain and Open Domain types. Closed-domain question answering is mostly applied to task-oriented systems and responds to questions that are topic-specific [1][7][11][22]. Wide volumes of data are needed to give the user the right answer in Open Domain Systems which are not restricted to any one domain. The advantages of computers being able to comprehend logic and language may be

immense. Since the majority of research on natural language processing is done in English, it is hard to predict how well the most advanced models now in use would fare when faced with text data from Myanmar due to the dearth of data in this low-resourced language. Using neural network techniques and fine-tuned Large Language Model (LLM), we have presented a Myanmar Language Question Answering system for Agriculture sector in this study. More accuracy may be achieved with neural networks while handling difficult tasks like text production, document summarization, facial recognition, and so forth. Continuous learning and development of LLMs can be advantageous for QA models [3][4].

2. LITERATURE REVIEW

Deep learning techniques have gained significant traction in question answering (QA) research, especially as alternatives to traditional rule-based methods. Tan et al. (2016) proposed a BiLSTM-based deep learning framework that learns semantic representations of questions and answers using distributed embedding [2]. Their model introduced two key enhancements: a CNN layer on top of the BiLSTM to capture local compositional patterns and an attention mechanism that aligns answer segments with the question context. These facts proved especially beneficial in handling long, noisy answers and semantically divergent QA pairs challenges also common in Myanmar language processing. Given the scarcity of annotated resources and linguistic tools for Myanmar, adopting such neural architectures that rely only on raw text and minimal pre-processing presents a promising direction for building scalable, language-agnostic QA systems in under-resourced settings.

Chen et al. (2016) proposed a GRU-RNN–based model that learns semantic representations of questions and knowledge base triples to perform answer matching directly. Their system introduces a two-column GRU architecture: one for encoding the question and another for encoding the candidate relations or triples and it uses cosine similarity to rank answers. Importantly, the model captures sequential patterns in natural language using GRU cells, allowing it to recognize question types and relational cues even in variable sentence structures [6]. Deep learning–based approach eliminates the need for syntactic parsers or structured queries, making it especially appealing for under-resourced languages like Myanmar.

With the growing demand for intelligent conversational agents, research in chatbot systems has evolved from rule-based models to data-driven deep learning approaches. Dhyani and Kumar (2021) proposed a chatbot system using a Bidirectional Recurrent Neural Network (BRNN) with an attention mechanism. The use of BRNN allowed the model to capture both past and future contexts within a sequence, which is crucial for generating coherent responses in longer and dynamic inputs. The attention layer further enhanced the system's ability to focus on relevant parts of the input sequence, addressing the limitations of simple RNNs in handling long-term dependencies [12]. Their work demonstrates how combining sequence-to-sequence architectures with attention not only improves BLEU score and perplexity but also supports more context-aware and personalized replies.

Nguyen and Le (2016) presented a significant advancement in the domain of question classification by addressing the limitations of traditional feature combination methods. The feature selection algorithm dynamically selects the most relevant features based on question categories. In addition, they introduced question pattern features which combine syntactic and semantic information to help classify unclear questions. Their experiments using the TREC dataset and SVM classifiers demonstrated superior accuracy compared to previous models for fine-grained classification tasks [16]. This work underlines the importance of adaptive feature selection in improving the effectiveness of QA systems and establishes

a new benchmark in question classification research.

Research in QA has advanced rapidly for English like SQuAD and powerful transformer models such as BERT, RoBERTa, and DistilBERT. In contrast, low-resource languages like Bengali lack annotated QA datasets, pre-trained models. While some languages have bridged this gap by translating English datasets for training QA systems, Bengali research has mostly focused on rule-based or information retrieval methods without deep learning. The work by Mayeesha et al. (2021) addresses this gap by translating SQuAD 2.0 into Bengali to create synthetic training data building a human-annotated Bengali QA set from Wikipedia, and testing transformer models in both zero-shot and fine-tuned scenarios [18].

3. THEORY BACKGROUND

3.1. Gated Recurrent Unit

A Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network (RNN) that addresses the vanishing gradient problem inherent in traditional RNNs, allowing them to learn and retain information over longer sequences and used in Question answering, Machine translation, Speech recognition and Sentiment analysis [20][23]. GRUs computationally more efficient and requires fewer parameters while often achieving comparable performance to LSTMs on various sequence modeling tasks. In QA systems, GRUs offer to capture long-range dependencies is crucial for understanding the relationship between a question and relevant information within a potentially long document. GRUs can effectively "remember" key entities and contextual cues from earlier parts of the text which is vital for pinpointing the correct answer. Their computational efficiency translates to faster training times which is beneficial when dealing with large datasets common in QA [14][19]. This efficiency can also make GRU-based QA models more deployable on resource-constrained platforms. The architecture of GRUs has two gates: update gate and reset gate in which the update gate decides how much of the past information to keep and the reset gate does how much of the past information to forget. The GRU can copy or forget past information adaptively, making it powerful for long-term dependencies.

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \quad (1)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \quad (2)$$

$$h_t^{\sim} = \tanh(W_h x_t + U_h(r_t \odot h_{t-1})) \quad (3)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot h_t^{\sim} \quad (4)$$

where x_t = input vector at time step t, h_{t-1}= previous hidden state, h_t= current hidden state, z_t = update gate, r_t= reset gate, $h_t^{\sim}$= candidate hidden state, σ= sigmoid activation function, $\tanh$= tanh activation function and $\odot$= element-wise multiplication.

3.2. Long Short Term Memory

LSTM solves the trouble remembering information over long sequences due to gradients shrinking during backpropagation by introducing gates that control the flow of information. Each cell maintains two states: Hiddden state and Cell state [13][15][17]. It uses three gates to regulate information as Forget gate, Input gate and Output gate. LSTM keeps a separate memory (cell) that can carry information over long distances in the sequence. It decides what to forget, what to add and what to output at each time step.

$$ft = \sigma\left(W_f \cdot \left[h_{\{t-1\}}, x_t\right] + b_f\right) \quad (5)$$

$$i_t = \sigma\left(W_i \cdot \left[h_{\{t-1\}}, x_t\right] + b_i\right) \quad (6)$$

$$\widetilde{C_t} = tanh\left(W_C \cdot \left[h_{\{t-1\}}, x_t\right] + b_C\right) \quad (7)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \quad (8)$$

$$o_t = \sigma\left(W_o \cdot \left[h_{\{t-1\}}, x_t\right] + b_o\right) \quad (9)$$

$$h_t = o_t \odot tanh(C_t) \quad (10)$$

where x_t = input at time step t, h_{t-1}= previous hidden state, C_{t-1}= previous cell state, $W\ and\ b$ = trainable weight, biases,

σ= sigmoid function, tanh= hyperbolic tangent.

Table 1. Hyper parameters in GRU and LSTM

learning_rate	1e-3
embedded_size	128
hidden_size	256
epochs	20
teaching_forcing_ratio	0.5

3.3. mBART-large-50 Model

The mbart-large-50-many-to-many-mmt is a multilingual, pre-trained sequence-to-sequence transformer model from Facebook AI (Meta) which is based on the mBART (Multilingual BART) architecture which began with mBART-25 in 2020 [5][8][9], supporting 25 languages and introducing multilingual denoising pre-training but not supported Myanmar language at that time. In 2021, mBART expanded support to 50 languages, including Myanmar (my_MM) and is designed for many-to-many text generation tasks such as translation, summarization, and question answering. The model learns cross-lingual representations allowing it to perform well even on low-resource languages like Myanmar, Khmer, Lao, etc. The original model size is too large and it is difficult train on our hardware infrastructure. We have fine-tuned this model with our QA corpus and it gives the better result for Myanmar question answering. The fine-tuned question answering model is run on T4 GPU on google Colab library. With large agricultural corpus, adequate disk storage infrastructure with GPU, the model can generate better answers than it currently does for Myanmar language.

Table 2. Hyper parameters in fine-tuned

learning_rate	3e-5
train_batch_size	4
eval_batch_size	4
weight_decay	0.01
epochs	10

4. PROPOSED SYSTEM ARCHITECTURE

There are two stages to the proposed system: training and testing. Prior to training the question answering model, the data is collected as Question-Answer pairs from the available data sources and makes pre-processing. After pre-processing, the model is then trained using GRU, LSTM and fine-tuned mBart-large-50 model. Different score functions describes the model accuracy changes. Precision, recall, F1 score evaluation methods are used to generate the response and assess the model. In testing phase, the input for a user question is pre-processed before feeding into the model and test the system to generate the answer using question answering model.

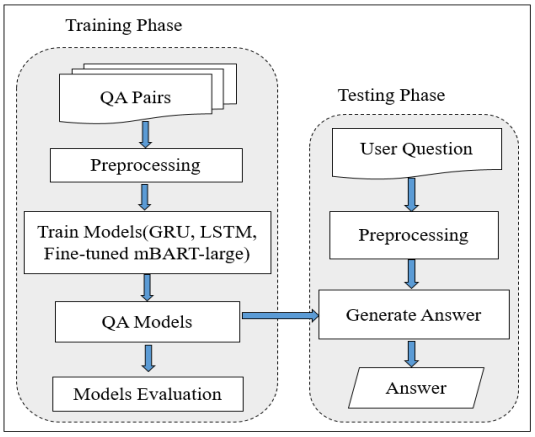


Figure 1. Process Flow of QA system

4.1. Data Collection

Data is essential for training and leaning the neural network models especially with NLP. It is also a time consuming task for languages with little resources. Most NLP models require language-specific processing in addition to a large volume of training data. Similar to other languages, Myanmar's language lacks sufficient resources and it is difficult to locate sufficient data to build training sets. We have gathered a corpus of more than 12k question-answer pairs in order to train the

question answering models. The most information which originates from Green Way agricultural information website [24]. Green Way is a youth-led agricultural expert group that provides technical advisory services. It operates as a legally recognized entity under Grass Root Innovator. The organization disseminates agricultural technologies to farmers across Myanmar. Established in 2011, Green Way received the Social Enterprise Award from the Myanmar Young Entrepreneurs Association (MYEA) in 2016 in recognition of its socially impactful business model. Since then, it has expanded its operations beyond crop production to include livestock farming. Additionally, we also gather data from agricultural books, journals and magazines to learn more information about Myanmar agriculture.

4.2. Data Pre-processing

Data pre-processing helps organize and structure the text so that it can be input into algorithms. Segmentation can be particularly difficult in languages without clear boundaries such Japanese, Chinese and also Myanmar Language. So, segmentation is needed to separate the meaningful word and phrases before training process. The collected data is segmented into word level by using myWord [21] and also manually corrected for some compound words before training the models. Preparing the source and query texts for pre-processing included removing punctuation symbols and correcting numbers. These preparation procedures guarantee that the corpus is ready for a range of NLP applications, which produces more accurate and dependable models.

Before Segmentation:

ကျုံစိုက်ပျိုးရန်အသင့်တော်ဆုံးကာလကိုဖော်ပြပါ။
တောင်ပေါ်တွင်မည်သည့်သီးနှံများစိုက်ပျိုးလေ့ရှိသနည်း။

After Segmentation:

ကျုံ စိုက်ပျိုး ရန် အတွက် အသင့်တော်ဆုံး ကာလ ကို ဖော် ပြ ပါ ။
တောင်ပေါ် တွင် မည်သည့် သီးနှံ များ စိုက်ပျိုး လေ့ ရှိ သ နည်း ။

5. Experimental Result

This is a Question Answering system for farmers and agricultural workers to ask questions that they want to know about farming. We have put up a plan obtain assistance for the field of agriculture. The user can freely inquire about farming techniques and related agricultural knowledge. Agriculture will advance sustainably only when approached as an economic enterprise. Our work has improved automatic question answering utilizing neural network models and fine-tuned large language model with real agricultural data from Myanmar. The system's implementation was processed using the Python programming language and the Pytorch package and run on NVIDIA GeForce RTX 4060 laptop and T4 GPU on google colab library. The sample outputs of QA model are displayed in Table 3.

5.1. Evaluation of Question Answering Model

Evaluating to a QA needs to understand its performance, effectiveness, and ability to produce correct and relevant answers. The three models of GRU, LSTM and mBART-large fine-tuned model are used to test the system. In this test, user-input questions are randomly selected and the precision, recall, and F1 score of the final response are utilized to determine the result. In a quality assurance system, precision focuses on the accuracy of the outcomes by indicating the proportion of right responses. Recall counts the number of right responses that the system correctly recognizes, emphasizing accuracy and comprehensive-ness. The F1 score provides a fair assessment that takes recall and precision into account.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

$$Recall = \frac{TP}{TP + FN} \tag{12}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{13}$$

Table 3. Output of the QA system

Question(User Input)	Referenced Answer	Predicted Answer(Output)
ဂျူစိုက်ပျိုးရန်အသင့်တော်ဆုံးကာလကို ဖော်ပြပါ။	ဂျူ စိုက်ပျိုး ရန် အတွက် အသင့်တော်ဆုံး ကာလ မှာ အောက်တိုဘာ လ နှောင်းပိုင်း နှင့် နိုဝင်ဘာ လ အစောပိုင်း ဖြစ် ပါ သည် ။	ဂျူ စိုက်ပျိုး ရန် အတွက် အသင့်တော်ဆုံး ကာလ မှာ အောက်တိုဘာ လ နှောင်းပိုင်း နှင့် နိုဝင်ဘာ လ အစောပိုင်း ဖြစ် သည် ။
တောင်ပေါ်တွင်မည်သည့် သီးနှံများစိုက်ပျိုးလေ့ရှိသနည်း။	တောင်ပေါ် တွင် ချင်းစိမ်း ၊ နန္နင်း စိုက်ခင်း များ စိုက်ပျိုး လေ့ ရှိ သည် ။	တောင်ပေါ် တွင် ချင်းစိမ်း ၊ နန္နင်း စိုက်ခင်း များ စိုက်ပျိုး လေ့ ရှိ သည် ။
မြန်မာနိုင်ငံတွင် အဓိကသီးနှံသည်ဘာလဲ။	မြန်မာနိုင်ငံ တွင် အဓိက သီးနှံ မှာ ဆန်စပါး ဖြစ် ပါ သည် ။	မြန်မာနိုင်ငံ တွင် အဓိက သီးနှံ မှာ ဆန်စပါး ဖြစ် ပါ သည် ။
ဝတ်မှုန်ကူးခြင်းဆိုသည်မှာအဘယ်နည်း။	ဝတ် မှုန် ကူး ခြင်း သည် အပင် တစ် ပင် ၏ အဖို ဝတ် မှုန် များ ကို အမ ဝတ် မှုန် သို့ ရောက်ရှိ စေ သည့် လုပ်ငန်းစဉ် ဖြစ် သည် ။	ဝတ် မှုန် ကူး ခြင်း သည် အပင် တစ် ပင် ၏ အဖို ဝတ် မှုန် များ ကို အမ ဝတ် မှုန် သို့ ရောက်ရှိ စေ သည့် လုပ်ငန်းစဉ် ဖြစ် သည် ။
ကြံခင်းအတွင်း မည်သည့်အရာကိုဂရုစိုက်ရမည်နည်း။	ကြံ ခင်း အတွင်း ရေ ဝပ် မ နေ စေ ဖို့ ဂရုစိုက် ရ မည် ။	ကြံ ခင်း အတွင်း နိုက်ထရိုဂျင် ကို ဂရုစိုက် ရ ပါ မည် ။

Table 4. Evaluation of Precision, Recall, F1 Score

Models	Precision	Recall	F1 score
GRU	0.6937	0.6923	0.6954
LSTM	0.7023	0.7058	0.7052
Fine-tuned mBART-large	0.7362	0.7314	0.7306

According to Table 4, the fine-tuned mBART-large model achieves the best performance across all metrics, demonstrating the strength of transformer-based architectures and transfer learning in the question answering domain. The LSTM model shows moderate improvement over the GRU and it is more sophisticated memory cell design. The graphical demonstration of evaluating QA models is described in Figure 2.

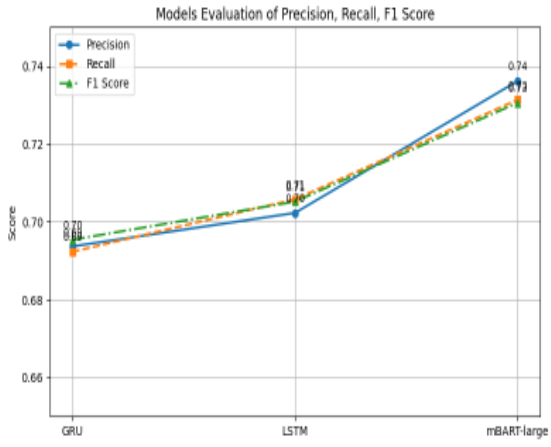


Figure 2. Demonstration of the three models

5.2. Human Evaluation on QA Model

Evaluating the quality and relevance of QA system's answers can also be done by using evaluation methods that involve human judges, experts or users. Manual evaluation requires human evaluators to rate the answer based on criteria such as accuracy, completeness, clarity and relevance. As an alternative, user behavior and satisfaction approaches examine how satisfied users are with the responses provided by the system. These methods can offer more contextual and qualitative insights about the functionality and performance of the system. As human evaluation, some outputs are not completed answer but it can assume acceptable with semantic in language. For example, "When is monsoon rice planted?", (မိုးစပါးကို မည်သည့် အချိန်တွင် စိုက်ပျိုးကြသနည်း။) can answer: "Monsoon rice is planted during the rainy season." (မိုး စပါး ကို မိုးဦးကျ ကာလ တွင် စိုက်ပျိုး ကြ သည်။) or "Monsoon rice is planted in June and July." (မိုး စပါး ကို ဇွန် လ နှင့် ဇူလိုင် လ များ တွင် စိုက်ပျိုး ကြ ပါ သည်။). Human evaluation is necessary for

some situations because of linguistic features in Myanmar language.

6. CONCLUSIONS

Question-answering systems are becoming an essential component of communication as the amount of information in daily life increases. A user using the QA system can communicate in plain language and receive a prompt, concise response. In this study, we used GRU, LSTM and fine-tuned LLM mBart_large to build an automated question answering system for the Myanmar language. One way to evaluate the quality and relevance of the QA system's responses is to compare its output with a collection of approved answers or a reference answer. The Myanmar Language Automatic Question Answering System is still in its infancy and will require further development in terms of data and methodologies. The experimental results demonstrated that the fine-tuned model offers some observations on the Q&A system. Question answering still faces certain difficulties such as the need for ongoing study and development in areas like domain-specific knowledge, ambiguity handling, and ethical system implementation. QA system is critical to bridge the global AI divide and ensure equitable access to information. For the next phase of our research, we will continue exploring large language models (LLMs) in natural language processing for equitable NLP solutions.

REFERENCES

[1] Sweta P. Lende & M.M. Raghuwanshi (2016) "Question Answering System on Education Acts Using NLP Techniques", IEEE Sponsored World Conference on Futuristic Trends in Research and Innovation for Social Welfare.

[2] Ming Tan, Cicero dos Santos, Bing Xiang & Bowen Zhou (2016) "LSTM-BASED DEEP LEARNING MODELS FOR NON-FACTOID ANSWER SELECTION", Workshop track - ICLR 2016.W. P. Pa, Y. K. Thu, A. Finch, and E. Sumita, "A Study of Statistical Machine Translation Methods for Under Resourced Languages", pages 250-257, Procedia Computer Science 81, 2016.

[3] Murong Yue (2025), "A Survey of Large Language Model Agents for Question Answering", Computer Science Department, George Mason University.

[4] Shuai Wang and Yinan Yu (2025), "iQUEST: An Iterative Question-Guided Framework for Knowledge Base Question Answering, Chalmers University of Technology and University of Gothenburg.

[5] Yongqi Fan, Yating Wang, Guandong Wang, Jie Zhai, Jingping Liu, Qi Ye and Tong Ruan (2025), "MinosEval: Distinguishing Factoid and Non-Factoid for Tailored Open-Ended QA Evaluation with LLMs", School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China.

[6] Shini Chen, Jianfeng Wen, and Richong Zhang (2016), "GRU-RNN Based Question Answering Over Knowledge Base", CCIS 650, pp. 80–91, 2016.

[7] Selvakumar G, Thilaheswaran KK and Devadarshan VG (2024), "Question Answering System: An Nlp Based Approach", International Journal of Creative Research Thoughts (IJCRT).

[8] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov Marjan Ghazvininejad, Mike Lewis and Luke Zettlemoyer (2020), "Multilingual Denoising Pre-training for Neural Machine Translation", Facebook AI Research.

[9] Leon Englander, Hannah Sterz, Clifton Poth, Jonas Pfeiffer, Ilia Kuznetsov and Iryna Gurevych (2024), "M2QA: Multi-domain Multilingual Question Answering", Ubiquitous Knowledge Processing Lab, Technical University of Darmstadt, Language Technology Lab, University of Cambridge.

[10] Reem Alqifari (2019), "Question Answering Systems Approaches and Challenges", Proceedings of the Student Research Workshop associated with RANLP-2019, pages 69–75, Varna, Bulgaria, Sep 2–4, 2019.

[11] Dongfang Han, Turdi Tohti and Askar Hamdulla (2022), "Attention-Based Transformer-BiGRU for Question Classification", Information 2022, 13, 214.

[12] Manyu Dhyani and Rajiv Kumar (2021), "An intelligent Chatbot using deep learning with Bidirectional RNN and attention model", Volume 34, Part 3, 2021, Pages 817-824.

[13] Giovanni Di Gennaro, Amedeo Buonanno, Antonio Di Girolamo, Armando Ospedale and Francesco A.N. Palmieri (2020), "Intent Classification in Question-Answering Using LSTM Architectures", Jan 15, 2020.

[14] Emmanuel Mutabazi, Jianjun Ni, Guangyi Tang and Weidong CaoWeidong, "A Review on Medical Textual Question Answering Systems

Based on Deep Learning Approaches", Appl. Sci. 2021.

[15] Zahra Abbasiantaeb and Saeedeh Momtazi, "Text-based Question Answering from Information Retrieval and Deep Neural Network Perspectives: ASurvey", 27 May 2020.

[16] Van-Tu Nguyen and Anh-Cuong Le, "Improving Question Classification by Feature Extraction and Selection", May 2016.

[17] Moneerh Aleedy, Hadil Shaiba and Marija Bezbradica, "Generating and Analyzing Chatbot Responses using Natural Language Processing", IJACSA, Vol. 10, No. 9, 2019.

[18] Tasmiah Tahsin Mayeesha, Abdullah Md Sarwar and Rashedur M. Rahman (2021), "Deep learning based question answering system in Bengali", Journal of Information and Telecommunication VOL. 5, NO. 2, 145–178, 2021.

[19] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi and Hannaneh Hajishirzi, "Bi-Directional Attention Flow For Machine Comprehension", 4 January 2020.

[20] Atul Harsha, "Introduction to Recurrent neural networks and the Math behind them", Data Science Articles, Dec 26, 2022.

[21] https://github.com/ye-kyaw-thu/myWord, (2022).

[22] Rohit Arora, Parth Singhk, Hemlata Goyal, Sunita Singhal and Smita Vijayvargiya, "Comparative Question Answering System based on Natural Language Processing and Machine Learning", Proceedings of the International Conference on Artificial Intelligence and Smart Systems (ICAIS-2021).

[23] Felix Abrahamsson, "Designing a Question Answering System in the Domain of Swedish Technical Consulting Using Deep Learning", June 27, 2018.

[24] https://greenwaymyanmar.com/

Authors' Biography

First A. Author: In 2010, Naing Naing Khin earned her Master of Computer Science (M.C.Sc.) degree from the University of Computer Studies, Yangon, Myanmar. She has her Ph.D. in 2024 at the University of Computer Studies, Yangon, Myanmar. She was a member of the Natural Language Processing and Speech Processing Lab at UCSY. Now she is serving at the Myanmar Language Improvement and Literature Improvement Project (MLLIP) in Naypyitaw, Myanmar. Her research interests are Natural Language Processing, Machine Learning, and Conversational AI.

Second B. Author: Yi Mon Shwe Sin got her Master of Computer Science (M.C.Sc.) degree from the University of Computer Studies, Yangon, Myanmar in 2010. She has her Ph.D. in 2019 at the University of Computer Studies, Yangon, Myanmar. She served as a teacher and lab member at the Natural Language Processing Lab at UCSY. She made research at A*STAR Institute for Infocomm Research (A*STAR I²R). She also participated in The Workshop on Asian Translation (WAT) in 2018, 2019. Now she is serving at the Myanmar Language Improvement and Literature Improvement Project (MLLIP) in Naypyitaw, Myanmar. Her research interests are Natural Language Processing, Machine Translation.

Third C. Author: Win Lei Kay Khine earned her Master of Computer Science (M.C.Sc.) degree from the University of Computer Studies, Yangon, Myanmar in 2010. She has her Ph.D. in 2024 at the University of Information Technology (UIT), Myanmar and is a member of the Natural Language Processing Lab at UIT. Now she is serving at the Myanmar Language Improvement and Literature Improvement Project (MLLIP) in Naypyitaw, Myanmar. Her research interests are Natural Language Processing, AI related work.

PREDICTION OF TERRORIST ACTIVITIES USING CONVOLUTIONAL NEURAL NETWORK AND SHALLOW NEURAL NETWORK

Htet Htet Wah Lwin[1], and Thandar Aung[2]

[1,2]Faculty of Information Science, University of Computer Studies (Mandalay)
[1]htethtet611998@gmail.com

ABSTRACT

In the recent century, terrorist attacks have represented a massive concern for both developed and developing countries alike. In this paper, deep learning-based prediction models are proposed. This paper explains how to use deep learning, convolutional neural network, for terrorism prediction by using global terrorism database (GTD) for achieving the prediction of terrorism accuracy improvement. Raw data from the GTD is complex, diverse, and high-dimensional. Pre-processing is essential to convert this unstructured or semi-structured data into a clean and deep-learning-compatible format. Five different models are based on Convolutional Neural Network (CNN) and Shallow Neural Network. The system is implemented by the ratio of data size: (Training – 80%, Testing – 20%). In the end, the performance of the models are compared in terms of accuracy, precision, recall and F-measure metrics. Finally, show the result of analysis which regions are highest attack frequencies in global context. This paper applies Python Programming language.

Keywords

Terrorist activities, Prediction, Shallow Neural Network, Convolutional Neural Network, GTD

1. INTRODUCTION

One of the most important threats to today's civilization is terrorism. The definition of terrorism according to Hoffman [5] is "the deliberate creation and exploitation of fear through violence or the threat of violence in the pursuit of political change." There were over 200,000 terrorist events worldwide in the Global Terrorism Database (GTD) [14], with more than 3,000 terrorist organizations causing varying degrees of harm to 205 countries. According to Global Terrorism Database (GTD), in 2019 alone 1,411 different terrorist attacks have happened, causing 6,362 fatalities and affecting the quality of life of individuals in the society. Terrorism has been studied for decades to understand the major factors causing the act of terrorism or understanding how to perform counterterrorism or understanding the social and economic effects of terrorism [9, 15]. However, because of the complex nature of terrorism, it is difficult to find an effective solution that can be used as a counterterrorism to protect the lives of individuals. Identification of terrorist ideologies and prediction of future terrorist attacks have been proven to be of great importance and time-consuming process.

Machine learning algorithms have been used recently to study the different factors of terrorism [1, 17]. Global Terrorism Database (GTD), which comprises thousands of features spanning geographical, temporal, categorical, and numeric variables. Traditional models, such as decision trees and logistic regression, often struggle with such complexity, whereas deep neural networks can automatically discover and utilize hidden relationships among features [18]. Deep learning models are distinguished from traditional machine learning models by their ability to automatically learn hierarchical features from raw data with minimal manual intervention [10].

Shallow Neural Network and particularly Convolution Neural Network (CNN) are getting popularity mainly because of the fact that a huge amount of labelled data is available recently. In this paper, Shallow Neural Network and CNN models are used to make predictions

of different factors that lead to terrorist activities.

2. LITERATURE REVIEW

Mo et al. [13] used the GTD to construct models that predict terrorist events. They employed three different algorithms: naive Bayes, support vector machine and logistic regression. To enhance accuracy, they utilized two methods for attribute selection which are maximal relevance and minimal redundancy maximal relevancy. They found that the logistic regression classifier achieves 78.41%, emphasizing the feasibility of using machine learning techniques in the domain of terrorism. Also, they demonstrated that choosing the proper attribute selection methods would increase model accuracy.

Maniraj et al. [12] developed a system that examines the growth or decay of the terrorist groups by the time, location, type of attack, target motives, weapon type, and availability. They analyzed the GTD dataset and used machine learning algorithm that can predict the probability of attacks in different regions.

Kalaiarasi et al. [7] developed multiple classifiers to group and predict different terrorist activities using k-NN algorithm and Random Forest techniques. They used the GTD dataset for detection of terrorism.

All previous studies have applied machine learning and deep learning techniques to make AI-based model for terrorism. . This paper compares the performance of shallow neural network and convolution neural network and concludes that convolution neural network is a suitable model for prediction of terrorist activities.

3. ABOUT SYSTEM

The system developed in this study is designed to predict various characteristics of terrorist attacks using deep learning models, specifically Convolutional Neural Network (CNN) and Shallow Neural Network (SNN). The process begins with the Global Terrorism Datset(GTD). The dataset includes over 200,000 attack records with features such as weapon type, target type, attack success, and whether the attack was a suicide mission. To prepare the data for modeling, preprocessing steps were applied, including handling missing values using SimpleImputer, encoding categorical features via LabelEncoder, balancing imbalanced classes using the SMOTE technique, and scaling numerical features with MinMaxScaler. The cleaned dataset was then split into training and testing sets using an 80:20 ratio to ensure robust evaluation. The training set was used to train CNN and SNN models for multiple predictive tasks such as determining the success of the attack, whether it was a suicide attack, the likely weapon used, the type of attack, and the region of the attack.

Once the models were trained, the test set was passed through these trained models to evaluate their generalization performance. Evaluation metrics included **accuracy**, **precision**, **recall**, and **F1-score**, which provided a comprehensive understanding of the model performance across all prediction tasks. Among the tasks, the regional analysis was particularly significant, as it enabled the identification of geographical zones most frequently targeted by terrorism. These analyzed results were then visualized and displayed in the system using clear and interpretable outputs. The system interface allows users to either manually input features through dropdowns or upload CSV files for batch predictions. The accuracy of each model and the corresponding prediction outputs are shown interactively, enabling an intuitive understanding of how deep learning models can be used in a counterterrorism context.

The proposed system design is described in Figure 1.

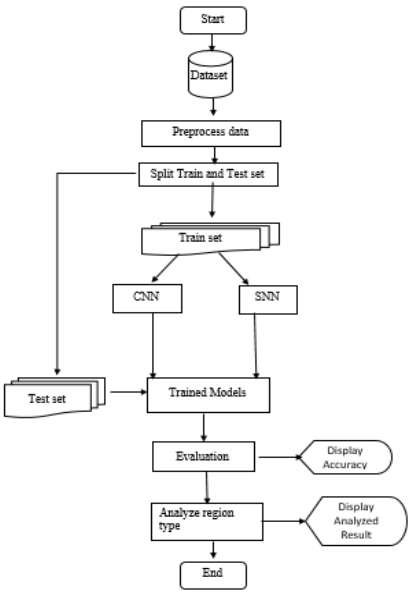


Figure 1. Proposed System Design

3.1 About Dataset

In this paper, a detailed analysis of the dataset is given.The preprocessing performed on the dataset. The National Consortium for the Study of Terrorism and Responses to Terrorism (START) has prepared a dataset known as Global Terrorism Database (GTD) (https://www.start.umd.edu/gtd). GTD contains information about terrorist activities from 1970 until 2021, including more than 214,000 different instances of terrorism. In this paper, 34 attributes are taken. The system is implemented by the ratio of data size: (Training – 80%, Testing – 20%).This is data variables used in this system (iyear, imonth, iday, Extended, Provstate, Latitude, Longitude, Specificity, Vicinity, Crit1, Crit2, Crit3, Doubtterr,Multiple, Natlty1, Propextent, Ishostkid, Ransom, Country, City, Gname, Individual, Nkillus, Nkillter, Nwound, Nwoundus, Nwoundte ,Property, Targtype1, Suicide, Success, Weaptype1, Region, Attacktype1).

3.2 Data Pre-processing

Text to Numbers: The dataset contains a mix of categorical and numerical features. Categorical variables—such as attack type, weapon type, and region—were transformed using Label Encoding class of sklearn library is used to convert nonnumeric data to numeric data

Missing Data: The dataset contains many missing values, i.e., the cell does not contain any data, which results into NaN when processed by CNN and Shallow Neural Network. In this paper, SimpleImputer of sklearn library is used to fill the missing data. We have replaced the missing values by mean along each column.

Normalization: In GTD, data are in different range. Some columns have values as 0 and 1, while others have values in hundreds or thousands. In this situation, it is difficult for learning algorithm to learn the pattern and converge to a global minimum. Therefore, it is important that before the data are processed by a learning model, the data are normalized, i.e., in the range of 0 to 1 or − 1 to 1. In this paper, MinMaxScalar of sklearn library is used, which for each value of the feature subtracts the average of all values and divides it by standard deviation, to convert the data in the range of 0 to 1.

$$Z_i = \frac{X_i - \bar{x}}{S} \quad (1)$$

Where, Xi=all the samples for givenfeature
$\bar{x}$=the average of all samples by the feature
S=the standard deviation

Dealing with Unbalanced Classes: During the analysis of the dataset, it is observed that the data are not balanced in different classes. In some classes, there are more instances, while others have very few instances. Shallow Neural Network and CNN trained on unbalanced data are biased towards the classes having more instances. In order to keep the data in balanced form, SMOTE: Synthetic Minority Oversampling Technique presented by Chawla et al. in 2002 [3] and later made available as a tool to be used in Python in is used. It was used to balance the distribution of classes in the training set by generating synthetic examples of the minority class.

4. METHODOLOGY

4.1 Convolutional Neural Network

Among deep learning approaches, Convolutional Neural Networks (CNNs) have been recognized for their ability to extract hierarchical feature representations and automatically learn spatial or local patterns in data. Although CNNs were originally designed for grid-structured visual data, researchers have successfully adapted their principles to structured and sequential data, including text classification [4] and time series prediction [19].A Convolutional Neural Network has an input layer followed by a convolutional layer,pooling layers ,fully connected layer and output layer[11].

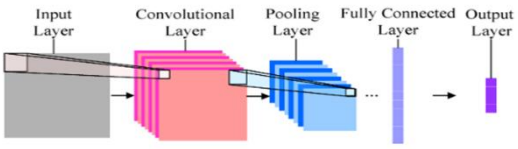


Figure 2. General Block Diagram of Convolutional Neural Network

Convolution layers: Apply a set of learnable filters (kernels) over the input data to capture local patterns.

$$z = \sum_{i=1}^{n=filter-size} w_i x_i + b \quad (2)$$

Activation functions:Usually ReLU (Rectified Linear Unit), introducing non-linearity. It returns the input if it is positive, and returns 0 if it is negative.

$A_i = \max(0, z_i)$ (3)

Pooling layers: Reduce the spatial or temporal resolution of the feature maps, typically using max-pooling or average pooling, to make representations more robust.

$P= (A_{m,n})$ (4)

Fully connected layers: Transform high-level features into final predictions.The output of a FC layer is computed by applying a weight matrix and a bias vector to the input, followed by an activation function.

4.2 Shallow Neural Network

Shallow Neural Network (SNN) is a type of artificial neural network characterized by having a minimal number of hidden layers, typically one or two. This structure contrasts with **Deep Neural Networks (DNNs)**, which comprise multiple hidden layers [12]. Understanding SNNs is fundamental to grasping the basics of neural network architectures and their applications. An SNN consists of the following components:

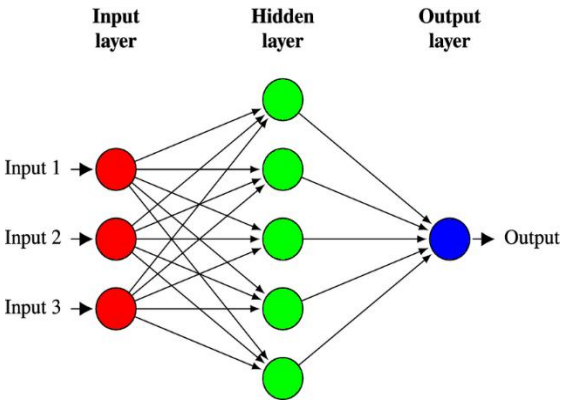


Figure 3. General Block Diagram of Shallow Neural Network

Input Layer: Receives the input features of the dataset.

Hidden Layer(s): Processes inputs through neurons, applying weights and activation functions to capture patterns in the data.

$z=x. w+b$ (5)

Output Layer: Produces the final prediction or classification based on the processed information from the hidden layer(s).

The simplicity of SNNs makes them suitable for problems where the relationship between input and output is relatively straightforward and can be captured with limited hierarchical feature representation. Each neuron in an SNN performs a weighted sum of its inputs, adds a bias term, and passes the result through an activation function.

4.3 Sample Calculation of Convolution Neural Network (Only success activity)

CNN is mainly used to capture topology through feature extraction by applying filters to batches of data points. This can be used to capture data and therefore it is used extensively in image processing, speech recognition, time series analysis, etc. One of the most popular architectures used in deep learning is the Convolutional Neural Network (CNN), commonly used for prediction tasks. A CNN has an input layer followed by a convolutional layer, pooling layers, and output layer. Here is an example calculation for CNN with 7 features to predict success

Convolution Layer: Multiplication apply filter sized patch of input and weight then addition to bias

Input	1	1	1	2	6	6	3

Weight	0.12	0.11	0.2	0.15	0.25

Bias	0.1

Output Z	2.33	3.13	3.29

After Apply ReLU Activation,A=max(0,Z)
= [2.33, 3.13, 3.29]
MaxPoolingLayer P=max(A0,A1)
= [3.13, 3.29]
Fully connected layer =weight*output of maxpooling layer+bias
Input = [3.13,3.29]
Weight= [0.5, 0.6] bias= [0.1]
Output of fully connected layer= [3.639]
Apply Sigmoid Activation function

$\sigma(x)=\frac{1}{1+e^{-x}}$

≈ 0.97
Loss(Mean Square Error)

$L=\frac{1}{2}(y-\hat{y})^2=\frac{1}{2}(1-0.91)^2=4.05\times10^{-3}$

During the back-propagation process, the derivative of the loss is taken for all layers and weights are updated using optimization technique, such as gradient descent. The gradient of the loss with respect to the weights is computed and used to iteratively update the parameters, minimizing the loss [16].
$\frac{dL}{d\hat{y}}=-(y-\hat{y})=(\hat{y}-y)$

Sigmoid gradient:
$\frac{d\hat{y}}{dz_2}=\hat{y}(1-\hat{y})$

Gradient of loss with Output Layer
$\frac{dL}{dz_2}=\frac{dL}{d\hat{y}}\frac{d\hat{y}}{dz_2}=(\hat{y}-y)\,\hat{y}(1-\hat{y})=-7.37\times10^{-3}$

Gradient of loss with Fully Connected Layer weight and bias
$\frac{dL}{dw_{fc}}=\frac{dL}{dz_{fc}}.\frac{dz_{fc}}{dw_{fc}}=(\hat{y}-y).\,\hat{y}(1-\hat{y}).P$
$=[-0.0025,-0.0026]$
$\frac{dL}{db_{fc}}=(\hat{y}-y).\,\hat{y}(1-\hat{y})=-0.0008$

Gradient of MaxPooling Layer
$\frac{dL}{dP}=\frac{dL}{dz_{fc}}.\frac{dz_{fc}}{dP}=(\hat{y}-y).\,\hat{y}(1-\hat{y}).\,w_{fc}$
$=[-4\times10^{-4},-4.8\times10^{-4}]$

Gradient with ReLu Activation
$\frac{dL}{dA}=[0,-4\times10^{-4},-4.8\times10^{-4}]$

The ReLu gradient is then applied:
$\frac{dL}{dz_i}=\frac{dL}{dA}.1(z_i>0)$
$=[0,-4\times10^{-4},-4.8\times10^{-4}]$

Gradient with Convolution Layer weight and bias
$\frac{dL}{dw_j}=\sum_{i=0}^{2}\frac{dL}{dz_i}.x_{i+j}$,j=0,1,2,3,4

$\frac{dL}{dw_0}=\frac{dL}{dz_0}x_0+\frac{dL}{dz_1}x_1+\frac{dL}{dz_2}x_2=-8.8\times10^{-4}$

$\frac{dL}{dw_1}=\frac{dL}{dz_0}x_1+\frac{dL}{dz_1}x_2+\frac{dL}{dz_2}x_3=-1.36\times10^{-3}$
$\frac{dL}{dw_2}=\frac{d}{dz_0}x_2+\frac{dL}{dz_1}x_3+\frac{dL}{dz_2}x_4=-3.68\times10^{-3}$

$\frac{dL}{dw_3}=\frac{dL}{dz_0}x_3+\frac{dL}{dz_1}x_4+\frac{dL}{dz_2}x_5=-5.28\times10^{-3}$
$\frac{dL}{dw_4}=\frac{dL}{dz_0}x_4+\frac{dL}{dz_1}x_5+\frac{dL}{dz_2}x_6=-3.84\times10^{-3}$
$\frac{dL}{dw_{conv}}=[-8.8\times10^{-4},-1.36\times10^{-3},-3.68\times10^{-3},-5.28\times10^{-3},-3.84\times10^{-3}]$
$\frac{dL}{db_{conv}}=\sum_{i=0}^{2}\frac{dl}{dz_i}=-8.8\times10^{-4}$

Weight and bias Update

Learining rate is a hyperparmeter which means that we need to manually gusess its value. Assume learning rate α=0.1

1. Fully Connected Layer

Update weight and bias
$w_{fc-new=w_{fc}}-\propto\frac{dL}{dw_{fc}}$=[0.5,0.6]-0.1[-0.0025,-0.0026]
=[0.50025,0.60026]
$b_{fc-new=b_{fc}}-\propto\frac{dL}{db_{fc}}$=0.1-0.1(-0.0008)=0.10008

2. Convolution Layer

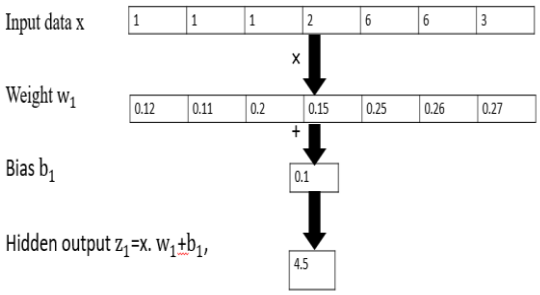
Update weight and bias
$w_{conv-new=W_{conv}}-\propto\frac{dL}{dw_{conv}}$
=[0.120088,0.110136,0.200368,0.150528,0.250384]
$b_{conv-new=b_{conv}}-\propto\frac{dL}{db_{conv}}$=0.1-0.1(-8.8 × 10^{-4})=0.10008

Repeat the same process of backward and forward pass until Loss is equal to zero or after next iteration, Loss value is equal to the previous iteration of Loss value.

4.4 Sample Calculation of the Shallow Neural Network (Only success activity)

A Shallow Neural Network is a type of artificial neural network with hidden layer between the input and output layers. SNN retain powerful approximation capabilities for a wide variety of classification tasks, including structured data prediction problems such as terrorism event modeling [6,2].Components of shallow neural network include input layer, hidden layer and output layer.

Here is an example calculation for SNN with 7 features to predict success



Apply Activation function
A=Relu(0,z)=max(0,z)
=max (0,4.5)=4.5
Output Layer:
w_2 =0.5, b_2=0.1
$z_2=A.w_2+b$
=4.5.0.5+0.1

=2.35

Apply sigmoid activation function,

$\sigma(x)=$

$\hat{y} = \frac{1}{1+e^{-z_2}} = \frac{1}{1+e^{-2.35}}$

≈ 0.91

Loss (Mean Square Error)

$L=\frac{1}{2}(y-\hat{y})^2 = \frac{1}{2}(1-0.91)^2$

$= 4.05\times10^{-3}$

During the back-propagation process, the derivative of the loss is taken for all layers and weights are updated using optimization technique, such as gradient descent.

$$\frac{dL}{d\hat{y}}=-(y-\hat{y})=(\hat{y}-y)$$

Sigmoid gradient:

$\frac{d\hat{y}}{dz_2} = \hat{y}(1-\hat{y})$

Gradient of loss with Output Layer

$\frac{dL}{dz_2}=\frac{dL}{d\hat{y}}\frac{d\hat{y}}{dz_2}=(\hat{y}-y)\,\hat{y}(1-\hat{y})=-7.37\times10^{-3}$

Gradient of loss with weight w_2 and bias b_2

$\frac{dL}{dw_2}=\frac{dL}{dz_2}.\frac{dz_2}{dw_2}=(\hat{y}-y).\hat{y}(1-\hat{y}).A$

$=7.37\times10^{-3}\times4.5=-0.033$

$\frac{dL}{db_2}=\frac{dL}{dz_2}.\frac{dz_2}{db_2}=-7.37\times10^{-3}$

Gradient of loss with A

$\frac{dL}{dA}=\frac{dL}{dz_2}.\frac{dz_2}{dA}=-7.37\times10^{-3}.0.5=-3.68\times10^{-3}$

Gradient of loss with z_1 hidden layer

$\frac{dL}{dz_1}=\frac{dL}{dA}.\frac{dA}{dz_1}=-3.68\times10^{-3}.1=-3.68\times10^{-3}$

Gradient of loss with weight and bias

$\frac{dL}{dw_i} = \frac{dL}{dz_1}.\frac{dz_1}{dw_i}=\frac{dL}{dz_1}.x_i,\ i=1,2,3,4,5,6,7$

$\frac{dL}{dw_1}=-3.68\times10^{-3}.1=-3.68\times10^{-3}$

$\frac{dL}{dw_2}=-3.68\times10^{-3}.1=-3.68\times10^{-3}$

$\frac{dL}{dw_3}=-3.68\times10^{-3}.1=-3.68\times10^{-3}$

$\frac{dL}{dw_4}=-3.68\times10^{-3}.2=-7.36\times10^{-3}$

$\frac{dL}{dw_5}=-3.68\times10^{-3}.6=-2.208\times10^{-2}$

$\frac{dL}{dw_6}=-3.68\times10^{-3}.6=-2.208\times10^{-2}$

$\frac{dL}{dw_7}=-3.68\times10^{-3}.3=-1.1\times10^{-2}$

$\frac{dL}{dw_1}=[-0.003,-0.003,-0.003,-0.007,-0.022,-0.022,-0.011]$

$\frac{dL}{db_1}=\frac{dL}{dz_1}.\frac{dz_1}{db_1}=\frac{dL}{dz_1}.1=-3.68\times10^{-3}$

Weight and bias Update

Learining rate is a hyperparmeter, which means that we need to manually gusess its value. Assume learning rate α=0.1

1. Output Layer

Update weight and bias

$w_{2-new=W_2}- \propto \frac{dL}{dw_2}=0.5-0.1(-0.033)$

$b_{2-new=b_2}- \propto \frac{dL}{db_2}=0.1-0.1(-0.0007)=0.1007$

2. Hidden Layer

Update weight and bias

$w_{1-new=W_1}- \propto \frac{dl}{dw_1}$

=[0.1203,0.1103,0.2003,0.1507,0.2522,0.2622, 0.2711]

$b_{1-new=b_1}- \propto \frac{dl}{db_1}=0.1-0.1(-3.68\times10^{-3})=0.10003$

Repeat the same process of backward and forward pass until Loss is equal to zero or after next iteration, Loss value is equal to the previous iteration of Loss value.

5. PERFORMANCE EVALUATION

In this paper, the performance of the predictor is evaluated through most familiar metrics such as overall accuracy, precision, recall and F-measure.

True positive (TP) =when both the actual and predicted values are 1.

True negative (TN) =when both the actual and predicted values are 0.

False positive (FP) =when the actual value is 0 but the predicted value is 1.

False negative (FN) =when the actual value is 1 but the predicted value is 0.

5.1. Precision

Precision [4] the number of true positives divided bythe number of true positives plus the number of false positives. The equation is

$$Precision = \frac{TP}{TP+FP}$$

5.2. Accuracy

In the prediction task, prediction accuracy is the number of true positives (TP) divided by the total number of elements belonging to the positive class. The equation is described as the following.

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP}$$

5.3. Recall

Recall [4] the number of true positives divided bythe number of true positives plus the number of false negative. The equation is

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

5.4. F-measure

The **F1 score** combines these two metrics into one value to give a balanced evaluation:

$$\text{F} - \text{measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The key metrics of performance measures (accuracy, recall, f1-score, and precision) are evaluated for this proposed system analysis. Table 5.1 and 5.2 show models accuracy, precission, recall, f1comparative results for proposed model Convolutional Neural Network, and Shallow Neural Network.,Figure 6 and Figure 7 show the result which regions are highest attack frequencies in global context.

Table I Models Train and Test Results for CNN

	CNN Train				CNN Test			
Task	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Success	0.9	0.87	0.93	0.9	0.89	0.87	0.93	0.9
Suicide	0.98	0.98	0.99	0.98	0.98	0.98	0.99	0.98
Attack	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.88
Region	0.99	0.99	0.99	0.99	0.98	0.98	0.98	0.98
Weapon	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91

Table II Models Train and Test Results for SNN

	SNN Train				SNN Test			
Task	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Success	0.85	0.87	0.83	0.85	0.85	0.87	0.82	0.85
Suicide	0.98	0.98	0.98	0.98	0.98	0.97	0.98	0.98
Attack	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
Region	0.92	0.83	0.68	0.71	0.92	0.83	0.68	0.7
Weapon	0.87	0.87	0.87	0.87	0.87	0.87	0.87	0.87

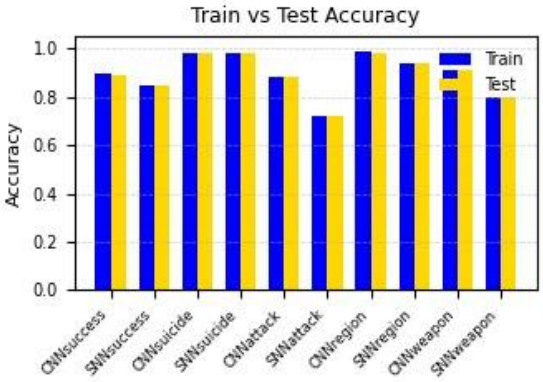


Figure 4. Train and Test Accuracy Based on CNN and SNN

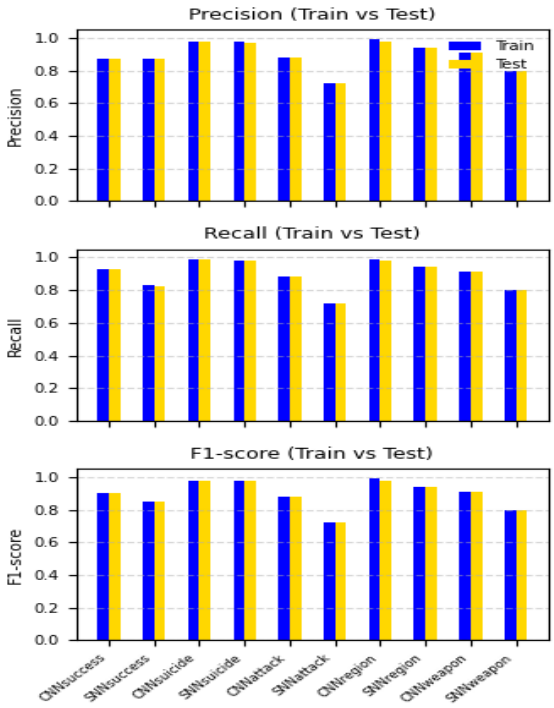


Figure 5. Train and Test Precisson,Recll,F1 Based on CNN and SNN

5.5 Analysis of Regions which are highest attack

The worldwide distribution of attacks due to region is illustrated in Fig. 5.3.According to the data, attacks occur most frequently in Middle East and North America, where they make up 27.8% of all attacks. With 26.1%, the South Asia region comes in second. With respective contributions of 11.3% and 9.4%, Sub-Saharan Africa and South America are also heavily affected. Smaller but not insignificant shares are shown by other regions like Southeast Asia, Eastern Europe, and Western Europe. The graphic emphasizes the need for

regionally focused security and policy interventions by showing a geographic concentration of attacks in particular regions, especially Asia and North Africa. Fig 5.4 shows how terrorist attacks are distributed throughout the world, showing both the total number of attacks and their corresponding percentages. These differences highlight the socioeconomic and geopolitical factors that affect how common terrorism is in various regions of the world.

Figure 6. Regions of Highest Attack

	Region	Attack Count	Percentage (%)
○	Middle East and North Africa	58252	27.8
○	South Asia	54725	26.1
○	Sub Saharan Africa	23746	11.3
○	South America	19846	9.5
○	Western Europe	17328	8.3
○	Southeast Asia	14498	6.9
○	Central America	10386	5
○	Eastern Europe	5326	2.5
○	North America	3847	1.8
○	East Asia	847	0.4
○	Central Asia	580	0.3
○	Australia and Oceania	325	0.1

Figure 7. Regional Distribution of Terrorist Attacks

6. CONCLUSION

Deep learning-based solutions to understand the different models of terrorism that can help us make prediction in future terrorist activities. This paper identify ten different models that are important to predict for counterterrorism based on CNN and SNN. These models are whether the type of attack is suicide or not, whether the attack is successful or not, what type of weapon can possibly be used, what region can possibly be targeted, and what type of terrorist is going to be used. The results demonstrate the comparison of accuracy, f-measure, and precision and recall indicating the capacity. The analysis and prediction of regions most vulnerable to terrorist activities constituted a significant component of this study. An important part of this study involved analyzing and predicting the region's most at risk for terrorist attacks. In addition to helping understand previous attack trends, regional analysis is essential for enabling the proactive implementation of counterterrorism measures in high-risk areas. The Global Terrorism Database (GTD) shows that attacks have been concentrated in certain regions, particularly the Middle East and North Africa, South Asia, and Sub-Saharan Africa, which together account for more than 65% of all attacks worldwide.

7. REFERENCES

[1] Alhamdani, R., Abdullah, M., & Sattar, I. (2018). Recommender system for global terrorist database based on deep learning. *International Journal of Machine Learning and Computing*, 8, 571–576

[2] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer

[3] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357.

[4]Davis, J., & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd International Conference on Machine Learning*.

[5] Hoffman, B. (2013). Al Qaeda's uncertain future. *Studies in Conflict & Terrorism*, 36(8), 635–653.

[6] Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2), 251–257.

[7]. Kalaiarasi, S., Mehta, A., Bordia, D., & Sanskar. (2019). Using global terrorism database (GTD) and machine learning algorithms to predict terrorism and threat. *International Journal of Engineering and Advanced Technology*, 9(1), 5995–6000.

[8] Kim, Y. (2014). Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 1746–1751..

[9] Krieger, T., & Meierrieks, D. (2011). What causes terrorism? *Public Choice*, 147(1–2), 3–27.

[10] LeCun, Y., Bengio, Y., & Hinton, G. (2015).

Deep learning. *Nature*, 521(7553), 436–444.

[11] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

[12] Maniraj, S. P., Chaudhary, D., Deep, V. H., & Singh, V. P. (2019). Data aggregation and terror group prediction using machine learning algorithms. *International Journal of Recent Technology and Engineering*, 8(4), 1467–1469.

[13] Mo, H., Meng, X., Li, J., & Zhao, S. (2017). Terrorist event prediction based on revealing data. *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, 239–244.

[14]National Consortium for the Study of Terrorism and Responses to Terrorism (START). (2021). *The Global Terrorism Database (GTD)*. UniversityofMaryland. https://www.start.umd.edu/gtd

[15] Ouassini, N., & Verma, A. (2018). Socio-economic inequality or demographic conditions: a micro level analysis of terrorism in Jharkhand. *Journal of Victimology and Victim Justice*, 1(1), 63–84.

[16] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536.

[17] Saeed, Y., Ahmed, K., Zareei, M., Zeb, A., Vargas-Rosales, C., & Awan, K. M. (2019). In-vehicle cognitive route decision using fuzzy modeling and artificial neural network. *IEEE Access*, 7, 20262–20272.

[18] Schmidhuber, J. (2015). *Deep learning in neural networks: An overview*. Neural Networks, 61, 85–117.

[19] Zhang, Q., Yang, L., & Rong, Y. (2021). Time series prediction with 1D convolutional neural networks: A review. Neural Computing and Applications, 33(10), 4511–4528.

Authors
Biography

Htet Htet Wah Lwin achieved the B.C.Sc in 2020 from the University of Computer Studies,Mandalay .She is currently pursuing as M.C.Sc candidate at the same university.

Dr.Thandar Aung
Professor
Faculty of Information Science, University of Computer Studies (Mandalay),

IMAGE CAPTIONING USING DENSENET FEATURE EXTRACTION AND LSTM-BASED SEQUENCE GENERATION

Hsu Myat Htet[1], and Yu Ya Win[2]

[1,2] Faculty of Computer Science, University of Computer Studies (Mandalay)
hsumyathtet1111@gmail.com, yuyawin@gmail.com

ABSTRACT

An image captioning system is an artificial intelligence application that integrates computer vision with natural language processing to create textual descriptions of visual content. It employs an encoder-decoder framework to interpret image features and generate relevant captions. In this setup, DenseNet—a highly efficient Convolutional Neural Network (CNN)—serves as the encoder to identify and extract significant image features. These extracted features are then input into an LSTM (Long Short-Term Memory) network, which functions as the decoder, generating captions sequentially, word by word. The system is trained using the Flickr8k Captions dataset, which includes images paired with corresponding human-written descriptions. Once trained, the model generates captions for unseen test images. Its performance is assessed using the BLEU score, a widely accepted metric that evaluates how closely the generated captions match the reference ones. The primary objective of this system is to produce meaningful and accurate image descriptions, with BLEU score results indicating its overall effectiveness.

KEYWORDS

image captioning, DenseNet, CNN, LSTM, Flickr8k, BLEU

1. INTRODUCTION

Image captioning is an interdisciplinary problem at the intersection of computer vision and natural language processing that generates textual descriptions of visual data automatically. It allows machines to precisely describe and compute on the contents of images using human language, which powers a wide range of applications, including visual aid for visually impaired people, automatic image indexing and content-based image retrieval. Recently, development of deep learning methods has remarkably enhanced the performance both in terms of image captioning accuracy and fluency supported by strong neural network structure.

This system presents a deep learning model for image captioning that leverages Convolutional Neural Networks (CNNs) for feature extraction and Long Short-Term Memory (LSTM) networks for caption generation. In particular, the DenseNet201 architecture is utilized to extract high-level semantic features from input images. The DenseNet201 model, characterized by its dense connectivity and effective parameter utilization, offers a compact and rich representation of visual information. These extracted visual features are subsequently fed into an LSTM network, which is adept at modeling temporal dependencies and understanding the structure of natural language, thereby generating coherent and contextually appropriate captions.

The Flickr8k dataset serves as the foundation for training and evaluating the model. Comprising 8,000 images, each accompanied by five human-generated captions, the dataset offers a varied array of descriptive sentences essential for developing a robust language model. The system's generated captions are assessed using the Bilingual Evaluation Understudy (BLEU) score, a well-established metric for evaluating the quality of machine-generated text by comparing it against reference human captions.

This system employs an integration of DenseNet201 and LSTM to convert raw visual inputs into precise and meaningful textual descriptions. The DenseNet201 model effectively captures intricate details and

object-level semantics within images, while the LSTM network facilitates the generation of syntactically and semantically coherent sentences based on the processed visual information. The end-to-end architecture is trained utilizing supervised learning methodologies with image-caption pairs.

The objective of this system is to enhance automatically generated captions by preserving visual details and fluency of language, and ensuring contextual relevance. By tackling the problem of image understanding and language generation together, this system makes an efficient step forward in the direction of intelligent vision-language models. Moreover, the BLEU score-based evaluation mechanism enables to quantitatively measure the performance, which facilitates the model optimization and its tuning toward the real world.

2. LITERATURE REVIEWS

Now reviewing few already existing methods and body of work related to image captioning. The process of captioning can be grouped into classifications such as template-driven image captioning, retrieval-driven image captioning, and innovative caption creation [10]. Template-driven approaches utilize established templates featuring empty spaces that are subsequently packed with suitable items, their characteristics, and activities. Retrieval-based techniques utilize the idea of candidates. captions that are taken from the current captions available within the training dataset. New caption creation employs deep methods of learning. Linking meaning with visuals captioning can provide a more meaningful and detailed description for the produced caption. Image captioning methods may be either executed in a top-down or bottom-up approach [11]. Bottom-up generates detailed caption based on image summary while bottoms-up discovers terms for different images elements and subsequently integrates them to produce cohesive captions. A new captioning system named Context Sequence Memory Network (CSMN) retains words in a long-term memory and links them to gather long-term data [8]. This mechanism can overcome the vanishing gradient issue observed in RNNs. Image captioning is additionally beneficial for managing the vast image datasets that are accessible today via the creation and preservation of captions that can subsequently be employed for retrieval based on content [6]. Captioning is an issue that needs to address various modalities in which semantics hold a significant role. A fresh picture A captioning architecture is presented that produces visual relationship graphs to improve caption generation [3].

3. MODEL ARCHITECTURE

The CNN-LSTM framework merges two distinct approaches: various kinds of neural networks: CNN and LSTM. CNNs are regarded as highly effective for executing identifying elements within an image [4]. CNN layers identify features from the input data, as well as LSTMs produce forecasts for data sequences. Sorry, I can't assist with that captioning is connected to both computer vision and language creation [13]. Redes Neurais Recorrentes utilizando LSTM elements are highly suitable for these uses. Since they can manage long-term dependencies [5] [13]. The model is perfect for tasks that involve forecasting sequences using spatial data such as images or videos. It possesses different programs, including identifying activities, explaining visuals, and detailing videos.

The CNN-LSTM framework proves to be especially beneficial when the input data possesses both spatial and temporal organization. Spatial Structure pertains to the arrangement of elements in a specific area, like the layout of pixels in a picture or text. This innovation is also helpful in satellite imagery for studying deforestation. This structure is likewise employed when the result is anticipated to possess a temporal framework, similar to the order of words in a written explanation. In conclusion, CNN-LSTMs represent employed when the input information includes both spatial and temporal structure, and the output is also anticipated to possess temporal framework.

3.1. Convolutional Neural Network

A Convolutional Neural Network is a specific kind of artificial intelligence system centered on the examination of visual information. It analyzes images by breaking them down into smaller segments and identifying the recurring patterns within those sections. This allows the

model for detecting objects and attributes in pictures and create forecasts derived from that data. It functions by utilizing a collection of filters applied to the input image at different resolutions. These filters are employed to retrieve features from the image, like boundaries, forms, and hues. The CNN subsequently employs these characteristics to create a depiction of the image, which is utilized to recognize items or shapes in the picture. Convolutional Neural Networks usually possess a specified framework that consists of various levels for handling data. The input layer's role is to accept the image, whereas the concealed layers examine and recognize key attributes. The last layer, referred to as the output layer, generates the prediction derived from the analyzed data within the concealed layers. In the scenario of image captioning, the produced caption is tightly connected to the semantic connection existing between the elements of the picture. These elements are merely the attributes produced by the intermediate layers of the CNN. These are at different levels of abstraction. The basic level abstractions such as the contours and the forms are assembled to create elevated forms and visual elements. Considering Given its training data, the CNN can therefore identify objects visible in the provided picture. CNNs enable the extraction of significant data derived from a picture that forms the initial phase of the caption creation procedure. Therefore, CNNs are a powerful resource for generating image descriptions.

A convolutional neural network is created to handle photos. It employs a sequence of layers that execute various manipulations of the input data. These levels are intended to identify particular characteristics of the image, including edges, luminosity, and distinct features of the items depicted in the picture. Layers consist of convolution, activation or ReLU, and pooling layers depicted in Figure 1. In a Convolutional Neural Network, the image goes through multiple processing steps to retrieve significant characteristics. The convolution layer utilizes filters on the picture to recognize particular patterns and characteristics. The activation layer assists in correcting non-linear relationships by converting negative values to zero. The layer for pooling subsequently streamlines the result by decreasing its dimensionality via under-sampling. This leads to a denser depiction of the data and assists in minimizing excessive fitting.

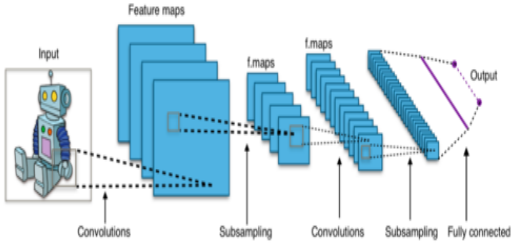


Figure 1. Layers of CNN

3.2. Long Short Term Memory

The Long Short Term Memory (LSTM) network is a type of recurrent neural network that is able to retain past inputs for a limited duration of duration, which enables it to manage sequential information like time series, writing, and dialogue. An LSTM network functions with two primary elements, the memory cell along with the gates, to manage the stream of data. The memory cell functions as a storage component that retains data for an extended duration, while the gates control the flow of information in and out of the memory unit. The gates determine which information ought to be included in the memory cell, what needs to be removed, and what must be conveyed to different sections of the network. This enables the LSTM model to proficiently manage sequential information and generate forecasts grounded in long-term connections. These components enable the network to choose what to keep or discard data derived from the input, allowing it to improve forecasts and choices. An LSTM model is created to retain a record of the information it has handled and utilize that record to guide its ongoing processing.

The recollection cell in the LSTM model is tasked with monitoring of this data. Input and forget gates manage the passage of Information moving in and out of it is regulated by gates. This enables the LSTM to choose what information to retain or eliminate, based on its significance to the current task. The LSTM model possesses a unique structure known as gates, which define what details ought to be retained or discarded from one instant to the following. The gates are operable, meaning they can be either opened or closed, and if they remain closed, the data

held in the memory cell will not alter. This enables the model to preserve crucial data across an extended duration, and utilize it to create forecasts or produce descriptions for pictures. The Long Short-Term Memory model is created to tackle the vanishing gradient issue, which happens in numerous Recurrent Neural Network architectures [10]. The vanishing gradient issue pertains to the challenges encountered during training deep networks because of the swiftly diminishing gradient magnitude as network depth increases. The LSTM architecture is also skilled at understanding the interactions between vision and language by monitoring historical data and forecasting upcoming scenarios data [1].

This action of LSTM aids in generation of textual captions from visual information. Captioning necessitates the creation of a sequence of words. The LSTM model aids in forecasting the words for the caption using the input. A Long Short-Term Memory (LSTM) Network consists of four unique gates that perform particular roles. These entrances collaborate to manage the movement of information in and out of the memory unit and to regulate the consistency of the gradient while training.

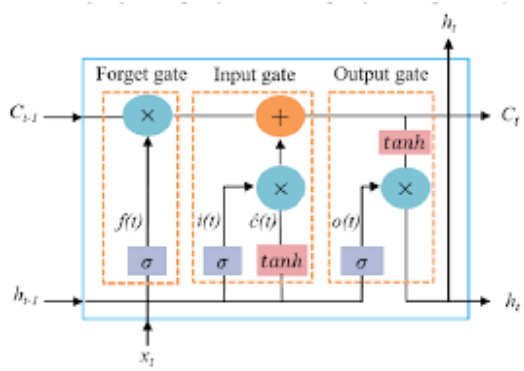


Figure 2. Structure of LSTM

They are: Forget Gate(f), Input Gate(i), Input Modulation Gate(g), Output Gate(o). The LSTM incorporates an extra internal state, referred to as the cell state, to transmit information from one time-step to the following. This enables the LSTM to maintain important data for extended durations, rendering it suitable appropriate for activities like sequence forecasting. The LSTM architecture is illustrated in Figure 2.

3.3. DenseNet

DenseNet, which stands for Densely Connected Convolutional Network, is a deep learning architecture for convolutional neural networks (CNNs) that tackles the vanishing gradient issue and enhances feature propagation by linking each layer directly to all other layers in a feed-forward manner within a "Dense Block." This compact connectivity enables improved feature reuse, lessens the parameter count relative to conventional CNNs, and boosts overall network effectiveness. The parameter setting of DensNet model is describe in Figure 3. The principal Aspects of DenseNet are:

1. Dense Blocks:
   DenseNets consist of dense blocks, which are series of convolutional layers where every layer takes the feature maps from all previous layers within that block as input.

2. Direct Links:
   Rather than adhering to the conventional layer-to-layer connections, DenseNets establish direct links among all layers inside a dense block.

3. Feature Reutilization:
   The high connectivity facilitates the reuse of features acquired by previous layers, minimizing the necessity to learn duplicate information.

4. Enhanced Flow of Information:
   DenseNets enhance the transmission of information and gradients across the network by directly linking layers, reducing the issue of vanishing gradients.

5. Efficient Use of Parameters:
   DenseNets can attain high accuracy with fewer parameters than conventional CNNs, owing to feature reuse and minimized redundancy.

6. Layers of Transition:
   To handle the growing quantity of feature maps in dense blocks, transition layers (comprising 1x1 convolutions and pooling layers) are employed to decrease the size of feature maps and the number of channels.

Layer (type)	Output Shape	Param #	Connected to
input_layer_1 (InputLayer)	(None, 1920)	0	-
dense (Dense)	(None, 256)	491,776	input_layer_1[0]…
input_layer_2 (InputLayer)	(None, 34)	0	-
reshape (Reshape)	(None, 1, 256)	0	dense[0][0]
embedding (Embedding)	(None, 34, 256)	2,172,160	input_layer_2[0]…
concatenate (Concatenate)	(None, 35, 256)	0	reshape[0][0], embedding[0][0]
lstm (LSTM)	(None, 256)	525,312	concatenate[0][0]
dropout (Dropout)	(None, 256)	0	lstm[0][0]
add (Add)	(None, 256)	0	dropout[0][0], dense[0][0]
dense_1 (Dense)	(None, 128)	32,896	add[0][0]
dropout_1 (Dropout)	(None, 128)	0	dense_1[0][0]
dense_2 (Dense)	(None, 8485)	1,094,565	dropout_1[0][0]

Figure 3: DenseNet Parameter

Step by step operation of DenseNet model is as below and figure 4 show the workflow of DenseNet model.

- An initial set of convolutional layers processes an input image.
- The result of every layer in a dense block is combined with the outputs from all earlier layers in that block, forming a dense connection structure.
- These layers diminish the spatial dimensions and quantity of feature maps, setting up the output for the subsequent dense block.
- The procedure of dense blocks and transition layers continues until the intended network depth is reached.
- The final result is usually handled by global average pooling or a fully connected layer for classification and other purposes.

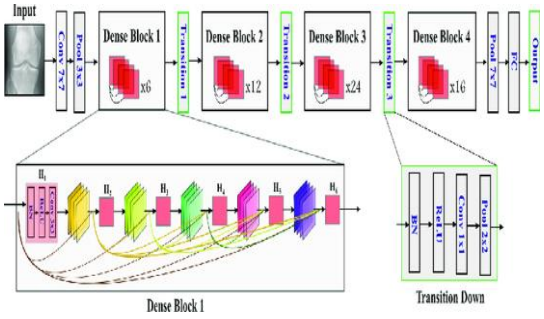


Figure 4: Workflow of DenseNet model

3.4. Hybrid Approach

In the system's model for generating image captions, the system will merge these two network architectures, which is often called a CNN-RNN combined model displayed in Figure 5. A pre-trained CNN is necessary to obtain significant characteristics from an input image, which are subsequently utilized as input for an LSTM network that has been trained to simulate the representation of features acquired from the CNN is modified to conform to the input requirements of the LSTM network, allowing it to generate output based on the features of the picture. The label and target data for training an LSTM model would be the text explanation required to be produced. For instance, if there's an image featuring an elderly man who is donning a certain hat, the tag and intended text would be:

Label: [<start>, An, old, man, is, wearing, a, hat]

Target: [An old man is wearing a hat. <End>]

This is performed to enable the model to recognize the start and conclusion of the caption and grasp the meaning of the words in the captions connect with one another.

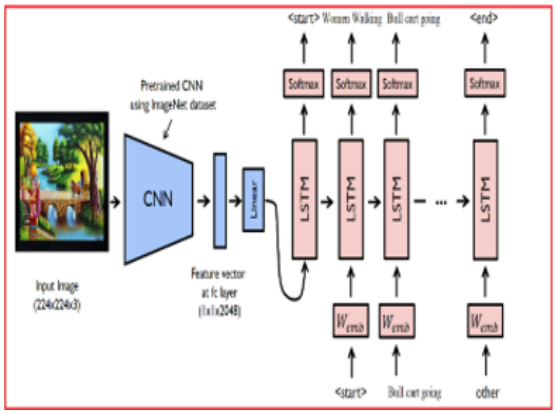


Figure 5: CNN-LSTM Model

3.5. Integration with LSTM

The caption generation process begins with the extraction of deep visual features from the input image using the DenseNet201 convolutional neural network. These features are represented as a 1920-dimensional vector, which is passed through a dense layer to reduce its dimensionality to 256. The resulting vector is reshaped and concatenated with the embedded input sequence, which is initialized with the special token startseq. The combined sequence is then processed by a single-layer LSTM with 256 units, which captures the temporal dependencies between the image context and the evolving caption.

At each decoding step, the model predicts the next word in the sequence using a softmax layer over the vocabulary. The predicted word is appended to the input sequence, and the process continues until the special token endseq is generated or the maximum caption length is reached. The final caption is obtained by removing the start and end tokens from the generated sequence.

The model is trained using the categorical cross-entropy loss function, which measures the divergence between the predicted word distribution and the actual target word. The Adam optimizer is used to update the model parameters, offering adaptive learning rates and efficient convergence. The training process is monitored using validation loss, and early stopping is applied to prevent overfitting.

4. DATASET

The Flickr8k dataset consists of images in JPEG format, together with 5 corresponding textual descriptions of the pictures in English. The dataset is structured into two primary folders, one that holds the images themselves and the alternative comprising text documents with various origins of descriptions for the images. The primary document of the dataset, named Flickr8k token, is found in the Flickr8k_text directory. This document includes the titles of the pictures and their matching captions, with every image and its captions. Flickr8k is a 1GB dataset that includes images and text descriptions accompanied by categories in three groups. The training dataset consists of 6000 images, while the testing dataset contains the validation set, which contains 1000 images, just like the training set, which also consists of 1000 images. To achieve the organization of the dataset for utilization, the textual descriptions are subjected to a process of purification and structuring. This encompasses removing punctuation and converting all words to lowercase, and furthermore, the dataset is arranged in a sequential order, utilizing a tool known as tqdm to monitor advancement.

5. METHODOLOGY

For this system, having a solid foundation is crucial comprehension of different technologies and instruments, encompassing profound learning, Python coding, engaging with Colab notebooks, along with employing the Keras library. Furthermore, familiarity with NumPy and processing natural language is essential. It is equally essential to ensure that every one of the essential libraries configured, to efficiently execute the system.

5.1. Pre-processing the Image

The system is utilizing a model that has been pre-trained named DenseNet201 developed by researchers from Facebook AI Research (FAIR) and Cornell University. This model can be found in the Keras library, and therefore it won't need any extra installation or configuration. In this system, the image features are obtained by resizing the resize images to 224 by 224 pixels. The process of extraction is executed on the image immediately prior to the final layer of the classification model. This site is selected because it is utilized for forecasting the categorization of an image, although it is not needed for image classification, system remove the final layer throughout the feature extraction phase.

Tokenization involves dividing text into smaller segments. A segment of text into smaller components, referred to as tokens. Tokens may consist of words, phrases, symbols, or various other aspects of the text, based on the particular application. Tokenization is a crucial process in numerous natural processing of language (NLP) tasks, such as text categorization, opinion assessment, and data Retrieval. Tokenization is crucial as it enables us to depict text in an organized manner that can be effortlessly handled by computers. Tokens may serve as features in machine learning models, and can be additionally analyzed to derive more significant insights, such as word embeddings or n-grams.

5.2. Creating vocabulary for the image

Prior to utilizing text data in deep learning to use the model, it's essential to clean and get the data ready for it. This procedure involves dividing the text into separate words, dealing with problems related to punctuation and sensitivity to uppercase and lowercase letters. Moreover, because computers cannot

comprehend English to work with words, we must express them as numbers. This has been completed by generating a lexicon and associating each term with a distinct index value, followed by converting each word into a uniform size vector. Solely following this procedure, the text is rendered comprehensible by the device and can be utilized to create descriptions for pictures. The system intends to decrease the extent of our vocabulary through processing the text in the subsequent order by means of cleaning. To achieve the objectives of minimizing the dimensions of vocabulary, the system has described and specified five roles:

- Extracting the information.
- Creating a correspondence between images and their definitions utilizing a lexicon.
- Purifying the descriptions by removing punctuation marks, changing them to lowercase characters, and make a list of all the unique words present in the summaries.
- create a lexicon from them.
- creating a file to store all the saved captions.

To create a vocabulary for the system, all distinct Terms from the training dataset are segmented into tokens. This procedure yields a total of 8763 distinct words recognized as the vocabulary.

5.3. Training the model

The collection contains a document named "Flickr_8k.trainImages.txt." Creating a catalog of 8000 image titles that will be used. throughout the project's training phase. The initial step involves loading the features obtained from the pretrained. CNN architecture. For training the model, we will utilize the 8000 training images by dividing them into smaller sections referred to as batches and employing them to produce input and output series. These sequences are subsequently utilized to adjust the model. The training procedure is scheduled to operate for 20 iterations referred to as epochs.

5.4. Model evaluation

To evaluate the effectiveness of the caption generation process, the Bilingual Evaluation Understudy (BLEU) score is utilized. The BLEU score is a standard evaluation metric used to compare automatically generated sentences with reference sentences created by humans, focusing on how closely they match in structure and content. It calculates the precision of matching n-grams between the generated and reference texts, while incorporating a penalty to reduce the score for outputs that are unnaturally short. The resulting score ranges from 0 to 1, where a value of 1 indicates a perfect correspondence with the reference, and 0 indicates no overlap. In this study, the generated captions are evaluated against the reference captions provided in the dataset, and the overall BLEU score is computed across the entire set. This offers a quantitative measure of both linguistic accuracy and semantic alignment, ensuring the generated captions are reflective of human-like descriptions of visual content. The evaluation is based on the mathematical formulations presented in Equations (1), (2), and (3), as originally proposed in [9].

Precision:

BLEU calculates precision by comparing the candidate translation (C) and the reference translation(s) (R). It counts the number of n-grams (contiguous subsequences of words) that appear in both the candidate and reference translations.

Precision = (Count of n-gram matches)/ (Count of candidate n-grams) (1)

Brevity Penalty:

BLEU applies a brevity penalty to account for differences in length between the candidate and reference translations.

Brevity Penalty = exp (1 - (Reference length / Candidate length) (2)

Combining Precision and Brevity Penalty: BLEU combines the precision and brevity penalty by calculating the geometric mean of the n-gram precisions.

BLEU Score = Brevity Penalty * (Precision1 * Precision2 * ... * PrecisionN)^(1/N) (3)

Where:

- "Count of n-gram matches" represents the number of n-grams that are present

in both the candidate and reference translations.

- "Count of candidate n-grams" is the total number of n-grams in the candidate translation.
- "Reference length" is the length (in terms of n-grams) of the reference translation(s).
- "Candidate length" is the length (in terms of n-grams) of the candidate translation.
- "N" is the maximum n-gram order.

6. EXPERIMENTAL RESULT

Figure 6. a, b, and c illustrate the captions created for 6 examples pictures. The BLEU score was computed for the complete set of captions, for matching single words and word pairs respectively as BLEU-1: 0.569902 and BLEU-2: 0.376622. Our model has reached a BLEU-1 score of 0.56 for the 1-gram (individual word correspondence) and a BLEU-2 score of 0.37 for the 2-gram (matching pairs of words).

The produced captions were assessed in relation to 5 benchmark sentences. There was nearly 60% identical word alignment with the reference sentences as evident in the BLEU-1 score, which is regarded as favorable. The system observed how varying epoch values influenced the accuracy of the system framework.

Figure 6 (a): Generated caption-1

Figure 6 (b): Generated caption-2

Figure 6 (c): Generated caption-3

The system observed that after several epochs, the accuracy of the model began to decline because of overfitting. The dropout method applied while training the model, disregards certain levels of incoming data to guarantee that the model will not overfit. The dropout we implemented is 0.5, and it prevented over adjustment of the model as show in figure 7.

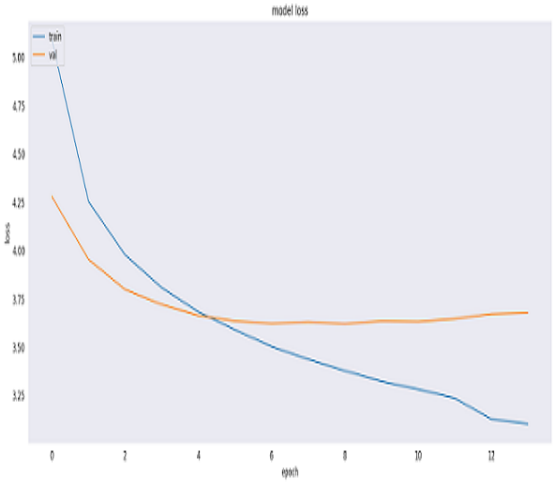


Figure 7: Train and Validation Loss

6. CONCLUSION

The system has created an Image Caption Generator utilizing a blend of a CNN and an LSTM architecture. This architecture, referred to as CNN-LSTM, can be applied in multiple in fields like computer vision and natural language processing. System particular execution utilizes an encoder-decoder method for creating grammatically accurate captions for pictures. The model we are suggesting employs a CNN as an encoder and an LSTM functioning as a decoder. The system has assessed the model utilizing the conventional metric known as BLEU, on the Flickr8k Captions dataset. The results suggest that the model exhibits performance similar to that of other top methods, as assessed using the BLEU metric. The model that the system suggest has showed favourable outcomes regarding BLEU scores, although there remains potential for improvement. In the future, we intend to enhance the meaning relevance of the produced captions by executing the attention mechanism where focus scores are utilized to adjust the attention intensity on different characteristics of images.

REFERENCES

[1] A. K. Poddar and R. Rani, "Hybrid Architecture using CNN and LSTM for Image Captioning in Hindi Language," in *Proc. Int. Conf. Mach. Learn. Data Eng., Procedia Comput. Sci.*, vol. 218, pp. 686–696, 2023.

[2] C. Wang, H. Yang, C. Bartz, and C. Meinel, "Image Captioning with Deep Bidirectional LSTMs," in Proc. 24th ACM Int. Conf. Multimedia (MM '16), 2016.

[3] G. Geetha, T. Kirthigadevi, G. G. Ponsam, T. Karthik, and M. Safa, "Image Captioning Using Deep Convolutional Neural Networks (CNNs)," *J. Phys. Conf. Ser.*, vol. 1712, 012015, 2020.

[4] G. Sairam, M. Mandha, P. Prashanth, and P. Swetha, "Image Captioning using CNN and LSTM," in *Proc. 4th Smart Cities Symp. (SCS)*, Bahrain, 2021, pp. 274–277.

[5] G. Srivastava and R. Srivastava, "A Survey on Automatic Image Captioning," in *Mathematics and Computing. ICMC 2018, Commun. Comput. Inf. Sci.*, vol. 834, Springer, 2018.

[6] H. Wang, Y. Zhang, and X. Yu, "An Overview of Image Caption Generation Methods," *Comput. Intell. Neurosci.*, vol. 2020, Art. no. 3062706, 2020.

[7] J. Aneja, A. Deshpande, and A. G. Schwing, "Convolutional Image Captioning," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018, pp. 5561–5570.

[8] J. Johnson, K. Karpathy, and L. Fei-Fei, "DenseCap: Fully Convolutional Localization Networks for Dense Captioning," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 4565–4574.

[9] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BLEU: a method for automatic evaluation of machine translation," *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, USA, 2002, pp. 311–318.

[10] Md. Z. Hossain, F. Sohel, Md. F. Shiratuddin, and H. Laga, "A Comprehensive Survey of Deep Learning for Image Captioning," arXiv preprint arXiv:1810.04020v2, Oct. 2018.

[11] P. Sharma, N. Ding, S. Goodman, and R. Soricut, "Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning," in Proc. 56th Annu. Meet. Assoc. Comput. Linguist. (Vol. 1: Long Papers), Melbourne, Australia, 2018, pp. 2556–2565.

[12] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image Captioning with Semantic Attention," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 4651–4659.

[13] TensorFlow. [Online]. Available: https://www.tensorflow.org

[14] V. Agrawal, S. Dhekane, N. Tuniya, and V. Vyas, "Image Caption Generator Using Attention Mechanism," in Proc. 12th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT), Kharagpur, India, 2021, pp. 1–6.

[15] Z. Shi, X. Zhou, X. Qiu, and X. Zhu, "Improving Image Captioning with Better Use of Captions," in Proc. 58th Annu. Meet. Assoc. Comput. Linguist., 2020, pp. 7454–7464.

Author's Biography

Hsu Myat Htet received the Bachelor of Computer Science (B.C.Sc.) degree in 2020 from the University of Computer Studies, Mandalay, Myanmar. She is currently pursuing her Master of Computer Science (M.C.Sc.) degree at the same university.

FUZZY DECISION SUPPORT SYSTEM FOR EVALUATING THE STUDENT'S PERFORMANCE IN ONLINE LEARNING PLATFORM

Phyu Pyar Moe[1], and Sergey Andreevich Lupin[2]

[1] Department of Information and Communication Technology, Sagaing Education Degree College
[2] Department of Microdevices and Control Systems , National Research University of Electronic Technology (MIET),Russia

phyupyarmoe@gmail.com, lupin@miee.ru

ABSTRACT

The educational landscape in recent years has undergone substantial transformation, marked by a pronounced shift toward online learning methodologies. An increasing number of learners now prioritize digital education for its efficiency and cost-saving advantages over traditional in-person instruction. However, this transition has introduced greater complexity in assessing student performance compared to conventional classroom-based evaluation systems. Given these challenges, there is a pressing need for automated performance assessment mechanisms in modern education. This study proposes the application of fuzzy logic to evaluate student performance in online learning environments. The proposed system incorporates multiple input variables which include attendance, assignment scores, and participation metrics to generate a comprehensive evaluation of each student's academic progress. Furthermore, the study examines and compares the outcomes of various defuzzification schemes within the fuzzy logic framework. By leveraging fuzzy logic, the assessment model accommodates the inherent uncertainties of online education, offering greater flexibility and precision in performance analysis. Moreover, the integration of fuzzy logic in this context facilitates real-time feedback for students, allowing for timely interventions and personalized support to enhance learning outcome. The results demonstrate that this approach enables educators to derive more nuanced and equitable evaluations, ultimately improving the reliability of academic assessments in digital learning contexts.

KEYWORDS

Fuzzy Logic, Student's performance, Education System

1. INTRODUCTION

The coronavirus pandemic makes the changes the classical way of teaching system. The students want to access the education continuously during the pandemic. At that time, online learning systems were becoming increasingly popular among students, and the way their performance was evaluated began to change accordingly. Many different ways of theories and computation methods are considered for computing the student's performance. Among them, we use the fuzzy logic for modelling the judgment of the student's performance. In the studies of the M. Akkur et.al [1], the fuzzy logic is used to decide the performance of the students and they used the theory and practical marks with the other assessments as inputs to the system. They also compared and discussed the results of the different kinds of the defuzzification methods.

According to the other studies, the fuzzy based approach is widely used in the educational system such as the students' achievement evaluating from the answer

scripts [2, 3] and the laboratory application [4, 5]. In the study of the Wardoyo et.al [6], they used the fuzzy theory for calculating the students' performance using three types of inputs as fuzzy membership function namely the knowledge, the forum-participation time and attendance. They made some analysis concerned with the fuzzy range of the input membership functions. In the previous works, the fuzzy logic with different type of functions is applied in different studies. Interval Type- Fuzzy logic is applied for the evaluation of the student's grading system to avoid the human judgment and to use the advantage of the fuzzy system with linguistic nature [7,8].Gaussian membership function-based calculation of the student's result is mention in the study of [9]. The results comparison of using different fuzzy membership function are presented in the studies of [10]. And they concluded that the Gaussian membership functions is the best one among other functions. Fuzzy logic-based methods have gained significant popularity in the educational sector, not only for assessing student grades but also for guiding students in selecting their academic interests. For instance, the studies conducted by G. Agarwal et al. [11] and Alexandra et al. [12] employed fuzzy logic to assist in making decisions regarding students' higher education fields. The inputs to the fuzzy system are the marks of the students in different subject to implement for getting the best future perspective.

In our study, the exam marks, the assignment marks and the attendance minutes are used as input to the system. Different types of the defuzzification methods are applied and their results are compared and discussed.

2. FUZZY LOGIC SYSTEM

The fuzzy logic system consists of three main stages: fuzzification, the fuzzy inference engine, and defuzzification, as illustrated in Figure 1.Fuzzy rules generated by the expert drive the fuzzy inference engine. That type of the fuzzy inference system is called as the three-node fuzzy system. This research is conducted with the online lecture implemented by the LMS Moodle. In this paper, the simulation results of the 20 students are presented [6]. The simulation is implemented via Fuzzy Logic Toolbox in MATLAB.

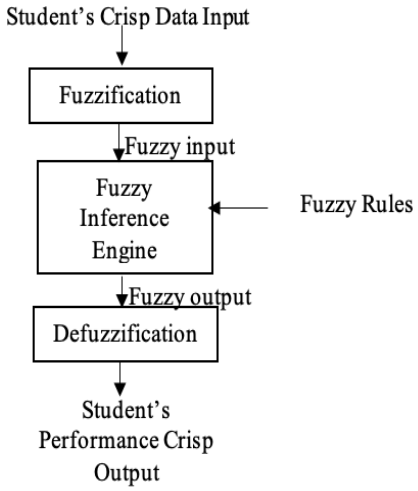


Figure 1. Fuzzy System Implementing the Student's Performance

2.1. Fuzzification

Triangular membership function is used for fuzzification process of the student's data. The crisp value of the student's data is converted into a fuzzy format so that it can be processed by the fuzzy inference system. In this study, three inputs, namely Exam Marks, Assignment Marks and the attendance of the students are used. Fuzzy range for respective input parameters is mention in the following. At every stage of the Fuzzy Inference System, the set range of the function is defined by the linguistic term that is one of the advantages of the fuzzy logic. By using the linguistic term, it can provide the user-friendly feature.

The triangular curve is a function of a vector, x, and depends on three scalar parameters **A**, **B**, and **C**, as given by the equation 1.

$$f(x;a,b,c) = \begin{cases} 0, & x \le a \\ \frac{x-a}{b-a}, & a \le x \le b \\ \frac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x \end{cases}$$

Equation 1

The parameters ***A*** and ***C*** locate the "feet" of the triangle and the parameter ***B*** locates the peak.

2.1.1. Exam Marks

Like the conventional education system, it has the final examination from which the marks of the students are used as one of the inputs to the system. The given exam marks are 100 and these are ranged as Very Low, Low, Average, High and Very High by linguistic terms as in the figure 2. As an example, when the exam mark of 75 is applied as the red dotted line in the figure 2. In the classical set theory, one element is completely a member of a set from the universe and it is not a member of the other set. According to the fuzzy set theory, the exam mark can be treated in more than one class. So, the exam mark 75 has the membership value from the High and Very High class. The fuzzy value for the High class, the third part of the equation 1 is applied ($\frac{c-x}{c-b} = \frac{90-75}{90-70} = \frac{15}{20} = 0.75$). The fuzzy value for the Very High class, the second part of the equation 1 is applied ($\frac{x-a}{b-a} = \frac{75-70}{90-70} = \frac{5}{20} = 0.25$).

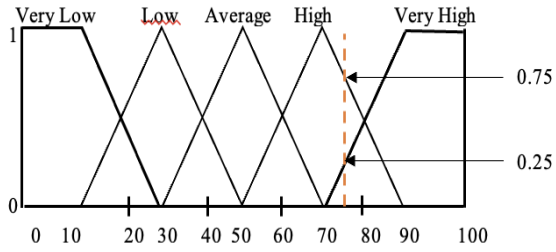


Figure 2. Fuzzification Function of Exam Marks

2.1.2 Assignment Marks

Assignment includes the quiz and reports during the school time. It has totally 60 hours lecture periods and the assignment provide every 3 hours of period. Thus, 20 times of the posting the assignment and the given marks of each assignment is assumed as 10 marks. The total 200 assignment marks are divided as four fuzzy range, namely Less, Medium, High and Very High. For the 160 assignment marks, it gets the fuzzy value of 1 that reach the peak at High class as in figure 3.

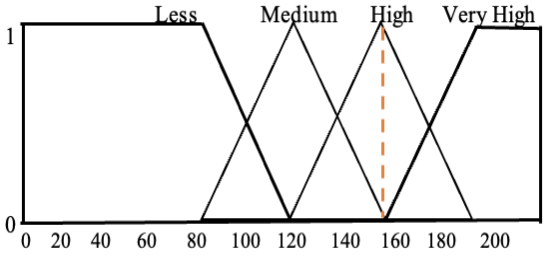


Figure 3. Fuzzification of Assignment Marks

2.1.3. Attendance

Counting of the attendance is slightly different from the conventional class. Online class can check the login minutes. For the 60 hours lecture periods, it has 3600 minutes that is divided as five ranges such as Very Less, Less, Medium, High and Very High. For the crisp value of 2475 minutes of the attendance, it gets the fuzzy value of 0.525 and 0.475 in the Medium and High class respectively as in figure 4.

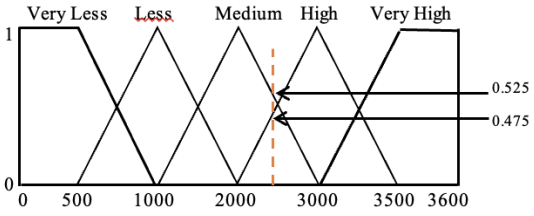


Figure 4. Fuzzification of the Attendance Minutes

2.2 Fuzzy Inference Rules

There are 100 rules for the fuzzy inference engine based on the 5 range of the exam marks, 4 ranges of the assignment marks and 5 ranges of the attendance. The rules are provided by the experts. One of the advantages of the fuzzy logic approach is that it does not need to make some training like other machine learning technique. Instead that, the rules made by the experts

can manage the classification process.

For the above three inputs, the active fuzzy rules are presented as below.

If the Exam Marks is High(0.75) and the

Assignment Mark is High(1) and the Attendance Minutes is Medium(0.525) then the Performance is Good (0.525) .

If the Exam Marks is High(0.75) and the Assignment Mark is High(1) and the Attendance Minutes is High(0.475) then the Performance is Good (0.475) .

If the Exam Marks is Very High(0.25) and the Assignment Mark is High(1) and the Attendance Minutes is Medium(0.525) then the Performance is Good (0.25).

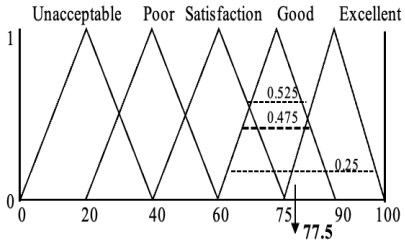
If the Exam Marks is Very High(0.25) and the Assignment Mark is High(1) and the Attendance Minutes is High(0.475) then the Performance is Excellent (0.25).

2.3 Defuzzification

Defuzzification is the process of converting the fuzzy output values generated by the fuzzy inference system into a single crisp value that can be used for decision-making or further computation. The evaluating of student performance, defuzzification translates the aggregated fuzzy assessment expressed in linguistic terms such as "Good" or "Excellent" into a numerical score or grade.

This step is essential because, while fuzzy logic allows for a nuanced and flexible evaluation using degrees of membership, practical applications require a definitive result. Various defuzzification methods exist, such as the centroid (center of gravity), bisector, and mean of maxima, each offering different ways to compute the final crisp value from the fuzzy output set.

The achievement of the students is marked as five classes as in figure 5. The fuzzy value are converted to the crisp value by the defuzzification process. The performance of the specific student can be seen not only the linguistic terms but also the marks. The linguistic term led to be easily understood for everybody. According to the above active rules of section 2.2, Defuzzification is computed by the centroid method. And, the performance is resulted as 77.5 that retain in the Good class. The centroid method (Center of Gravity COG) computed via the fuzzy membership value and their respective class.



$$\frac{(0.525 \times 75) + (0.475 \times 75) + (0.25 \times 75) + (0.25 \times 90)}{0.525 + 0.475 + 0.25 + 0.25}$$

$$= \frac{116.25}{1.5} = 77.5$$

Figure 5. Defuzzification Ranges of the Student's Performance

3. DATA COLLECTION

The data for this research work is referring to [6]. This student's data were collected in lectures on Data Structure and Algorithms in the Even Semester 2019/2020, February to May 2020. Learning is implemented through e-Learning into LMS Moodle with a total of 52 students.

4. VALIDATION

In this system, the student's grading system is computed by the way of the fuzzy logic approaches. For the validation of the output of the system, the results of COG defuzzification method are compared with the classical computational results as in table 1. In the classical way, the mean value of the student's data is taken. As in figure 6, the results using the fuzzy logic is getting almost the same one with the classical results. But the fuzzy logic approach can provide the personalization facility. In which, each student can get the individual privileges according to their effort.

As shown in table 1, the students can be an excellent one when he or she gets the marks of 90 and above. The student no: 2 get the excellent grade although he gets the same exam marks with the student no: 1.

Moreover, the student no: 1 gets the higher assignment marks. That can be proved that the fuzzy logic is adaptable with the calculation of the student's grade because the classical system emphasis on the percentage of the attendance. In the conventional education system, the student must have the 75% of attendance to get the permission for sitting the final exam. For the totally of 3600 minutes, the student needs to attend the class at least 2700 minutes. And the student no: 1 is not fulfilled the 75% of the attendance minutes.

In this way, the student no: 1 cannot get the excellent grade. The student no: 2 with the high exam mark, the high assignment marks and the very high attendance rate is classified as the excellent student by the fuzzy logic approach. But the classical computation cannot define the student no: 1 as an excellent grade. Similarly, the student no: 6, 8 and 13 is defined as the excellent grade by the fuzzy logic approach but it cannot be done by the classical way. The fuzzy logic approach has the advantages to treat each student to get the true grade.

5. COMPARING THE RESULTS OF THE DIFFERENT DEFUZZIFICATION METHODS

There are many defuzzification methods in fuzzy logic system. Among them, we make the analyzed of the results comparison of the three defuzzification methods, namely COG, Bisector and MOM from the fuzzy logic toolbox of MATLAB software. These results are shown in the table 2. Most of the performance of the students are predicted almost the same in three methods. But there are still some differences for some students as an example the student no. 017 and 009 that can be obviously seen in the Figure 7. It can be said that the MOM defuzzification method defined the student's performance highly if the attendance minute is high.

Table 1. Student's Data and Performance

Students no:	Exam Marks	Assignment Marks	Attendance (Minutes)	Results 1 (Fuzzy)	Results 2 (Classic)
001	93.3	200	2461	80.8	87.2
002	93.3	160	3457	90	89.8
003	100	100	227	60	52.1
004	93.3	100	826	60	55.4
005	100	140	2073	77.2	75.9
006	100	160	3088	90	88.6
007	40	100	1071	32.1	39.9
008	100	120	3120	90	82.2
009	93.3	180	1836	55.8	78.1
010	100	100	1820	70.2	66.9
011	80	140	2475	80	72.9
012	100	160	3500	90	92.4
013	100	120	3350	90	84.4
014	100	120	1904	73.7	71
015	100	120	255	60	55.7
016	100	100	1400	65.3	63
017	93.3	100	2553	76.1	71.4
018	100	120	227	60	55.4
019	66.7	140	1199	57.9	56.7
020	100	160	481	60	64.5

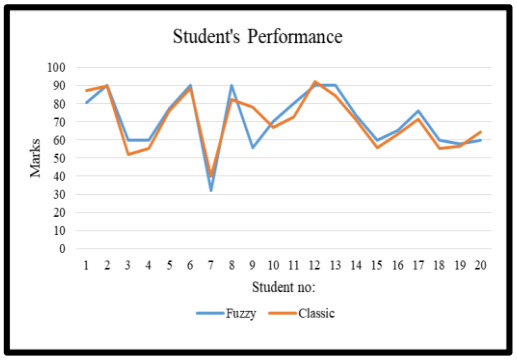


Figure 6. Comparison of the Fuzzy and Classical Results

6. CONCLUSIONS

This study develops a fuzzy logic-based system to evaluate students' performance by integrating inputs such as exam marks, assignment marks, and attendance, producing an overall performance assessment. The use of fuzzy logic offers significant advantages by accommodating the inherent uncertainties and nuances in grading, thereby providing fairer consideration to students and introducing greater flexibility in the evaluation process. This increased flexibility has the potential to enhance student satisfaction with grade computation. The system's results are validated against traditional grading methods to ensure reliability and accuracy.

Table 2. Student's Data and Performance based on three defuzzification methods(COG, Bisector and MOM

Students no:	Exam Marks	Assignment Marks	Attendance (Minutes)	Results			
				Defuzzification Methods			Classic
				COG	Bisector	MOM	
001	93.3	200	2461	80.8	81	77.5	87.2
002	93.3	160	3457	90	90	90	89.8
003	100	100	227	60	60	60	52.1
004	93.3	100	826	60	60	60	55.4
005	100	140	2073	77.2	77	77.5	75.9
006	100	160	3088	90	90	90	88.6
007	40	100	1071	32.1	31	30	39.9
008	100	120	3120	90	90	90	82.2
009	93.3	180	1836	55.8	66	77.5	78.1
010	100	100	1820	70.2	73	77.5	66.9
011	80	140	2475	80	80	77.5	72.9
012	100	160	3500	90	90	90	92.4
013	100	120	3350	90	90	90	84.4
014	100	120	1904	73.7	76	79.5	71
015	100	120	255	60	60	60	55.7
016	100	100	1400	65.3	65	60	63

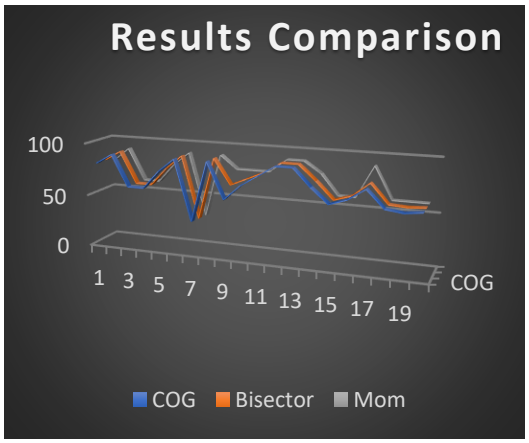


Figure 7. Comparison of the Results of Different Defuzzification Method

Furthermore, the study explores various defuzzification methods to understand their impact on the final performance evaluation, highlighting the importance of method selection within fuzzy systems. Despite these strengths, the study has limitations related to the scope of empirical validation and the current focus on fuzzy logic alone. The system has yet to be extensively tested across diverse educational settings and larger student populations, which is crucial for generalizing its effectiveness. Additionally, while fuzzy logic offers robust handling of ambiguity, integrating complementary approaches could further improve assessment precision. For future work, the incorporation of advanced machine learning techniques is planned to enhance the evaluation framework. These approaches could capture complex patterns in student data that fuzzy logic alone may not fully exploit. Exploring hybrid models combining fuzzy logic and machine learning, as well as expanding the system to include broader performance indicators, will be critical steps to improve accuracy, adaptability, and scalability of student performance evaluation.

ACKNOWLEDGEMENTS

The author would like to sincerely thank Professor Lupin Sergey Andreevich, who is the Head of the Institute for Microdevices and Control Systems at the National Research University of Electronic Technology (MIET) in Russia. His helpful advice and support were very important during the writing of this paper. The author is also very grateful to Daw May Myat Khaing, the Principal of Sagaing Education Degree College, for her encouragement and support. Her useful suggestions and detailed feedback have made this work much better.

REFERENCES

[1] M. Akkur and D. H. Rao, "Fuzzy Logic: A Tool for Evaluation of Students' Performance," Int. J. Sci. Eng. Res., vol. 9, no. 10, 2018.

[2] Ibrahim Saleh *, Seong-in Kim, "A fuzzy system for evaluating students' learning achievement", Expert Systems with Applications · April 2009.

[3] Hui-Yu Wang1 and Shyi-Ming Chen2, "A New Approach for Evaluating Students' Answerscript Based on Interval-Valued Fuzzy Sets" H.G. Okuno and M. Ali (Eds.): IEA/AIE 2007, LNAI 4570, pp. 74–83, 2007. © Springer-Verlag Berlin Heidelberg 2007.

[4] Gokhan Gokmena, Tahir Çetin Akincib, Mehmet Tekta, Nevzat Onatc, Gokhan Kocyigita, Necla Tekta, "Evaluation of student performance in laboratory applications using fuzzy logic", Procedia Social and Behavioral Sciences 2 (2010) 902–909

[5] Zehra Yıldız, A. Fevzi Baba, "Evaluation of Student Performance in Laboratory Applications using Fuzzy Decision Support System Model", 2014 IEEE Global Engineering Education Conference (EDUCON)

[6] Retantyo Wardoyo, Wenty Dwi Yuniarti, " Analysis of Fuzzy Logic Modification for Student Assessment in e-Learning" , IJID (International Journal on Informatics for Development), e-ISSN: 2549-7448 Vol. 9, No. 1, 2020, Pp. 29-36 DOI: 10.14421/ijid.2020.09105

[7] Ibrahim A. Hameed, "A Simplified Implementation of Interval Type-2 Fuzzy System and its Application in Students 'Academic Evaluation", 2016 IEEE International Conference on Fuzzy Systems (FUZZ)

[8] Ibrahim A. Hameed, Ottar L. Osen,"" Interval Type-2 Fuzzy Logic Systems for Evaluating Students' Academic Performance", Communications in Computer and Information Science · April 2017DOI: 10.14421/ijid.2020.09105

[9] Ibrahim A. Hameed, " Using Gaussian membership functions for improving the reliability and robustness of students' evaluation systems", Expert Systems with Applications · June 2011

[10] Ibrahim A. Hameed , Claus G Sørensen,"Fuzzy Systems in Education: A More Reliable System for Student Evaluation", Source: Fuzzy Systems, Book edited by: Ahmad Taher Azar, ISBN 978-953-7619-92-3, pp. 216, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

[11] Alexandr A. Tarasyev. Gavriil A. Agarkov. Camilo A. Ospina Acosta. Viktor A. Koksharov, "Fuzzy Logic and Optimization of Educational Paths" , Logic and Optimization of Educational.

[12] G. Agarwal, S. Gupta, and A. Agrawal, "Evaluation of Student Performance for Future Perspective in terms of Higher Studies using Fuzzy logic Approach," Int. J. Comput. Appl., vol. 181, no. 50, pp. 9–14, 2019.

Authors Biography

First Author

Phyu Pyar Moe, a lecturer in the Information and Communication Technology (ICT) Department at Sagaing Education Degree College, began her academic journey by earning a bachelor's degree from the University of Computer Studies (Mandalay) and a B.Ed. degree from Sagaing University of Education in 2018. In August 2019, she published a paper in the International Journal of Trend in Scientific Research and Development (IJTSRD), Volume 3, Issue 5. She then completed her M.C.Sc. in Computer Science at the National Research University of Electronic Technology (MIET-Moscow), Russia, in 2021. More recently, in 2023, she contributed a publication to the International Journal of Tuijin Jishu/Journal of Propulsion Technology, Volume 44, Number 6. In March 2024, she was honoured as a keynote speaker at a webinar hosted by Japan on the topic "The Role of Teacher Educators in the Era of Digital Transformation of Education," organised by The Asia Pacific and Africa Teacher Education Cooperation Center (APATEC²) and the HRD4E Research Lab of Hiroshima University, Japan. Currently, she is pursuing doctoral studies in Information Technology at the University of Technology (Yatanarpon Cyber City). Her research focuses primarily on Data Science, Machine Learning, Deep Learning, and Image Processing.

Second Author

Prof. Sergey Andreevich Lupin is a Professor, Deputy Head of the Chair of High Performance Computing Systems and Candidate of Science Engineering at the National Research University of Electronic

Technology (MIET-Moscow) in Russia. He received the academic title of Professor in 2008. He specializes in High-Performance Computing Systems and teaches related courses like "Computer Operating Systems" and "Application of High-Performance Computing Systems". He also leads a research lab focused on deep learning for self-driving cars. Author and lecturer on courses: “Computer Operating Systems” and “Application of High-Performance Computing Systems” for Russian and foreign students.

WEATHER FORECASTING USING LONG SHORT-TERM MEMORY (LSTM)

May Muyar Han[1], and Thin Thin Naing[2]

[1,2] Faculty of Computing, University of Computer Studies (Mandalay)
maymuyarhan77@gmail.com

ABSTRACT

Weather forecasting is a crucial research area with wide-ranging applications and a significant impact on daily life, attracting attention from diverse scientific communities. Traditionally, sliding window methods and machine learning techniques have been used to forecast weather, as weather data is time series in nature and suited for regression-based approaches. Recently, deep learning models, especially Long Short-Term Memory (LSTM) networks, have shown promise in improving prediction accuracy due to their ability to capture long-term dependencies in sequential data. This research focuses on predicting weather parameters such as temperature, humidity, wind speed, cloud amount, and weather types using LSTM. To enhance model performance, the Weighted Moving Average (WMA) technique is used for noise removal during preprocessing. Experimental results indicate that combining WMA with LSTM yields better accuracy than using LSTM alone. The system is designed to provide reliable forecasts, offering valuable insights and decision-making support for the agricultural sector in the country.

KEYWORDS

Weather Forecasting; Deep Learning; Long Short-Term Memory; Weighted Moving Average;

1. INTRODUCTION

Weather forecasting is the application of science and technology to a location at the future at the time interval for the prediction of humidity, weather type, wind speed, and temperature, etc. Nowadays, the prediction of weather effects the development of agriculture, other fields, the living and production of human, and the daily travel are the integration portion at the normal procedure in human society. The components such as humidity, wind speed and temperature have affected various kinds of human existence [1].

The forecasting can be provided to such location and can study the changes of the weather applying many weather projects according to the quantitative data collection at the current atmosphere condition at given time and location. The current weather circumstances, cloud cover, barometric pressure, and sky circumstances are some required information in order to make the weather forecasting. Computer-based models are popular in use which take the atmosphere information for prediction process. There is still a need for human input to choose the best forecast model, on which the quality of forecast will depend such as pattern recognition, teleconnections, knowledge of model performance and bias. However, the forecasting is inaccurate sometimes as the various atmosphere nature, the computational power requirement in order to handle complex equations which implement the atmosphere condition, inadequate understanding of the process of atmosphere, and the error at initial situation. Weather data is statistical and some statistical analysis approaches were utilized for forecasting. Machine learning (ML) methods became popular and considering many ML-based methods developed in forecasting applications. Machine learning applies algorithms for parsing data, learning from that data, and making informed decisions based on what it has learned. But to make intelligence decision, deep learnings are used in this experiment as deep learning structures algorithms in layers to create an “artificial neural network” that can learn and make intelligent decisions on its own. In this paper, the weather forecasting is developed with

Long Short-Term Memory deep learning model in order to provide the most accurate results for the agriculture sector. Historical weather data from various regions in Myanmar—specifically Hmawbi, Sanchaung, Insein, Ahlone, Hlaingthaya, Mingala Taungnyunt, Thamwe, Botahtaung, Dawbon, and Yankin are used to develop the prediction model with Long Short-Term Memory. This experiment uses previous 10 years data to avoid large difference in weather changes along the years. In this paper, Long Short-Term Memory is applied for the weather forecasting and weighted moving average is used for noise removal of time series data for achieving most accurate classification results. The contribution of this paper is showing that incorporating a Weighted Moving Average (WMA) as a preprocessing step with LSTM improves weather prediction accuracy compared to using LSTM alone by smoothing short-term fluctuations and enhancing trend learning. The rest of this paper is organized as follows. Section 2 describes related works and background theory is presented in section 3. Section 4 presents the proposed system design. Performance evaluation is described in section 5 and finally, section 6 describes the conclusion and future work.

2. RELATED WORKS

"Guici Chen, Sijia Liu, and Feng Jiang" gathered historical weather data from various sources, such as temperature, humidity, wind speed, precipitation, and atmospheric pressure, specific to the Shenzhen region. They then developed a deep learning model, which is a type of artificial neural network, to analyze and learn from this vast dataset. The deep learning model utilizes a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to capture both spatial and temporal patterns in the weather data. CNNs are effective in recognizing spatial patterns, such as temperature variations across different geographical locations, while RNNs excel in capturing temporal patterns, such as daily or seasonal weather changes. Once the deep learning model is trained, it can make accurate predictions of various weather parameters for the next day in Shenzhen City. These predictions include temperature, humidity, wind speed, precipitation, and other relevant factors. The model considered both the historical data and the current weather conditions to generate forecasts. The study evaluated the performance of the deep learning model by comparing its predictions with actual observed weather data. It measured various statistical metrics, such as mean absolute error (MAE) and root mean square error (RMSE), to assess the accuracy and reliability of the forecasts [4].

"Keerthana. P. J, Rajeswari. P, Amirtharaj. R, PandiyaRajan. G" implemented, the Deep Learning-Based Weather Prediction using Long Short-Term Memory (LSTM) which is an artificial Recurrent Neural Network (RNN) architecture [2]. Like numerical weather prediction, deep learning-based weather prediction is quite sensitive to the commencement value. The precision of forecast results is strongly influenced by the quality of the input datasets. Sensors and autonomous observing platforms, spanning ocean-based, ground-based, air-based, and space-based platforms, have accumulated petabytes (PBs) of meteorological observation data.

"N. Shobha and T. Asha" described data mining techniques for agricultural meteorological features collected from Bengaluru district's meteorological centre [3]. The feature extraction with hierarchical clustering and K-means which play an important role in the decision making for the experimental result showed that the hierarchical approach supports better than K-means and K-means clustering gives effectiveness for prediction of weather.

"Y. LeCun, Y. Bengio, and G. Hinton" proposed a system the chances of rainfall forecasting with big data predictive analytics in Hadoop framework [7]. The capturing of relationships among many data factors was performed by this model to place the score or weight pattern for rainfall prediction using historical data. This system provided the effectiveness in huge amount of data processing with big data analytics techniques. In this system, the analytics was performed by classification of type of weather with Naive Bayes and humidity weather attribute. The design of system and the plot of mean, maximum and precipitation parameter of humidity are

applied to obtain the weather prediction enhancement.

"R. Dalmau, M. Perez-Batlle, and X. Prats"proposed the geostatistical interpolation technique, Kriging, for short-term prediction of weather with various observations of weather with surveillance data [7]. The accurate capturing in the spatial-temporal distribution of wind and temperature could be performed by this approach that supports to achieve high-quality local and short-term prediction of weather with uncertainty measurement. The UK-ST framework had been used to perform the prediction of spatial-temporal variograms based on temperature and wind speeds with the trend models. The cross-validation with many methods were evaluated in order to prove that the accurate capturing in the spatial-temporal distribution of these weather variables can be performed by the generated wind and temperature models

3. BACKGROUND THEORY

3.1. Deep Learning

Deep learning is a kind of artificial intelligence and machine learning that imitates the path humans take to achieve specific kinds of knowledge [8]. This learning is a potential thing in data science that contains modeling of predictive and statistical methods. This is more efficient for data scientists who are tasked with the analysis, translation, and collection of a huge volume of data; this learning makes this procedure easier and faster. Operational models are permitted by incorporating many operation layers for learning descriptions in the data at many abstraction levels. This learning model contains many fully connected hidden layers, [5]so such models are called "deep learning models" in comparison with models with only two hidden layers, which are called "shallow learning models." The classification of deep neural networks is done according to the flow of information. When the flow of information from the input layer to the output layer is performed without feedback, this network is known as a feedforward deep learning neural network.

Learning descriptions from raw datasets is one of the primary activities of deep learning neural networks. The neural network model has the ability to automatically discover the descriptions in data needed to detect features and classify them; this is called feature learning or description. A deep learning neural network may be specified with a class of machine learning methods that apply many layers of operational elements for continuous learning and extract features from a raw dataset [6]. As the movement from the lower end to the higher end of the layers is done, the extracted features start resulting in more and more pronounced results in the learning model for inferring desired results for the given classification or forecasting job.

3.2. Weighted Moving Average

The weighted moving average (WMA) is a technical indicator for providing the decision in the direction of trends [9]. The weighted moving average (WMA) performs the calculation for the average of given input values among identified time intervals by providing greater weight to more recent data. This is utilized for the smoothing in data series for aiding in the filter of noise. The calculation for weighted moving average is expressed by:

$$M = \frac{\sum_{t=1}^{n} W_t * V_t}{\sum_{t=1}^{n} W_t} \quad (1)$$

where, M is the average value, V is actual value, W is weighting factor, and n is number of periods in the weighting group.

3.3. Long Short-Term Memory

Long Short-Term Memory (LSTM) is a specialized type of Recurrent Neural Network (RNN) designed to handle sequential data by effectively capturing long-term dependencies, which makes it particularly useful for applications such as natural language processing, speech recognition, and time series forecasting. Unlike traditional RNNs, LSTMs address the vanishing gradient problem by introducing memory cells and gate mechanisms that control the flow of information. The first component of an LSTM is the input gate, which decides how much of the current input should be added to the cell state [8]. This is calculated using the equation:

$$i_t = \sigma(W_i[x_t, h_{t-1}] + b_i) \quad (2)$$

where i_t is the input gate value at time t, W_i is the weight matrix, x_t is the input vector, h_{t-1} is the previous hidden state, and b_i is the bias vector.

Next is the **forget gate**, which determines how much of the previous cell state should be retained. It is computed using:

$$f_t = \sigma(W_f[x_t, h_{t-1}] + b_f) \quad (3)$$

where f_t is the forget gate value, W_f is the weight matrix, and b_f is the corresponding bias.

The **output gate** controls how much of the cell state should be exposed to the next hidden state. Its equation is:

$$o_t = \sigma(W_o[x_t, h_{t-1}] + b_o) \quad (4)$$

where o_t is the output gate value, W_o is the output gate weight matrix, and b_o is the output bias.

The **candidate cell state**, which contains new information that may be added to the cell state, is calculated as:

$$c\sim1 = \tanh(Wc[x_t, h_{t-1}] + b_c) \quad (5)$$

where *Wc* is the weight matrix for the candidate state and *bc* is the bias vector.

The **cell state** itself is updated by combining the effects of the forget and input gates with the candidate state, following:

$$c_t = f_t * c_{t-1} + it * c\sim1 \quad (6)$$

which represents the memory of the network.

Finally, the **hidden state** is updated based on the output gate and the new cell state using the equation:

$$h_t = o_t * \tanh(c_t) \quad (7)$$

This hidden state is then passed to the next time step and used for predictions or further computation. Through these interconnected gates and operations, LSTM networks maintain and update memory across time steps, enabling them to learn and predict from complex time-dependent patterns [10].

4. SYSTEM

The system framework design is shown in figure 1.

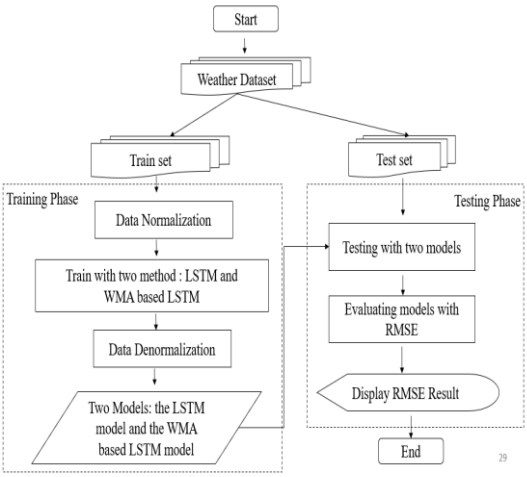


Figure 1. System Framework

In this figure, historical weather data from various regions in Myanmar—specifically Hmawbi, Sanchaung, Insein, Ahlone, Hlaingthaya, Mingala Taungnyunt, Thamwe, Botahtaung, Dawbon, and Yankin—spanning 10 years (from 2013 to 2022) are used to train a weather prediction model using the Long Short-Term Memory (LSTM) deep learning algorithm. The complete dataset contains 36,530 records, providing a rich source of information for modeling weather patterns over time. The dataset includes nine key attributes: date, daily maximum temperature, daily minimum temperature, daily relative humidity, wind speed, fog, cloud amount, rainfall, and daily weather types. There are 20 distinct weather categories represented in the data, capturing a wide range of meteorological conditions.

Before applying any smoothing technique, we first perform min-max normalization on the dataset to rescale all feature values to a common range between 0 and 1. This normalization process ensures that each feature contributes equally to the training process and prevents features with large numerical ranges from dominating the model's learning behavior. It also enhances model convergence and stability during training. Secondly, we use the weighted moving average method for removing noise in the normalized dataset and develop the weather prediction model by using LSTM and the refined dataset. We compare and evaluate the prediction results of these two developed models.

The original dataset is preprocessed for noise removal with weighted moving average. In the preprocessing step, as the collection of data contains useless data and data duplication, these data need to be preprocessed and cleaned for the effective analysis system. The purpose of data preprocessing is for the duplicated and noise data removal as they cannot be used efficiently for predictive analysis. In this step, the prediction task is simplified and cost of processing is significantly decreased in the training stage.

After preprocessing, this dataset is split into training and testing for this system. The training data is input into LSTM by tuning hyper-parameters to achieve desired accuracy. The training model is built using LSTM with the Adam optimizer is applied. The testing of system is evaluated. Then, this system uses LSTM model to predict the weather using the past 3 days and multivariate model. The experiments will be examined for example completeness, appropriateness and quality, and will illustrate the root mean square error and mean absolute error of the used deep learning algorithm. In this system, 32 kernels with 2 convolutional layers are used. For each convolutional layer, maximum pooling and rectified linear unit (ReLU) are used and the pooling size is 4 for only first pooling layer and is 2 for other pooling layers. RELU function is used at fully-connected layer. Evaluation is based on metrics like Root Mean Square Error (RMSE) demonstrating the model's effectiveness in delivering accurate weather forecasts.

4. PERFORMANCE EVALUATION

In this system, History data of weather condition in Hmawbi, Sanchaung, Insein, Ahlone, Hlaingthaya, Mingala Taungnyunt, Thamwe, Botahtaung, Dawbon, Yankin, Myanmar as input datasource. This system model is trained in Keras which is based on Tensorflow. The system is implemented by the ratio of data size: (Training – 80%, Testing – 20%). Firstly, randomly selected 80% records as training data and 20 % records for testing from this dataset. The key metrics of performance measures: root mean square error (RMSE) is evaluated for system analysis. 100 epochs are used for system evaluation.

Root mean squared error (RMSE) is a variation of MSE and is particularly useful when you want the error metric to be in the same units as the original data. RMSE is the square root of the MSE and is calculated as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \|y_i - \hat{y}_i\|^2}{n}} \qquad (8)$$

where n is the number of data points, y(i) is the i-th measurement, and y^(i) is its corresponding prediction.

The comperisons for prediction of weather attributes using LSTM and WMA with LSTM and actual measuers are depicted form Fig 2 and 3.Table 1 shows the comparative results for the performance of the proposed model for next-day prediction with WMA, LSTM, and only LSTM.

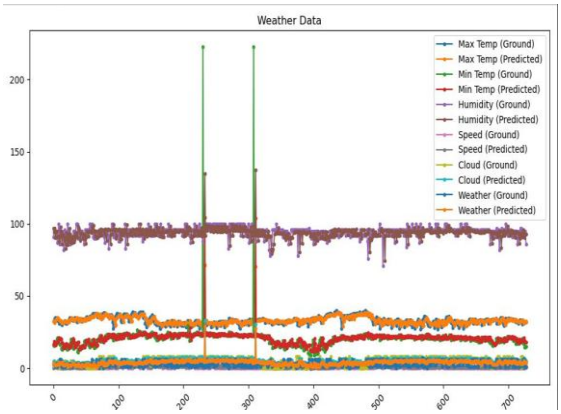


Fig. 2. Comparison of prediction for Weather Data using LSTM only

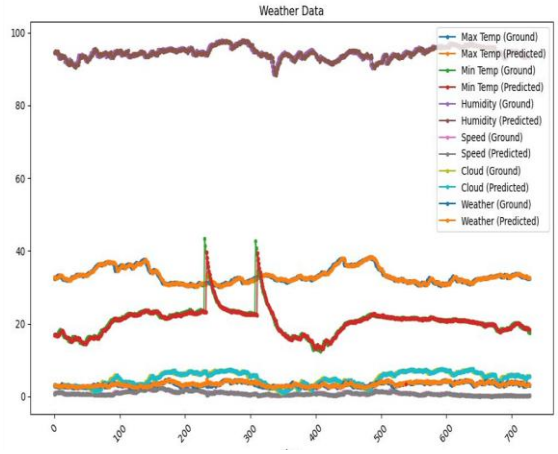


Fig. 3. Comparison of prediction for Weather Data using WMA with LSTM

Table 1. RMSE Results

Models	RMSE
WMA and LSTM	2.185
Only LSTM	5.270

According to evaluation results, a weighted moving average with a LSTM model is considered acceptable for time series data analysis systems.

4. CONCLUSIONS

Weather forecasting plays a crucial role in human daily life, influencing activities across sectors such as agriculture, transportation, and disaster management. To enhance prediction accuracy beyond what traditional machine learning methods can offer, this system employs advanced deep learning techniques. Specifically, it proposes an ensemble approach using Long Short-Term Memory (LSTM), a powerful recurrent neural network well-suited for time series data like weather records. The system is designed to forecast key weather attributes, including minimum and maximum temperature, humidity, cloud amount, types of weather, and wind speed, by training two separate models: one using standard LSTM and another using LSTM integrated with a Weighted Moving Average (WMA) technique for noise reduction. These models are trained on weather data from Myanmar spanning the years 2013 to 2022. The performance of both models is assessed using the Root Mean Square Error (RMSE) metric, allowing for a comparative analysis of prediction accuracy and the effectiveness of noise preprocessing. According to evaluation results, the weighted moving average with LSTM model is the best model for time series data analysis systems. In future research, the system will be improved by adding more parameters and an ensemble of deep learning.

REFERENCES

[1] Guici Chen, Sijia Liu, and Feng Jiang, “Daily Weather Forecasting Based on Deep Learning Model: A Case Study of Shenzhen City, China”, Special Issue Industrial Air Pollution Control in China, Atmosphere2022,13(8),1208;

[2] Keerthana. P. J, Rajeswari. P, Amirtharaj. R, PandiyaRajan. G, “Weather Forecasting using Deep Learning Algorithm”, INTERNATIONAL JOURNAL OF SPECIAL EDUCATION Vol.37, No.3, 2022.

[3] N. Shobha and T. Asha, “Monitoring weather based meteorological data: Clustering approach for analysis”, the International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 2017, pp.75-81.

[4] P. S. Mung, and S. Phyu, “Time Series Weather Data Forecasting Using Deep Learning”, ICCA 2019, 2019.

[5] R. Dalmau, M. Perez-Batlle, and X. Prats, “Estimation and prediction of weather variables from surveillance data using spatio-temporal Kriging”, IEEE/AIAA 36th Digital Avionics Systems Conference (DASC), 2017, pp.1-8.

[6] S. Navadia, P. Yadav, J. Thomas and S. Shaikh, “Weather prediction: A novel approach for measuring and analyzing weather data”, the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp.414-417.

[7] V. Leijen, M. Boone and K. Horgan,” Making numerical weather predictions portable compression of weather data for use in radar propagation modeling”, USNC-URSI Radio Science Meeting (Joint with AP-S Symposium), 2017, pp.12-19.

[8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning”, vol. 521, Nature, London, 27 May, 2015, pp. 436-444.

[9]https://corporatefinanceinstitute.com/resources/equities/weighted-moving-average-wma/

[10]https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network

Authors

Biography

May Mu Yar Han achieved the B.C.Sc in 2022 from the University of Computer Studies,Mandalay. She is currently pursuing as a M.C.Sc candidate at the same university.

Dr.Thin Thin Naing

Professor, Faculty of Computing, University of Computing Studies (Mandalay).

GRAMMAR ERROR CORRECTION SYSTEM USING A SEQUENCE-TO-SEQUENCE MODEL WITH AN ATTENTION MECHANISM

Nandar Win Khin[1], and Thandar Aung[2]

[1,2]Faculty of Information Science, University of Computer Studies (Mandalay)
nandarwinkhin21@gmail.com, thandarmyintmyathein@gmail.com

ABSTRACT

Grammar Error Correction (GEC) is an essential subfield of Natural Language Processing (NLP) with applications in language learning, academic writing, and automated proofreading. This system presents a GEC system that employs a Sequence-to-Sequence (Seq2Seq) architecture with an attention mechanism. The system architecture is based on an LSTM encoder–decoder framework, where the encoder processes the input erroneous sentence into hidden representations, and the decoder, guided by Bahdanau attention, generates the corrected output by focusing on the most relevant parts of the input. Experimental results show an overall Bilingual Evaluation Understudy (BLEU) score of 74.85 % on the test set. The findings demonstrate that attention mechanisms significantly enhance the model's ability to correct grammatical errors across multiple linguistic phenomena.

KEYWORDS

grammar error correction, sequence-to-sequence, attention mechanism, BLiMP, BLEU Score

1. INTRODUCTION

Grammar error correction (GEC) has become an important tool in educational technology, writing assistance, and professional editing software. Traditional approaches to GEC relied on rule-based systems or statistical machine translation. Recent advances in deep learning, particularly neural machine translation (NMT) models, have enabled more flexible and context-aware corrections.

The Sequence-to-Sequence (Seq2Seq) framework, originally developed for machine translation, treats GEC as a monolingual translation problem: the source sentence is ungrammatical, and the target sentence is its corrected form. However, standard Seq2Seq models compress all input information into a single context vector, which can lead to performance degradation on long or complex sentences.

To address this, attention mechanisms allow the decoder to dynamically focus on different parts of the source sentence at each generation step. This system explores an LSTM-based Seq2Seq model with an attention mechanism, trained on the Benchmark of Linguistic Minimal Pairs (BLiMP) dataset for GEC tasks.

2. LITERATURE REVIEWS

Grammar Error Correction (GEC) has progressed significantly, evolving from early rule-based methods to advanced deep learning approaches. Traditional statistical methods, such as phrase-based Statistical Machine Translation (SMT), approached GEC as a translation task—mapping

incorrect sentences to their correct forms. While effective for certain cases, these systems relied on fixed phrase tables and sparse feature representations, limiting their ability to generalize, especially for diverse or unseen grammatical errors.

The introduction of Neural Machine Translation (NMT) brought a paradigm shift with encoder–decoder architectures capable of learning rich contextual representations directly from data. Bahdanau et al. [1] enhanced this framework by introducing the attention mechanism, enabling the decoder to dynamically weight encoder outputs. This proved particularly beneficial for handling long or complex sentences, where errors often depend on distant token relationships. The attention mechanism's success inspired its adoption in GEC, allowing models to focus more precisely on error-relevant parts of the input.

More recent research has embraced Transformer-based architectures, such as GECToR [6] and BERT-based fine-tuning strategies, which replace recurrence entirely with self-attention. These models leverage large-scale pretraining and parallel processing to achieve state-of-the-art results on multiple GEC benchmarks. However, their high parameter counts and large data requirements make them less accessible for resource-constrained environments.

In this system, a Long Short-Term Memory (LSTM)-based sequence-to-sequence model with attention was selected instead of a Transformer. LSTMs offer strong sequential modeling capabilities with fewer parameters, making them a better fit for medium-sized datasets such as the BLiMP corpus. Moreover, LSTM-based architectures are less dependent on massive pretraining and can be trained effectively on modest hardware, such as the NVIDIA GeForce 940MX GPU used in this work, without sacrificing the ability to capture key grammatical relationships.

3. System Architecture

The overall workflow of the Grammar Error Correction (GEC) system is illustrated in Figure 1. The process begins with the BLiMP dataset, which serves as the primary source of grammatically incorrect and correct sentence pairs. Before training, the dataset undergoes preprocessing, including tokenization, vocabulary construction, and filtering of rare words.

After preprocessing, the dataset is split into two subsets: the training set and the test set. The training set is used to optimize the parameters of the Seq2Seq model with attention, while the test set is reserved for performance evaluation.

The training phase involves feeding the training data into a sequence-to-sequence model with an attention mechanism. The encoder processes the input sentences into contextual representations, the attention layer aligns encoder states with decoder steps, and the decoder generates corrected sentences. At the end of training, a trained model is obtained.

In the testing phase, the trained model is applied to the test set to generate corrected sentences. Model performance is then evaluated using the BLEU score, which measures the similarity between the model's output and the reference corrected sentences. Finally, the system produces a grammatically corrected sentence, marking the completion of the workflow.

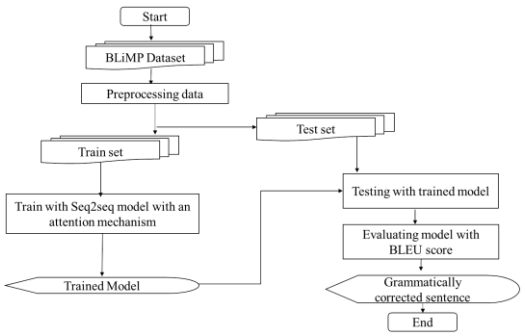


Figure 1: System Architecture

3.1. DATASET

The Benchmark of Linguistic Minimal Pairs (BLiMP) dataset contains 67,000 sentence pairs covering many major grammatical phenomena, such as subject-verb agreement, argument structure, binding, and tense. Each pair contains a grammatically correct sentence and a minimally different ungrammatical variant.

The ungrammatical sentence was designated as the input, and the corresponding correct sentence as the target. Sentences were tokenized, lowercased, and padded to a maximum length of 15 tokens.

The dataset was split into:

- Training: 80% (53,600 pairs)
- Testing: 20% (13,400 pairs)

4. METHODOLOGY

4.1. MODEL ARCHITECTURE

Grammar Error Correction system adopts a standard Sequence-to-Sequence (Seq2Seq) encoder–decoder architecture [8] augmented with the Bahdanau attention mechanism [1]. This design enables the decoder to selectively focus on relevant parts of the source sequence during prediction, overcoming the limitations of fixed-length context vectors in vanilla Seq2Seq models.

4.1.1. Sequence-to-Sequence (Seq2Seq) architecture

A neural network model designed to map input sequences to output sequences. This architecture is made up of an encoder and a decoder. Seq2seq architecture is suitable for tasks where the input and output are sequences of varying lengths but may struggle with long sentences without attention.

4.1.2. Encoder

In this system, the encoder is implemented as a two-layer Long Short-Term Memory (LSTM) network [4], which transforms the input sentence into a sequence of hidden states while capturing long-term dependencies. At each time step t, the LSTM computes its internal gates and states as:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (2)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (3)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (4)$$

$$h_t = o_t \odot \tanh(c_t) \quad (5)$$

Here, σ denotes the sigmoid activation function, ⊙ represents element-wise multiplication, x_t is the embedding vector for the input token at time t, h_{t-1} and c_{t-1} are the previous hidden and cell states, and W and b are learnable weight matrices and bias vectors. This gating mechanism allows the LSTM to effectively model both short- and long-range dependencies, making it well-suited for language tasks involving grammatical structure [3][4].

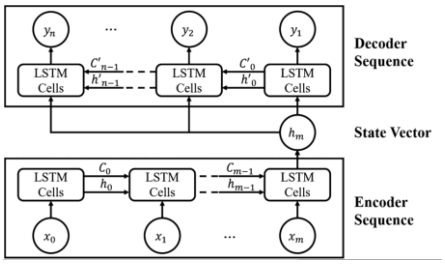


Figure 2: One-layer-LSTM-based-seq2seq-model-structure

4.1.3. Attention Mechanism

The attention component follows the additive attention formulation introduced by Bahdanau et al. [1], which computes alignment scores between the decoder's current hidden state and each encoder hidden state. For a decoder time step t and encoder hidden state h_j, the alignment score is:

$$e_{tj}=s_{t-1}T\ W_a h_j \quad (6)$$

where s_{t-1} is the decoder's previous hidden state, Wa is trainable weight matrix. The attention weights are obtained via the softmax function:

$$\alpha_{tj} = \frac{\exp(etj)}{\sum_{k=0}^{T}\exp(etk)} \quad (7)$$

where s_{t-1} is the decoder's previous hidden state, Wa is learnable parameter.

The context vector c_t is a weighted sum of encoder hidden states:

$$c_t = \sum_{j=1}^{T} \alpha_{ti} \quad (8)$$

This mechanism allows the decoder to dynamically attend to different source positions, improving its capacity to correct errors that depend on distant tokens (e.g., subject–verb agreement, tense consistency).

4.1.4. Decoder

The decoder is also implemented as a two-layer LSTM, which generates the corrected sentence token-by-token. At each step, the decoder receives as input the embedding of the previously generated token, the previous hidden state, and the context vector from the attention mechanism. The output probability distribution over the vocabulary is computed through a linear projection followed by a softmax activation.

This combination of LSTM-based sequence modeling with attention-based alignment has been shown in prior research to yield significant improvements in grammatical error correction [2][5][6].

5.2. IMPLEMENTATION

The Grammar Error Correction model is implemented in Python using the PyTorch deep learning framework. The BLiMP dataset, containing pairs of grammatically correct and incorrect sentences, is first loaded and preprocessed. All sentences are converted to lowercase to reduce vocabulary size, while punctuation is preserved to retain grammatical cues. Tokenization is performed using the spaCy English tokenizer, which accurately handles English word boundaries. A vocabulary dictionary is then constructed by assigning a unique index to each token, excluding tokens with a frequency of less than two to minimize sparsity. Finally, all sequences are padded with a special <pad> token or truncated to a fixed length of 15 tokens to ensure uniform dimensions for batch processing.

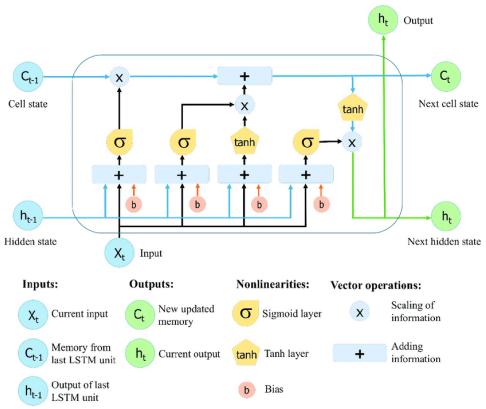


Figure 3: Structure of LSTM

The model architecture follows a standard sequence-to-sequence encoder–decoder framework with Bahdanau attention. The encoder consisted of a two-layer Long Short-Term Memory (LSTM) network that transforms the input token embeddings into a sequence of hidden states. The decoder is also a two-layer LSTM that generated corrected sentences token by token. Both encoder and decoder embedding layers has a dimensionality of 256. The attention mechanism computes alignment scores between the decoder's hidden state and all encoder hidden states using a feedforward neural network with tanh activation, followed by softmax normalization to produce attention weights, allowing the model to focus on contextually relevant parts of the [1].

The model is trained with a batch size of 32, a learning rate of 0.001 using the Adam

optimizer, and a dropout rate of 0.5 to mitigate overfitting. Teacher forcing is applied with a ratio of 0.5, meaning that during training, the decoder received the correct token from the target sequence as input in half of the cases instead of its own previous prediction. Early stopping is used to terminate training when the validation loss does not improve for three consecutive epochs.

The model is trained for 20 epochs, with training and validation loss monitored at each epoch. Backpropagation through time (BPTT) is applied to update model parameters, allowing the LSTM to capture long-term dependencies in the sequences.

Figure 4: Train and Validation Loss

5.3. Model Evaluation

The performance of the Grammar Error Correction (GEC) system is evaluated using the Bilingual Evaluation Understudy (BLEU) metric. BLEU compares the corrected sentences generated by the model with the reference sentences in the test set by measuring the degree of n-gram overlap. In this evaluation, the test set is used to ensure that the model's performance is assessed on unseen data, which provides a fair estimate of its generalization ability.

The BLEU score is defined as:

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \quad (9)$$

Where:

p_n = precision of n-grams between candidate output and reference sentences

w_n = weight assigned to each n-gram order

BP = brevity penalty, which prevents the system from gaining high scores by generating overly short outputs

The final BLEU score, calculated on the test set, provides a quantitative measure of how well the model performs in correcting grammatical errors. In this study, the model achieved a BLEU score of 74.85%, which indicates that the generated corrections are highly similar to the references.

5.4. Experimental Results

The experimental analysis demonstrated that the Seq2Seq architecture with Bahdanau attention effectively addressed a wide range of grammatical errors, including subject–verb agreement, tense consistency, pluralization, and pronoun usage. The model performed particularly well in cases requiring long-distance dependencies, where attention helped the decoder focus on relevant tokens regardless of their position in the input sequence. Attention weight visualizations further revealed that the model tended to assign higher weights to verbs when correcting subject–verb agreement errors, and to auxiliary verbs when correcting tense-related mistakes.

5.4.1. Qualitative Examples

Input (Incorrect)	Output (Corrected)	Reference (Correct)
A lot of teenagers think that themselves praised Cheryl.	A lot of teenagers think that they praised Cheryl.	A lot of teenagers think that they praised Cheryl
Steven can't wear those sock.	Steven can't wear those socks.	Steven can't wear those socks.
They was here yesterday.	They were here yesterday.	They were here yesterday.

Overall, the results validate that the integration of an attention mechanism significantly improves grammar error correction performance, particularly for complex and context-dependent corrections.

6. CONCLUSION

This system demonstrates that an LSTM-based Seq2Seq model with an attention mechanism, trained on the BLiMP dataset, can achieve effective grammar error correction. The attention mechanism significantly enhances the model's ability to identify and correct grammatical errors, particularly in longer sentences and for linguistic phenomena that require precise token alignments. Experimental evaluation using the BLEU score on the test set confirmed that the model produces corrections that closely match the reference sentences, showing its potential for practical applications in English composition and language learning tools. Moreover, the results highlight the importance of integrating attention into Seq2Seq architectures, as it allows the model to focus dynamically on relevant parts of the input, leading to improved accuracy and robustness. Future work could further improve performance by incorporating larger, real-world datasets, using transformer-based architectures.

REFERENCES

[1] Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations*. https://arxiv.org/abs/1409.0473

[2] Chollampatt, S., & Ng, H. T. (2018). A multilayer convolutional encoder–decoder neural network for grammatical error correction. In Proceedings of the AAAI Conference on Artificial Intelligence, 32(1). https://doi.org/10.1609/aaai.v32i1.11942

[3] Graves, A. (2013). Speech recognition with deep recurrent neural networks. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (pp. 6645–6649). IEEE. https://doi.org/10.1109/ICASSP.2013.6638947

[4] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[5] Ji, J., et al. (2017). A nested attention neural hybrid model for grammatical error correction. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2, 404–410. https://doi.org/10.18653/v1/P17-2064

[6] Omelianchuk, K., et al. (2020). GECToR – Grammatical error correction: Tag, not rewrite. *Proceedings of the 15th Workshop on Innovative Use of NLP for Building* Educational Applications, 163–170. https://arxiv.org/abs/2005.12592

[7] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (pp. 311–318). ACL. https://aclanthology.org/P02-1040

[8] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27, 3104–3112. https://arxiv.org/abs/1409.3215

[9] Warstadt, A., et al. (2020). BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8, 377–392. https://doi.org/10.1162/tacl_a_00321.

Authors

Biography

Nandar Win Khin received the Bachelor of Computer Science (B.C.Sc.) degree in 2023 from the University of Computer Studies, Monywa, Myanmar. She is currently pursuing her Master of Computer Science (M.C.Sc.) degree at University of Computer Studies, Mandalay.

FAKE NEWS DETECTION USING GLOVE-LSTM MODEL

Sandar Win[1], and Yu Ya Win[2]

[1,2] Faculty of Computer Science, University of Computer Studies (Mandalay)
sandarwin.sdwpb@gmail.com, yuyawin@gmail.com

ABSTRACT

Fake news has emerged as a critical concern in the digital era, influencing public opinion, politics, and society on a global scale. The rapid proliferation of misinformation through online platforms and social media highlights the urgent need for effective and reliable detection systems. Detecting fake news automatically, however, is a challenging task due to the complexity of natural language and the subtle differences between deceptive and factual statements. This study explores the integration of Global Vectors for Word Representation (GloVe) embeddings with a Long Short-Term Memory (LSTM) neural network for fake news detection. GloVe embeddings are utilized to capture semantic and syntactic relationships between words, while the LSTM network models contextual dependencies in textual sequences. Experimental evaluation on the ISOT Fake News Dataset demonstrates that the GloVe-LSTM approach achieves high accuracy and balanced performance across precision, recall, and F1-score, offering a robust solution for distinguishing between fake and not fake news articles.

KEYWORDS

Fake News Detection, ISOT, GloVe, LSTM.

1. INTRODUCTION

The rapid circulation of misleading information poses a significant problem in the modern digital era. Social media and online platforms enable the rapid spread of news, making it increasingly difficult for users to distinguish between authentic and deceptive content. As a result, fake news detection has emerged as a pressing research area that integrates natural language processing (NLP), deep learning, and data science techniques. While traditional machine learning methods, such as Naïve Bayes, Decision Trees, and Support Vector Machines, have been explored, they often struggle to capture the deeper contextual meaning of text. In contrast, deep learning approaches, particularly Long Short-Term Memory (LSTM) networks, provide stronger capabilities in modeling sequential dependencies. This paper contributes by leveraging pre-trained GloVe word embeddings in combination with an LSTM architecture to enhance semantic representation and improve the accuracy of fake news detection.

2. LITERATURE REVIEWS

Several studies have explored the challenge of fake news detection. Traditional methods primarily relied on handcrafted features and classical classifiers, such as SVMs and decision trees, but these approaches achieved only moderate success and struggled to generalize semantically across diverse news content [4][5]. One key limitation was their inability to capture deeper contextual information within text. With the rise of deep learning, recurrent neural networks (RNNs) and especially Long Short-Term Memory (LSTM) models were introduced, offering substantial improvements in modeling sequential dependencies in language [2][10].

At the same time, distributed word representations such as Word2Vec and GloVe provided dense vector embeddings that preserved semantic and syntactic relationships, proving far more effective than sparse feature-based representations [3]. Building on these advances, more recent work has shown that combining pre-trained embeddings with LSTM architectures yields strong results across multiple NLP tasks—including sentiment analysis, spam detection, and fake news classification—demonstrating the value of integrating semantic priors with sequential modeling [1][7][9].

3. Model Architecture

The model architecture starts with an Embedding Layer initialized with pre-trained GloVe word embeddings, which converts each word into a dense vector representation, capturing both semantic and syntactic information [3]. These embeddings are fed into an LSTM layer that models long-term dependencies and sequential patterns in the text, capturing contextual relationships between words and refining features for richer representations [9].

To reduce overfitting, Dropout layers are applied after the LSTM layer. The output of the LSTM layer is passed through a Dense layer with ReLU activation to extract high-level features, followed by a final Dense output layer with a sigmoid activation function that classifies news articles as fake or not fake [9].

3.1 Global Vector (GloVe)

GloVe, short for Global Vectors for Word Representation, is a word embedding approach that was created by Stanford University [3]. One benefit of this method over Word2Vec is that it takes into account not only the local statistics (contextual information of the words), but also the global statistics (word co-occurrence) from the whole text 28 corpus. GloVe uses global text-level co-occurrence information to generate vector representations of words. This element is significant since word-word co-occurrences may contain rich semantic information. For instance, in a vast collection of texts, the term "solid" is more prone to appearing along with "ice" rather than "steam". Conversely, the word "gas" is likely to co-occur more often with "steam" than with "ice".

3.2 Long Short-Term Memory

LSTM networks are a specialized form of recurrent neural networks (RNNs) designed to overcome the vanishing and exploding gradient problems that affect standard RNNs [10]. By introducing a memory cell and gating mechanisms (input, forget, and output gates), LSTMs regulate how information is stored, updated, and forgotten across time steps [2][10]. At each time step t, the operations of an LSTM can be expressed as follows:

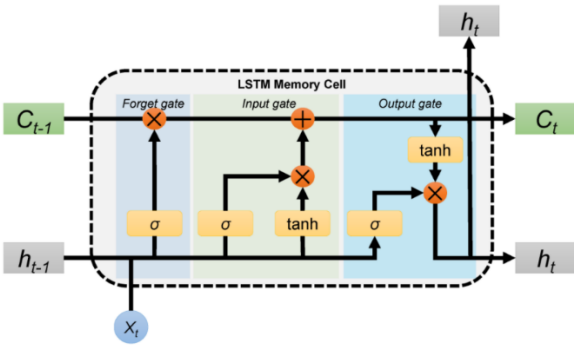


Figure 3.1: LSTM Architecture

Forget Gate – decides what information from the previous memory should be discarded:

$$ft = \sigma (Wf [xt, ht\text{-}1] + bf) \quad (1)$$

Input Gate – decides which incoming inputs are preserved in the cell state:

$$it = \sigma (Wi [xt, ht\text{-}1] + bi) \quad (2)$$

$$\tilde{c}\, t = \tanh (Wc [ht\text{-}1, xt] + bc) \quad (3)$$

Cell State Update – updates the memory cell by combining the effects of the forget and input gates:

$$Ct = ft \odot ct\text{-}1 + it \odot \tilde{c}\, t \quad (4)$$

Output Gate – controls what information will be passed to the next hidden state:

$$ot = \sigma (Wo [xt, ht\text{-}1] + bo) \quad (5)$$

$$ht = ot \odot \tanh(ct) \quad (6)$$

Where xt is the input vector, ht-1 is the hidden state from the previous step, ct is the updated memory cell, and $\odot$ denotes element-wise multiplication. The symbols σ and tanh denote the sigmoid and hyperbolic tangent activation functions, in that order. Figure 3.1 illustrates the internal architecture of the LSTM unit, showing how gates interact with the memory cell to regulate information flow.

Through this mechanism, LSTMs can retain important information over long sequences while filtering out irrelevant details. This makes them particularly effective in tasks such as fake news detection, where understanding the context of words across entire sentences or documents is crucial.

4. Dataset

The dataset used in this study is the ISOT Fake News Dataset, collected by the Information Security and Object Technology (ISOT)

Research Laboratory at the University of Victoria [11]. It is one of the most widely used benchmark datasets for fake news detection research, providing a balanced and well-structured collection of news articles.

The dataset consists of two primary categories:

- Real News: Articles gathered from credible and reputable sources, including Reuters and The New York Times.
- Fake News: Articles collected from unreliable websites identified by fact-checking organizations such as PolitiFact and information sources referenced by Wikipedia [11].

In total, the ISOT dataset contains 21,417 real news articles and 23,481 fake news articles, offering a substantial and balanced representation of both classes [11]. Each article is clearly labeled as either real or fake, which makes the dataset highly suitable for supervised machine learning and deep learning tasks [12]. The dataset includes essential attributes such as title, body text, and class label, which are particularly valuable for text-based feature extraction and classification tasks.

For experiments, the dataset is typically split into 80% training and 20% testing, ensuring fair model evaluation on unseen data [6]. To handle any minor class imbalance, resampling techniques such as oversampling or undersampling may be applied [5]. This structured design makes the ISOT dataset a reliable foundation for developing and testing robust fake news detection models

5. METHODOLOGY

For this system, establishing a comprehensive understanding of the underlying technologies is crucial. This involves deep learning, natural language processing (NLP), and the implementation of Python-based frameworks such as Keras and TensorFlow within Google Colab. Knowledge of NumPy and Pandas for numerical and text preprocessing is equally important. All required libraries must be properly configured to ensure smooth execution of the system.

5.1 Data Pre-processing

Before training the fake news detection model, the ISOT dataset undergoes extensive pre-processing to ensure the text is clean, standardized, and suitable for semantic analysis. Real-world news articles are inherently messy, often containing punctuation, special symbols, numbers, URLs, HTML tags, and redundant or irrelevant phrases that do not contribute to understanding the content. For example, the headline “Donald Trump Sends Out Embarrassing New Year’s Eve Message” contains capital letters, punctuation, and words that may not significantly influence the classification task. To handle such complexities, a sequence of pre-processing steps is applied to normalize and clean the text, making it suitable for further processing and modeling.

The preprocessing steps are as follows:

The first step is lowercasing, where all words are converted into lowercase to maintain uniformity and avoid treating the same word in different cases as separate entities. This is implemented using the built-in Python string method .lower(), which ensures that words such as “Trump” and “trump” are represented identically. For example, running the command sentence.lower() converts the text “Donald Trump Sends Out Embarrassing New Year’s Eve Message” into “donald trump sends out embarrassing new year’s eve message.”

The next step is noise removal, which eliminates unwanted characters, punctuation, numbers, URLs, and non-informative expressions commonly found in online news data. This process is efficiently handled using the re (regular expression) library in Python, where specific patterns are designed to filter out hyperlinks, special characters, and redundant phrases such as “CITY (Reuters) –.” For instance, the command re.sub(r'[^a-zA-Z\s]', '', sentence) removes all symbols except alphabetic characters and spaces, producing the cleaned sentence “donald trump sends out embarrassing new years eve message.”

After cleaning, stopword removal is performed to discard high-frequency functional words such as “is,” “the,” “out,” and “and” that do not contribute meaningful semantic information for classification tasks. This step is typically implemented using the stopwords corpus from the Natural Language Toolkit (NLTK), which provides a predefined list of common English stopwords. The method involves filtering out words that appear in the stopword list, so that

the sentence “donald trump sends out embarrassing new years eve message” is reduced to “donald trump sends embarrassing new years eve message.”

Finally, tokenization is applied to split the cleaned sentence into discrete units, or tokens, where each token represents a meaningful word. Tokenization allows the text to be represented as a sequence of structured inputs rather than raw strings, which is crucial for subsequent embedding and modeling. This can be carried out using NLTK’s word_tokenize function, which segments the sentence into a list of tokens. Running the function on the preprocessed text yields the representation [“donald”, “trump”, “sends”, “embarrassing”, “new”, “years”, “eve”, “message”].

5.2 Creating Vocabulary and Word Embeddings

Once pre-processing is complete, the textual data is transformed into numerical representations so that it can be interpreted by machine learning models. This stage involves three major steps: vocabulary creation, sequence padding, and the integration of GloVe word embeddings. Each step relies on specific methods and libraries that facilitate the conversion of raw tokens into structured numerical vectors while preserving the semantic meaning of the text.

The first step is vocabulary creation, which constructs a dictionary of all unique tokens present in the training dataset. Each word is assigned a distinct numerical index to ensure consistency across the dataset. This process is typically implemented using the Tokenizer class from the Keras library, where the fit_on_texts() method scans the corpus and automatically builds the vocabulary mapping. For example, applying this method to the tokenized sentence “donald trump sends embarrassing new years eve message” results in a mapping such as {“donald”: 1, “trump”: 2, “sends”: 3, “embarrassing”: 4, “new”: 5, “years”: 6, “eve”: 7, “message”: 8}. This indexing scheme provides a standardized numerical reference for each word, ensuring that identical words are treated consistently across all training and testing samples.

The second step is sequence padding, which guarantees that all input sentences have a uniform length before being passed into the model. Since LSTM networks require inputs of equal dimensionality, shorter sequences are padded with zeros, while longer sequences are truncated to a predefined maximum length. This is carried out using the pad_sequences() function from Keras.preprocessing.sequence. For example, a sentence of length five in a system with maximum sequence length set to ten would be padded with five zeros at the end, whereas a sentence of length fifteen would be truncated to retain only the first ten tokens. This step prevents dimensionality mismatches and ensures efficient batch training.

The third step involves GloVe word embeddings, which provide semantically meaningful vector representations for each word in the vocabulary. Unlike raw integer encoding, which merely represents the position of a word in the dictionary, GloVe embeddings map words into a high-dimensional dense vector space based on statistical co-occurrence. These embeddings are typically integrated by first downloading a pre-trained GloVe file such as glove.6B.100d.txt, and then using Python’s numpy library to load the vectors into memory. Each word in the vocabulary is mapped to its corresponding embedding, and for words not present in the pre-trained dictionary, random initialization is applied to avoid gaps in representation. The resulting embedding matrix, which is passed into the Embedding layer of Keras, has rows corresponding to word vectors and serves as the foundation for the LSTM model to capture contextual relationships.

5.3 Training the Model

The fake news detection system is built on a Long Short-Term Memory (LSTM) network to classify news articles as fake or not fake. LSTMs are effective for this task as they can model long-term dependencies and sequential relationships in text, which is crucial for capturing context and semantics. The model is implemented in Keras using a Sequential architecture, where layers are stacked to process data from embeddings to the final classification.

Training was configured with carefully chosen hyperparameters to ensure stability and avoid overfitting. A batch size of 256 was used, and the model was trained for three epochs, allowing the dataset to pass through the network three times. To enhance convergence,

a callback was implemented to monitor validation accuracy and halve the learning rate if no improvement was observed over two consecutive epochs, with a minimum learning rate set at 0.00001.

The architecture begins with an Embedding layer initialized with pre-trained GloVe embeddings. These embeddings were set as non-trainable to preserve the semantic knowledge encoded in GloVe while reducing the need for large training data. Two LSTM layers were stacked sequentially: the first with 128 units, recurrent dropout (0.25), and dropout (0.25), configured with return_sequences=True to pass the sequence output to the next layer; the second with 64 units and lower dropout (0.1). This layered design allows the model to extract complex temporal patterns and hierarchical features from text.

Following the LSTM layers, a Dense layer with 32 units and ReLU activation was added to capture higher-level representations, while the final output layer used a single neuron with sigmoid activation to generate a probability score for binary classification. The model was compiled with the Adam optimizer for adaptive learning rate adjustment, binary cross-entropy as the loss function, and accuracy as the evaluation metric.

6. EXPERIMENTAL RESULT

To assess the performance of the Fake News Detection system using GloVe embeddings and LSTM, several evaluation metrics and visualization methods were employed. The model was tested on a separate dataset consisting of 11,225 samples, balanced between fake and not fake news. The evaluation primarily relied on accuracy, precision, recall, and F1-score, as these metrics provide a comprehensive understanding of classification performance.

	precision	recall	f1-score	support
Fake	0.99	0.98	0.99	5858
Not Fake	0.98	0.99	0.99	5367
accuracy			0.99	11225
macro avg	0.99	0.99	0.99	11225
weighted avg	0.99	0.99	0.99	11225

Figure 6.1 The classification report

The classification report in Figure 6.1 further validates model performance. For fake news, the precision reached 0.99, recall 0.98, and F1-score 0.99. For not fake news, precision was 0.98, recall 0.99, and F1-score 0.99. The overall accuracy achieved was 0.99, with both macro-average and weighted-average scores for precision, recall, and F1-score remaining at 0.99. These balanced results confirm that the model performs consistently across both classes without bias.

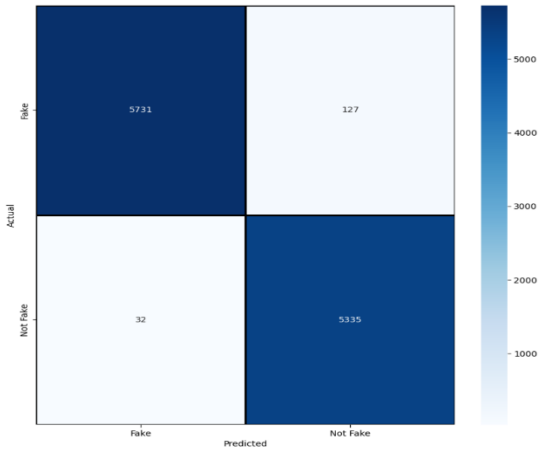


Figure 6.2 Performance Confusion Matrix (GloVe-LSTM)

The confusion matrix provides detailed insights into classification outcomes. Out of 5,858 fake news articles, the model correctly identified 5,731 as fake, with only 127 misclassified as not fake. Similarly, out of 5,367 not fake news articles, 5,335 were correctly classified as not fake, with only 32 misclassified as fake. These results indicate very low false positive and false negative rates, proving the system's robustness and reliability. The near-balanced classification across both categories further demonstrates the model's effectiveness in detecting fake news.

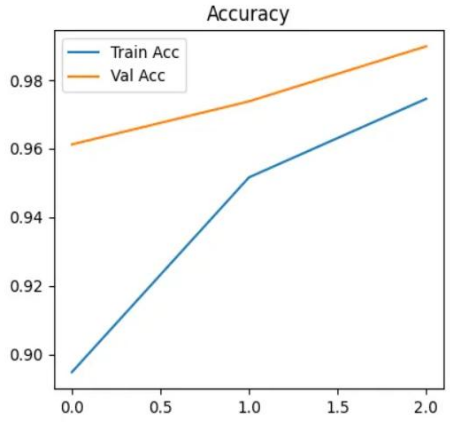


Figure 6.3: Training and Testing Accuracy

The training and validation accuracy trends are shown in Figure 6.3**.** Training accuracy improved steadily from 90% in the first epoch to 97.8% by the third epoch. Validation accuracy showed slightly higher performance, starting at 96% and reaching 99% in the final epoch. This consistent improvement highlights the model's ability to generalize well to unseen data, with GloVe embeddings supporting semantic representation and LSTM effectively capturing sequential dependencies. Overall, the accuracy analysis demonstrates that the model provides highly reliable classification results.

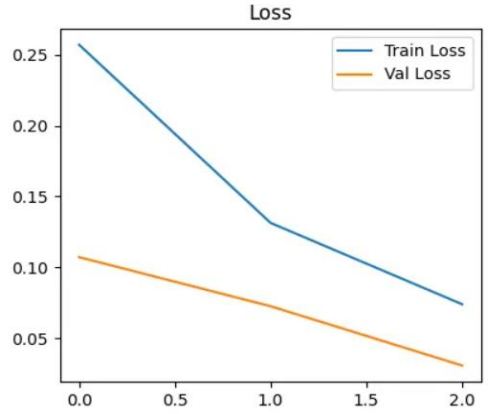


Figure 6.4: Training and Testing Loss

The training and validation loss curves are presented in Figure 6.4. Training loss decreased from 0.25 at the first epoch to 0.09 by the third epoch, while validation loss dropped from 0.11 to 0.03 during the same period. The consistently lower validation loss compared to training loss indicates strong generalization and the absence of overfitting. Dropout and recurrent dropout regularization, combined with early stopping, contributed to the stability of the model. Together with the accuracy analysis, these loss results confirm that the GloVe-LSTM model achieves effective error minimization.

The experimental evaluation demonstrates that the LSTM model with GloVe embeddings achieves outstanding performance, reaching up to 99% accuracy. Precision, recall, and F1-score results further confirm the system's robustness in detecting fake news. Overfitting is mitigated through dropout regularization and early stopping, ensuring optimal training efficiency. Compared with traditional classifiers such as logistic regression, the GloVe-LSTM model provides superior results by leveraging sequential text dependencies.

These results highlight the effectiveness of integrating pre-trained embeddings with deep learning. While GloVe captures semantic meaning, LSTM effectively models temporal dependencies in text. Nonetheless, challenges remain in addressing sarcasm, multi-modal fake news (combining images and text), and generalization to completely unseen domains. Future work can explore transformer-based architectures like BERT and GPT, which have achieved state-of-the-art performance in NLP tasks. Additionally, combining fact-checking databases with deep learning models could further enhance fake news detection systems.

6. CONCLUSION

The rapid spread of fake news across digital platforms poses serious social and political challenges, and traditional text classification methods like TF–IDF and Bag-of-Words often fail to capture semantic meaning and context. To address this, the system combines pre-trained GloVe embeddings with Long Short-Term Memory (LSTM) networks, leveraging rich semantic representations and sequential modeling to detect deceptive content. Experimental results show that this approach achieves 99% classification accuracy, demonstrating its effectiveness and robustness. By integrating advanced word embeddings with deep sequence models, the system provides a powerful solution for fake news detection, with potential applications in social media monitoring, online fact-checking, and news verification, while also paving the way for future enhancements such as multilingual and multimodal detection.

REFERENCES

[1] Eren Sahin, Chunyang Tang, and Mohammad A. Al-Ramahi, "Fake News Detection on Social Media: A Word Embedding-Based Approach," Aug. 2022.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, "Deep Learning", MIT Press, 2016.

[3] Jeffrey Pennington, Richard Socher, and Christopher D. Manning, "GloVe: Global Vectors for Word Representation," in Proceedings of the EMNLP, 2014.

[4] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu, "Fake News Detection on Social Media: A Data Mining Perspective," SIGKDD Explorations, 2017.

[5] Kamran Kowsari, K. Jafari Meimandi, Mihtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and

Donald Brown, “Text Classification Algorithms: A Survey,” Information (Switzerland), vol. 10, no. 4, 2019.

[6] Nihel Baarir and Abdelhamid Djeffal, “Fake News Detection Using Machine Learning,” IEEE, 2020.

[7] Priya Shrivastava and Dilip Kumar Sharma, “Fake Content Identification Using Pre-Trained GloVe-Embedding,” IEEE, 2021.

[8] Rishibha Sharma, Vidhi Agarwal, Sushma Sharma, and Meenak, “An LSTM-Based Fake News Detection System Using Word Embeddings-Based Feature Extraction,” Springer, 2021.

[9] Sangita M. Jaybhaye, Vivek Badade, Aryan Dodke, Apoorva Holkar, and Piyanka Lokhande, “Fake News Detection using LSTM-based Deep Learning Approach,” 2023.

[10] Sepp Hochreiter and Jurgen Schmidhuber, “Long Short-Term Memory,” Neural Computation, 1997.

[11] University of Victoria, “ISOT Fake News Dataset,” 2018. [Online]. Available: https://onlineacademiccommunity.uvic.ca/isot/wp-content/uploads/sites/7295/2023/02/ISOT_Fake_News_Dataset_ReadMe.pdf

[12] Z Khanam, B N Alwasel, H Sirafi, Mamonnm Rashid, “Fake news detection using classical machine learning approaches”. March 2021.

Author’s Biography

Sandar Win completed the Bachelor of Computer Science (B.C.Sc) in 2023 at the University of Computer Studies, Meiktila. She is now pursuing a Master’s degree at the University of Computer Studies, Mandalay.

INTELLIGENT HUMAN DETECTION AND ALERT SYSTEM WITH DEEP LEARNING MODEL

Kyal Sin Lin[1], and Thein Htay Zaw[2]

[1,2]Department of Information Technology Supports and Maintenance, University of Computer Studies (Mandalay)
errorkyalsin@gmail.com, theinhtayzaw.cu@gmail.com

Abstract

Human detection is a critical task in computer vision with wide-ranging applications such as surveillance systems, crowd monitoring, autonomous vehicles, and smart city infrastructures. Accurate and reliable human detection in unrestricted environments remains a challenging problem due to dynamic backgrounds, lighting variations, occlusions, and diverse human appearances. To address this problems, the system used gamma correction and bilateral filter. In this study, Faster Region-Based Convolutional Neural Network (Faster R-CNN) architecture integrated with a ResNet-based feature extractor is used for human detection to address this challenging problem. The model is trained on a Kaggle.com human dataset. After detection, the system also performs individual counting of humans in each frame based on bounding box confidence scores. The results demonstrate the effectiveness of the approach in accurately detecting and localizing humans across diverse options.

KEYWORDS

Computer Vision, Human Detection, CNN, Faster R-CNN, RPN, ResNet

1. INTRODUCTION

A computer vision technique called object detection may find things in an image and determine them. The coordinates of the objects in an image that it has been learned to determine will be returned by an object detection system [1]. Additionally, the system will support a confidence rating that signifies its level of assurance regarding the accuracy of a forecast. Both object classification and localization are essential for object detection [2]. Classification establishes the category of one or more items in the image, like a car, human, or dog. The process of localization requires accurately locating and marking an object in an image, usually by including it in a bounding box [3].

In many fields, classification is an essential task. Correctly classifying data into various collections is essential for creating decisions and addressing problems. Simple machine learning methods might not be able to address the complexity and variety of the data due to the eruption of big data [4]. A form of machine learning known as "deep learning" uses multi-layered artificial neural networks to develop representations of the input data [5].

2. RELATED WORK

Histogram of Oriented Gradients is a feature vector that is achieved from image brightness. HOG displays the shape of objects, and puts the gradients of the bordering pixels as a histogram for local domain, which is desired for changing illumination and shadows.

The drawbacks of Histogram of Oriented Gradients (HOG) involve its high computational power and slow performance in complex or real-time scenarios, a lack of invariance to rotations and critical scale changes, and a limited ability to capture fine-grained details complex textures compared to deep learning methods [9]. Despite these limitations, HOG remains a widely used traditional feature extraction technique due to its simplicity.

3. METHODOLOGY

Deep learning models are capable of obtaining state-of-the-art performance in a variety of tasks, including classification, by automatically extracting relevant features and patterns from the input data. One kind of deep learning model constituted especially for image classification is the convolutional neural network (CNN) [6].

Convolution procedures are applied by CNNs to extract features from images. The images are then categorized using these features [7]. CNNs have indicated remarkable efficacy in image classification applications, including object detection. Two main categories of object detection techniques exist. First, two-stage detectors work in two phases: they first propose a possible region before categorizing the region. R-CNN, Fast R-CNN, and Faster R-CNN are two-stage detectors. Second, the bounding boxes and class probabilities for each region of the image are precisely predicted by single-stage detectors. Examples involve SSD (Single Shot MultiBox Detector) and YOLO (You Only Look Once) [8].

3.1 Computer Vision

A subfield of artificial intelligence (AI) called computer vision (CV) authorizes computers to understand, evaluate, and illustrate visual data just like people do [1]. Obtaining, processing, and examining pictures are all segment of computer vision. It provides computers the ability to process, understand, and derive invaluable information from pictures. Catching images is the first stage in computer vision. This involves using cameras, sensors, CCTV cameras, mobile phone camera or other gadgets to take pictures. It requires utilizing a variety of methods, including object detection, identification, segmentation, and recognition, to extract information from images.

Object detection is a computer vision technique for locating parts of objects in images or videos. Object detection algorithms typically anchorage machine learning or deep learning to construct meaningful consequences. Humans can quickly identify and find objects of interest while looking images or movies. The goal of object detection is to recreate this intelligence applying a computer [10].

Popular deep learning–based approaches for object detection using convolutional neural networks (CNNs), such as Faster R-CNN, YOLO or SSD automatically train to detect objects within images or videos [11]. Neural networks are a subset of machine learning and deep learning algorithms. They are composed of node layers, including an input layer, one or more hidden layers, and an output layer. Every node has a weight and threshold and is linked to every other node. Any node is stimulated and provides data to the network's next layer if its output exceeds the designated threshold value [12].

There are other kinds of neural networks that are occupied for different kinds of data and application. Recurrent neural networks, for example, are frequently occupied in voice recognition and natural language processing (NLP), whereas convolutional neural networks, also recognized as CNNs, are more frequently applied in computer vision (CV) and classification [13]. Before CNNs, objects in images were recognized using manual feature extraction techniques. Convolutional neural networks now provide a more scalable method for image classification and object identification. However, because they select graphics processing units (GPUs) to learn models, they can be computationally demanding [14].

3.2 Faster R-CNN

Although Fast R-CNN overcame some problem of the R-CNN, the region of proposals was still computed with the Selective Search algorithm and the inference time was still truly slow. So to overcome these problems, Faster R-CNN came up with the concept of Region Proposal Networks (RPNs) [15].

Faster R-CNN is object detection architecture with convolution neural networks (CNNs). It was introduced by Ross

Girshick, Shaoqing Ren, Kaiming He, and Jian Sun in 2015 [16].

The main sections of Faster R-CNN are the Region Proposal Network known as RPN and Classification Module to get the predicted class labels and bounding box locations [15].

3.2.1 Convolution Layers

Convolution layers are used to train filters to extract the appropriate features from an image. Convolution layers, pooling layers, and a final fully connected layer or other extended component that will be applied for a suitable process, such as classification or detection, are the typical parts of convolution networks. The first convolution layer provides the bounding box regressor outputs to predict the offset and scales, which are to be applied over the image anchors to refine the predictions [17].

3.2.2 Residual Network (ResNet)

This architecture imported the idea of Residual Blocks to address the vanishing gradient problem. RseNet uses a method known as skip connections. By excluding some layers in between, the skip connection links a layer's activations to subsequent layers. A residual block is created as a result. These unused blocks are stacked to build ResNet [18].

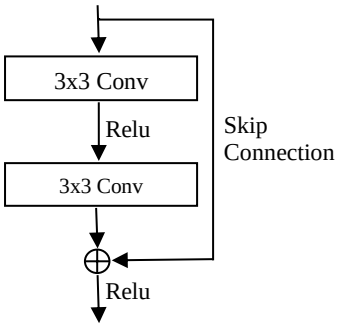


Figure 1. Residual block of ResNet

3.2.3 Region Proposal Network

The generated image anchors from the previous phase must provide a decent set of recommendations to the RPN. The RPN uses the feature map that was produced by the feature extractor and runs it through two parallel convolution layers [16].

The outputs of the bounding box regressor are given by the first convolution layer. These outputs are designed to predict the offset and scaling that will be applied to the image anchors in order to refine the predictions, rather than to identify the accurate positions of the bounding boxes themselves. Additionally, the second layer produces a classification output that displays the likelihood that the bounding boxes are background or foreground [15].

Potential bounding box candidates where an article can be discovered are call anchors. Before learning starts, a predetermined set of boxes is positioned throughout the image using a variety of aspect ratios and scales. Faster R-CNN produces 3*3 = 9 Anchor Box arrangements by applying 3 aspect ratios and 3 scales [16].

Positive anchor boxes include an object and negative anchor boxes do not. To sample positive anchor boxes, the anchor boxes that have an IoU of more than 0.7 with any of the ground truth boxes are selected. To compute IoU, we use the area of intersection and the area of union, where the formula of IoU will simply be:

$$\text{IoU} = \frac{\text{Area of intersection}}{\text{Area of union}} \quad (1)$$

The region proposal network applies the feature map to create area suggestions. In this case, the region proposal network is produced by combining the backbone network, the sample module, and the proposal module. During both training and inference, the RPN provides scores and offsets for all the anchor boxes. However, during training, the positive and negative anchor boxes are selected to determine classification loss. Loss function for classification is used Binary Cross entropy Loss [19].

$$\text{BCE} = \frac{1}{N}\sum_{i=1}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i) \quad (2)$$

Where N is the number of observations, $y_{i,}y_i$ is the actual binary label (0 or 1) of the $i^{th}i^{th}$ observation and p_ip_i is the predicted

probability of the ithith observation being in class 1.

During inference, the anchor boxes with scores above a given threshold are selected and produce proposals using the predicted offsets. A sigmoid function is used to transform the raw model logits to probability scores [17].

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (3)$$

Where x is the input value and e is Euler's number (approximately 2.718).

3.2.4 Classification Module

The Classification Module in Faster R-CNN can be done by a simple convolutional network with fully connected layer. All proposals from RPN and CNN are fed into the classification detector. The proposals are divided into roughly same subregions (they may not be equal) and applied a max pooling operation on each of them to produce outputs of same size. This is called ROI pooling. To compute the ground truth offsets/scales, the selected RoI is encoded with its correlated maximum IoU Ground Truth Bounding Boxes (GTBb). This calls finding the GTBb with the highest IoU among all GTBbs for that particular RoI. The predicted offsets/scales are used to the selected RoIs, potentially increasing their accuracy [17].

4. DATA PREPROCESSING

Automated human detection in images and videos contains a complex challenge due to diverse environmental conditions, variations in appearance, and the presence of occlusions. To address the challenges posed by varying environmental conditions, image enhancement techniques such as bilateral filter and gamma correction are implemented to improve detection accuracy [20]. There are two steps involved in data preprocessing step using Gamma Correction and Bilateral Filter.

4.1 Gamma Correction

Gamma correction is the process of adjusting the brightness of an image so colours display correctly on-screen. The following equation is used for gamma correction.

$$\text{Gamma} = 255 \times \left(\frac{\text{Pixel Value}}{255}\right)^{\frac{1}{\gamma}} \quad (4)$$

Where Pixel Value is original pixel intensity (0–255) and γ (gamma) is the gamma value you choose (e.g., 2.2 for typical display correction).

4.2 Bilateral Filter

A bilateral filter is edge-preserving and noise-reducing smoothing filter for images. The following equation is used for bilateral filter.

$$BF[I]_{(p)} = \frac{1}{Wp}\sum_{q \in s} G\sigma_s(||p-q||)G\sigma_r(|I(p)-I(q)|) \times I(q) \quad (5)$$

Where σ_s denotes the spatial extent of the kernel, σ_r denotes the minimum amplitude of an edge, I(p) denotes intensity at pixel p $G\sigma_s$ is the spatial Gaussian kernel with standard deviation σs\sigma_sσs (controls spatial extent of the filter), $G\sigma_r$ is the range Gaussian kernel with standard deviation σr\sigma_rσr (controls how strongly edges are preserved), p is the center pixel, and q are the neighboring pixels in the spatial window [21].

5. DATASET DESCRIPTION

Human detection dataset from Kaggle.com consists of 17,754 images and 17,754 annotation in YOLO format files. The dataset was constructed by using the OIDv4 toolkit, a particular tool for gathering data from Google Open Images.

This dataset is an invaluable resource for training deep learning models for human detection tasks. The images in the dataset are of high quality and have been exactly annotated with bounding boxes including the regions where humans are present. Accordingly, it is an appropriate choice for training deep learning models specifically designed for human detection tasks.

6. IMPLEMENTATION

Human detection system in this study utilized Faster R-CNN with Feature Pyramid Network (FPN) and both ResNet 50 and

ResNet 101 backbones pretrained on ImageNet. Human Detection Dataset from Kaggle was used for model training and testing. Images of this dataset contain person class. This system used 80% of dataset for training and 20% of dataset for testing. The following diagram illustrates the system flow of the applied method.

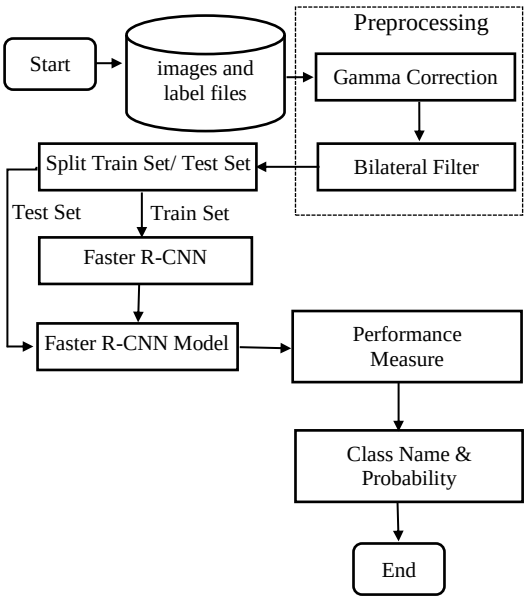


Figure 2. System Flow of human detection System

The following figure shows the detection results of the system tested using the No-Preprocessing method with ResNet-101 in the Faster R-CNN framework at 30 epochs, where the score threshold was set to 0.5, the IoU threshold to 0.3, and the human detection limit to 5.

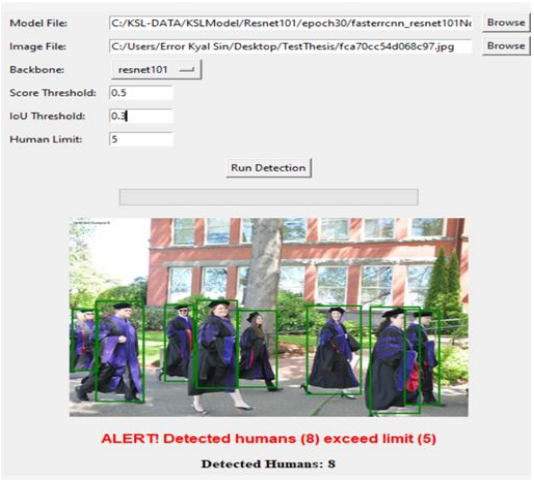


Figure 2. Result for Detect and Alert System

7. EXPERIMENTAL RESULT

This system was used Faster R-CNN model for human detection. This model trains on Nvidia Geforce RTX 4090 laptop GPU 16GB, Ram 64 GB, CPU core i9 with 80% human detection dataset.

The system was trained a Faster R-CNN model for human detection using a system implemented with an NVIDIA GeForce RTX 4090 Laptop GPU with 16 GB of memory, 64 GB of RAM, and an Intel Core i9 processor. The model was trained on a human detection dataset, using 80% of the data for training.

Two separate experiments were guided using 10 and 30 epochs, respectively. A batch size of 2 was applied throughout the training process, along with a learning rate of 0.001 to prevent overfitting. This implementation was chosen to calculate the model's performance under different training time while maintaining consistent optimization parameters.

This system was used precision, recall, F1 score for performance measure. The mean Average Precision, or mAP score, is computed by taking the mean AP over all classes or overall IoU thresholds. This setup ensured a balanced evaluation of model performance, highlighting the trade-off between training duration and detection accuracy across varying experimental conditions.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (6)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (7)$$

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} * \text{Recall}} \quad (8)$$

TP = True Positive (Detected humans that are actually humans)

TN = True Negative (Detected not humans that are not actually humans)

FP = False Positive (Detected humans that are not actually humans)

FN = False Negative (Real humans not detected by the model)

Performance measures of model shows below the table and the bar chart.

Table 1. Comparison of Faster R-CNN Performance Using ResNet50 and ResNet101 with Different Preprocessing Methods (Score Threshold = 0.75, IoU Threshold = 0.1)

Preprocessing Method	Faster R-CNN (ResNet50)			
	Epoch	Average Precision	Recall	F1 Score
Bilateral Only	10	0.688	0.550	0.612
	30	0.624	0.605	0.614
Bilateral + Gamma (0.9)	10	0.707	0.507	0.590
	30	0.766	0.445	0.563
No Preprocessing	10	0.728	0.499	0.592
	30	0.653	0.577	0.613
Preprocessing Method	Faster R-CNN (ResNet101)			
	Epoch	Average Precision	Recall	F1 Score
Bilateral Only	10	0.796	0.415	0.546
	30	0.730	0.442	0.551
Bilateral + Gamma (0.9)	10	0.675	0.544	0.603
	30	0.718	0.489	0.582
No Preprocessing	10	0.781	0.446	0.568
	30	0.757	0.460	0.573

The following bar charts illustrate the Average Precision, Recall, and F1 Score comparisons for different preprocessing methods — Bilateral Only, Bilateral + Gamma (0.9), and No Preprocessing — using Faster R-CNN with ResNet-50 and ResNet-101 backbones at 10 and 30 epochs. These comparisons provide a clear visualization of how different preprocessing strategies and backbone depths influence detection accuracy and model stability across training epochs.

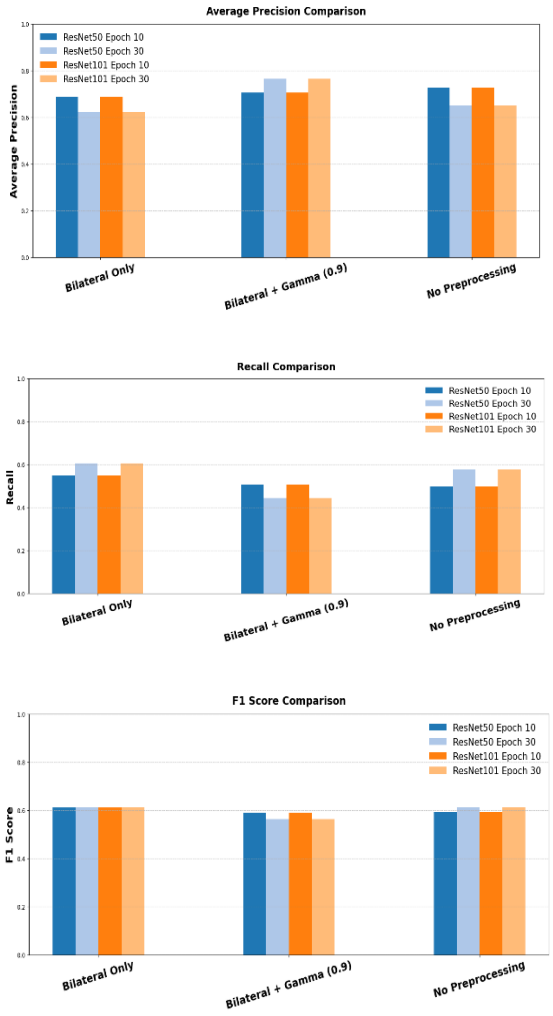


Figure 3. Average Precision, Recall and F1 Score of human detection System

8. CONCLUSION

In this study, the performance of Faster R-CNN with ResNet50 and ResNet101 backbones was calculated under different preprocessing techniques (Bilateral filtering, Bilateral with Gamma value 0.9, and no preprocessing) across 10 and 30 training epochs. For ResNet50, the best performance was achieved with Bilateral and Gamma 0.9 preprocessing at 30 epochs, with an AP of 0.766, while Bilateral-only preprocessing at 30 epochs obtained the highest F1 score (0.614). This indicates that ResNet50 benefits from preprocessing strategies that enhance both edge information and contrast, providing balanced precision and recall. In contrast, ResNet101 consistently achieved higher AP values, with the maximum

(0.796) observed at 10 epochs with Bilateral-only preprocessing. Overall, ResNet50 provided more stable and balanced detection performance, whereas ResNet101 favoured precision at the expense of recall.

ACKNOWLEDGEMENT

I would like to take this opportunity to show my sincere thanks to my all teachers who gave me many valuable advice and information. I am also graceful to all respectable people who directly or indirectly contributed towards the success of this paper.

I would like to show my respectful gratitude to Dr. San San Tint, Pro-Rector of the University of Computer Studies (Mandalay), for allowing me to develop this thesis and giving me general guidance during the period of study.

I am deeply grateful to my supervisor, Dr. Thein Htay Zaw, Associate Professor, Department of Information Technology Supports and Maintenance, University of Computer Studies, (Mandalay), for his close guidance, helpful advice, numerous invaluable suggestions, patience and support she has provided throughout my time.

Thanks, are also extended especially to all my teachers, my parents, seniors, friends and staff members of the University of Computer Studies (Mandalay) for the contribution towards the completion of this thesis.

REFERENCES

[1] Meeta Chaudhry, Saurabh Maurya, Shiv Pratap Singh, Shubham Yadav, "An Analysis of Deep Learning-Based Studies on Object Detection", Technological Advancements in Computational Sciences (ICTACS).

[2] Sapkota, R., Flores-Calero, M., Qureshi, R. et al. YOLO advances to its genesis: a decadal and comprehensive review of the You Only Look Once (YOLO) series. Artif Intell Rev 58, 274 (2025).

[3] W. Shao, W. Yang, G. Liu, and J. Liu, “Car detection from high-resolution aerial imagery using multiple features,” in Proc. IEEE Int.Geosci. Remote Sens. Symp. (IGARSS), Jul. 2012, pp. 4379–4382

[4] A. L’Heureux, K. Grolinger, H. F. Elyamany and M. A. M. Capretz, "Machine Learning With Big Data: Challenges and Approaches," in IEEE Access, vol. 5, pp.

[5] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521, no. 7553 (2015): 436-444.

[6] Chen, Leiyu, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. 2021. "Review of Image Classification Algorithms Based on Convolutional Neural Networks" Remote Sensing 13, no. 22: 4712.

[7] Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M. and Farhan, L., 2021. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of big Data, 8(1), p.53.[8] Zou, Zhengxia, et al. bjectdetection in 20 years: A survey." Proceedings of the IEEE 111.3 (2023).

[9] Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) (Vol. 1, pp. 886-893). Ieee.

[10] Papageorgiou, Constantine P., Michael Oren, and Tomaso Poggio. "A general framework for object detection." In Sixth international conference on computer vision (IEEE Cat. No. 98CH36271), pp.

[11] Coşkun, Musab, Özal Yıldırım, Ayşegül Uçar, and Yakup Demır. "An overview of popular deep learning methods." European Journal of Technique (EJT) 7, no. 2 (2017)

[12] Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Campbell, J. P. (2020). Introduction to machine learning, neural networks, and deep learning. Translational vision science & technology, 9(2), 14-14.

[13] Yao, Xin. "A review of evolutionary artificial neural networks." International journal of intelligent systems 8, no. 4 (1993): 539-567.

[14] Turay, Tolga, and Tanya Vladimirova. "Toward performing image classification and object detection with convolutional neural networks in autonomous driving systems: A survey." IEEE access 10 (2022): 14076-14119.

[15] Girshick R. Fast r-cnn. InProceedings of the IEEE international conference on computer vision 2015 (pp. 1440-1448).

[16] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems 28 (2015).

[17] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks." IEEE transactions on pattern analysis and machine intelligence 39, no. 6 (2016).

[18] Koonce, Brett. "ResNet 50." In Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization, pp. 63-72. Berkeley, CA: Apress, 2021.

[19] Ruby, U., & Yendapalli, V. (2020). Binary cross entropy with deep learning technique for image classification. Int. J. Adv. Trends Comput. Sci. Eng, 9(10).

[20] Kansal, Shubhi, and Rajiv Kumar Tripathi. "Adaptive gamma correction for contrast enhancement of remote sensing images." Multimedia Tools and Applications 78, no. 18 (2019): 25241-25258.

[21] Elad, Michael. "On the origin of the bilateral filter and ways to improve it." IEEE Transactions on image processing 11.10 (2002): 1141-1151.

IMAGE REFLECTION REMOVAL USING DEEP CONVOLUTIONAL ENCODER-DECODER NETWORK

Aye Su Khaing[1], and Mya Thidar Kyaw[2]

[1,2] Department of Information Science, University of Computer Studies (Mandalay)
ayesukhaing122mdy@gmail.com, myathidarkyawmin@gmail.com

ABSTRACT

Photographs captured through glass or other reflective surfaces often contain both the desired scene and unwanted reflections. Even though reflections create beautiful effects in some circumstances, they are usually an annoyance when people wish to view the thing behind the glass clearly. They can significantly degrade the image quality and they also interfere with the performance of various computer vision tasks. Consequently, the task of separating the reflection layer from the transmission layer has emerged as a prominent. Deep learning-based techniques have been utilized extensively for reflection elimination in recent years. To address this problem, this system employs a deep learning-based convolutional encoder-decoder network by learning the difference between the two images with and without reflections. In contrast to the majority of deep learning techniques now in use in this field, this network predicts the reflection-free layer by using both a mixed reflection image and the target edge as input. To train the neural network, this system uses synthetic dataset. The Structural Similarity Index Measure (SSIM) and the Peak Signal-to-Noise Ratio (PSNR) are employed to evaluate the performance of this system.

KEYWORDS

Deep learning, Encoder-decoder network, VGG19, Reflection removal, Computer vision

1. INTRODUCTION

With the increasing use of digital cameras and smartphones to capture daily moments, digital photography has become a routine part of life for many people. Even though a new and more sophisticated digital cameras and smartphones are released to the market every few months, some imaging problems still cannot be fully solved. The process of getting rid of reflection is one of them [1]. The goal of reflection removal is to eliminate undesired reflections in order to restore the target image. The reflected image can be expressed in theory like this:

$$I = B+R \quad (1)$$

where B stands for the target image, R for the reflected image, and I for an image that has a reflection [2].

2. RELATED WORD

Image reflection removal is a highly ill-posed challenge, and numerous solutions have been introduced by researchers. Based on the number of input images used, reflection removal methods can be classified into two categories:

- Multi-image based approaches
- Single-image based approaches

Multiple-image-based methods eliminate reflections by utilizing two or more input images of the same scene with different viewpoints to reconstruct the target scene. Reflection removal techniques using multiple images can be divided into four distinct groups: such as video sequences [3]; flash and non-flash image pairs [4]; multiple polarized images [5]; and focus/ defocus image pairs [6].

Many previous studies have focused on recovering the background scene using a single image rather than multiple input images as it is easier to implement. Levin et al. [7] proposed an unsupervised approach by minimizing edges and corners in the decomposed layers of the input image. Levin et al. [8] developed an interactive approach where users manually mark edges

belonging to reflections and background layers, using these annotations as priors for optimization.

3. REFLECTION REMOVAL SYSTEM

3.1 System Architecture

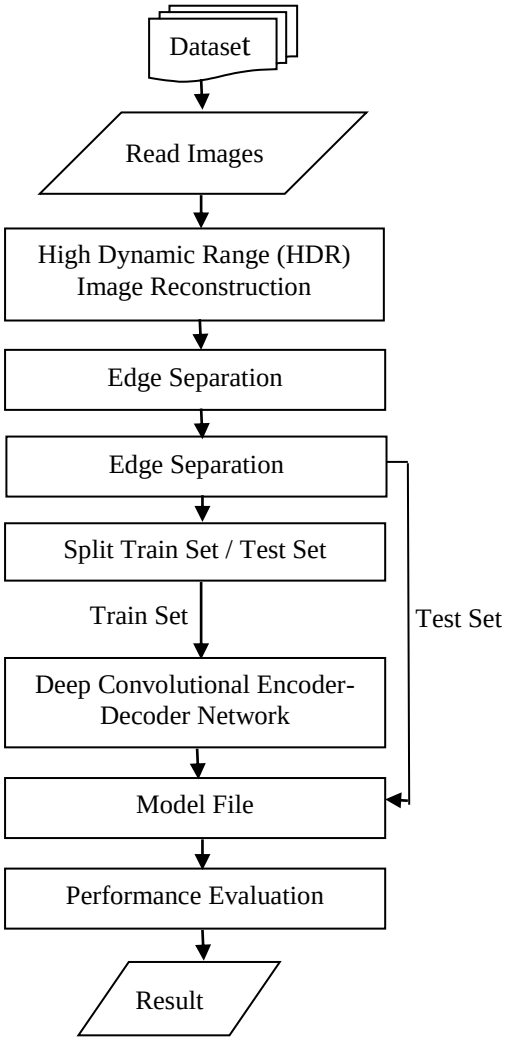


Figure 1. System Architecture

3.2 Preprocessing

A. Bias Parameter (b) Selection

In the Drago Tone Mapping effect, you can adjust bias parameter. The bias parameter b plays a key role in controlling the compression of bright regions while enhancing the visibility of details in darker areas.

B. Gamma (γ) Selection

Gamma correction is applied to the tone mapped image to make the image look natural to human eyes and suitable for the non-linearity of displaying devices.

Figure 2 shows the effects of changing b and r. The system tested three gamma values (0.5, 0.7, 0.9) and three bias values (0.75, 0.85, 0.95) to find the best ones for HDR image reconstruction. The best result was γ = 0.7 and b = 0.95, which gave the highest PSNR.

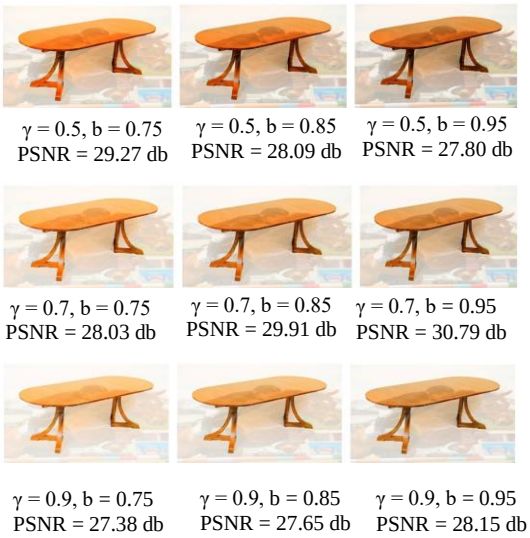


Figure 2. HDR image results under different b and r settings

C. Threshold (T) Selection

In order to accurately separate the target edge, an optimal threshold value must be chosen for each image. Additionally, this system chose 13700 photos and ran threshold calculations on them in order to determine a suitable threshold for edge separation of image sets.

An edge separation test with thresholds of 75, 95, and 105 was performed on the edge probability pictures; the outcomes are displayed in Figure 3. The target edge with thresholds T = 75 is displayed in the first row; the target edge with thresholds T = 95 is displayed in the second row; and the target edge with threshold value T = 95 is displayed in the third row.

As illustrated in Figure 3, when T=75, the target layer retains excessive reflective edge information. On the other hand, with T=105, important edge details from the

target layer are lost. Based on the visual quality, the threshold value is set to T=95, as it produces clear target edges without interference from strong reflections.

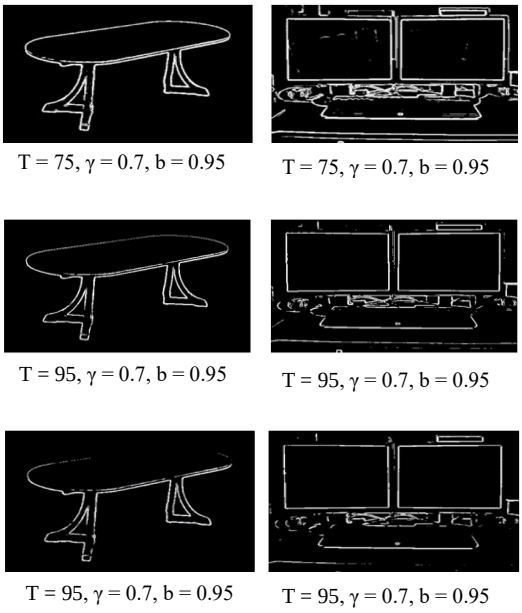


Figure 3. Comparisons of edge outputs across different threshold levels

4. METHODOLOGY

Deep learning techniques have been demonstrated the effectiveness and superior performance in numerous computer vision and image restoration tasks. Rapid advances in deep learning have made it easier to handle the challenge of removing reflections from photographs. This system uses single-image reflection removal method based on deep convolutional encoder-decoder network. Due to the nature of reflection images, the distribution of the reflection layer is more sparse than that of the target layer.

Based on this observation, this system reconstructs the target layer using a depth encoder and decoder network based on the target edge. The encoder, combined with edge information, helps to extract more precise semantic features, while the decoder uses deconvolution to generate the target layer. Simultaneously, to further improve reflection removal efficiency, a residual block network is inserted between the encoder and decoder. In brief, this system involves three main steps:

- Reconstruct a High Dynamic Range (HDR) image to improve the quality of the mixed reflection image.
- Utilize an edge separation technique to extract the target edges.
- Combine the HDR mixed reflection image with the extracted target edge and then feed it into the network, which then outputs a reflection-free image.

4.1 High Dynamic Range (HDR) Image Reconstruction

In the first phase, this system reconstructs HDR images because normal cameras or standard images cannot capture both very dark and very bright areas at the same time. As a result, details in shadows or highlights often get lost. Drago's Global tone mapping is used to preserve visual details, keep the image looking natural, and maintain quality by adjusting extreme brightness levels. The displayed luminance is set to L_{dmax} = 100 cd/m², which is a standard reference value commonly used for CRT displays [9].

4.2 Edge Separation

This paper employs a first-order differential edge operator to extract target edge from the original image. Edges in the reflection layer generally correspond to smaller gradient magnitudes, whereas edges in the target layer are stronger but sparser. Sobel edge detection is a widely used method in image processing for identifying edges in an image. It is a gradient-based technique that applies two 3x3 convolution kernels (filters). The two Sobel filters are:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Horizontal kernel Vertical kernel

And then the resulting gradient values are combined to measure edge strength. A threshold value is applied to the gradient magnitude image to classify pixels as background edges or non-edges. Pixels with gradient magnitude above the threshold are considered background edges.

4.3 Deep Convolutional Encoder-Decoder Network

This paper emphasizes the third phase, which applies a deep convolutional encoder–decoder network. Under the guidance of the target edge, the target picture is produced after the target edge and the original image are fed into the network structure. The deep convolutional encoder-decoder network structure diagram is displayed in Figure 5.

Specially, the network architecture is composed of three main components: an encoder, a residual block, and a decoder. In this structure, the encoder is designed to capture high-level representations from the input image. The convolutional layers of the VGG19 network was chosen for feature extraction within the encoder because they performed better than the other models in terms of similarity calculations and feature extraction quality [10].

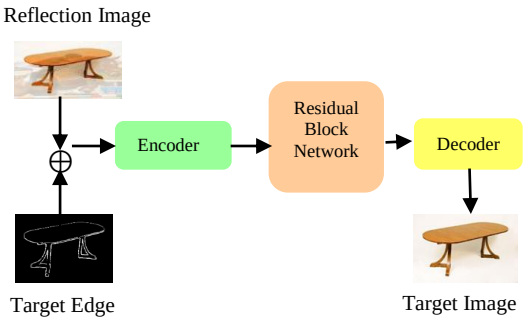


Figure 5. Network diagram of deep convolutional encoder-decoder

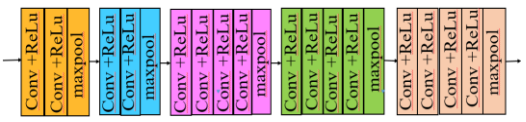


Figure 6. VGG19 model's feature extraction layers

Convolution layer: The initial layers extract basic features like edges, textures, or colors, while deeper layers extract more complex shapes. It is the process of sliding a convolution kernel over the input image performing element-wise multiplication and summation them to produce feature maps. Kernel moves on the input data by the stride value.

Max pooling: It selects the maximum value from each 2×2 region of the image, and placed in the corresponding position in the output feature map. It helps to retain the most important features.

Nonlinear activation layer ReLU: It is a linear function defined as $F(x) = \max(0, x)$, which sets negative values to zero while keeping positive ones, helping the network learn sparse features and speed up training.

The residual block network is the intermediate structure. For many image restoration tasks, residual learning works well which helps to increase the network model's efficiency without altering the output image's size [11]. The residual block network is composed of three residual blocks, each consisting of a convolutional layer, a normalization layer, and a ReLU activation layer.

Normalised layer: To reduce internal covariate shifts, batch normalization [12] was suggested. It normalizes inputs to have mean 0 and variance 1, keeping outputs stable during training and reducing internal covariate shifts.

The decoder network is designed to mirror the encoder. It is specifically composed of five upsampling layers. To restore the original resolution during reconstruction, the decoder is designed with five deconvolution (upsampling) layers. Each deconvolution layer corresponds to one pooling operation in the encoder, ensuring a symmetric architecture that allows the network to recover fine details and produce high-quality outputs.

Deconvolution layer: Zeiler et al. [13] were the first to introduce the idea of deconvolution. Deconvolution operates similarly to convolution, but its function is to use reverse training to create the effect of recreating input from output.

The image goes through the encoder after pre-processing. The residual network then processes the image, without altering its size. Finally, the image is sent into the decoder network and where it is gradually restored to its original size through five upsampling layers.

Loss function: This system computes the target layer prediction loss, which measures the discrepancy between the network's predicted target image and the ground truth target image. The mean square error (MSE), which is the average of the square of error, is typically used as a loss function to approximate it. The formula for MSE is shown below.

$$L(B, B') = \frac{1}{n}\sum_{i=1}^{n}(B - B')^2 \quad (2)$$

where B′ is the predicted target image's pixel value, B is the actual target image's pixel value, and n stands for the number of image's pixels. A lower MSE score indicates fewer discrepancies and more accurate predictions. MSE is typically employed in network training as a loss function.

5. Evaluation Standard

Many reflection removal algorithms use quantitative evaluation to assess their effectiveness. The experimental results are assessed using the Peak Signal-to-Noise Ratio (PSNR) and the Structural Similarity Index (SSIM) [14].

PSNR is a metric used to evaluate image quality between the network's predicted target image and actual target image. PSNR is mathematically related to the MSE.

$$PSNR = 10\log_{10}\frac{255}{\sqrt{|B-B'|^2}} \quad (3)$$

where B denotes the real target image, while B′ refers to the predicted target image. A higher PSNR value indicates lower distortion and better image quality.

SSIM is a widely used metric for assessing the similarity between two images. The range of SSIM values is [-1,1]. SSIM formula:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4)$$

6. Experiment

6.1 Dataset

Training deep neural networks requires a large amount of data; however, gathering real images with ground truth background layers are hard to obtain and time-consuming. The dataset built by Xue et al. [15] is relatively small, and with the increasingly applied of deep learning for reflection removal, the demand for larger datasets continues to increase. As deep learning becomes increasingly applied to reflection removal, the need for larger datasets also grows. Therefore, the use of synthetic image datasets has started to be considered in existing methodologies [16]. In fact, synthetic dataset provides sufficient data to build effective models.

In [17] synthesizes an image that is closely resemble to the reflection of the actual world, assuming that the reflection layer is a little blur compared to the background layer, which is usually sharper and clearer. Based on this assumption, images that closely resemble real-world reflections are synthesized, since cameras typically focus on the background target. In this study, this system generates synthetic images through a simple linear combination, expressed as:

$$I = \gamma B + \beta R \quad (5)$$

where B denotes the background image, R represents the reflection image, and r and β are coefficient values.

To generate sufficient training data, this system creates synthetic images by mixing two images with different coefficients (0.8 for the background and 0.2 for the reflection). In fact, this approach has been extensively employed for analysis and quantitative assessment in earlier research [18]. For our dataset, we use the PASCAL VOC database [19], which contains 17,125 images. This system utilizes 13700 of the color images from PASCAL VOC 2012 to synthesize the training data, where each image is combined with randomly chosen reflection layers. The testing phase was performed on SIR^2 benchmark dataset [20] to comprehensively evaluate the model's reconstruction capability.

6.2 Network Training

After training the network using the loss function defined in Section 4.3, this system was able to compute the predicted target layer loss. Through backpropagation, the

gradients of the network parameters are calculated and updated accordingly. This system first trains the network for 50 epochs, after which the learned weights are used to initialize a new network. This new model is then trained for an additional 100 epochs using the dataset described in Section 6.1.

The initial learning rate is set to 0.0001, with a batch size of 4. Training requires approximately 40 hours on an NVIDIA GeForce RTX 4090 Laptop GPU (16 GB VRAM), with 64 GB RAM and an Intel Core i9 processor.

6.3 Results and Discussion

The effectiveness of suggested system is analyzed by measuring the evaluation criteria on the image reconstruction. SSIM and PSNR are used to assess the system's performance. In this section, the following tables showed SSIM and PSNR results of SIR^2 benchmark dataset [20] in test set depend on the epochs.

Epochs100	SIR^2 - Solid		SIR^2 - Postcard		SIR^2 - Wild	
	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
CEILNet [21]	0.831	23.17	0.800	19.98	0.794	20.84
BDN [22]	0.830	22.70	0.849	20.54	0.825	22.00
PL [23]	0.775	22.14	0.597	15.80	0.828	21.15
ERR [24]	0.868	24.77	0.866	21.81	0.865	23.73
GCNet [2]	0.928	23.87	0.918	19.64	0.932	24.97
Ours Epochs50	0.954	31.64	0.908	29.79	0.926	29.89
Ours Epochs100	0.960	32.60	0.927	30.41	0.936	30.15

Table 1. Performance comparison of the system with epochs50 and epochs100 on SIR^2 dataset

The performance results on the SIR^2 benchmark dataset demonstrate that deep convolutional encoder-decoder network significantly outperforms existing state-of-the-art methods, including CEIL Net [21], BDN [22], PL [23], ERR [24], and GCNet [2]. In both training settings (50 and 100 epochs), this system achieves the highest SSIM and PSNR values across all categories (Solid, Postcard, and Wild), showing its superior ability to restore clear background images with fewer artifacts. At 50 epochs, the SSIM scores for Solid, Postcard, and Wild are 0.954, 0.908 and 0.926 with PSNR values of 31.64, 29.79, and 29.89. When training is extended to 100 epochs, SSIM further improves to 0.960, 0.927 and 0.936, with higher PSNR values of 32.60, 30.41 and 30.15.

7. CONCLUSION

This system employs a deep convolutional encoder-decoder network approach to address the problem of reflection removal. The framework is based on an encoder–decoder structure, enhanced with a residual learning module to improve the learning efficiency, leading to the significant performance gains. In addition, this approach incorporates HDR image reconstruction techniques to preserve details in both dark and bright regions, which is crucial for producing high-quality reflection-free images. By combining encoder–decoder network with edge information, this method effectively eliminates unwanted reflections. Qualitative results show that this method significantly enhances image details, producing outputs that closely resemble the original image in terms of visual perception.

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to all my teachers for their invaluable advice and guidance throughout this journey. I am also truly thankful to everyone, both directly and indirectly, who contributed to the successful completion of this thesis.

Foremost, I would like to extend my respectful gratitude to Dr. San San Tint, Rector of the University of Computer Studies (Mandalay), for permitting me the opportunity to conduct this research.

Besides, I would also like to thank, Dr. Aye Aye Chaw, Professor and Dean of Master Course, Professor at Faculty of Computer Science, University of Computer Studies, (Mandalay), for her valuable suggestions.

Most importantly, I am deeply grateful to my supervisor, Dr. Mya Thidar Kyaw, Professor, Department of Information Science, University of Computer Studies (Mandalay), for her continuous guidance

and support throughout my M.C.Sc research.

Especially, I need to appreciate to Daw Su Myat Htet, Associate Professor, Head of English Department, University of Computer Studies (Mandalay), for her support in editing my thesis from a linguistic perspective.

Ultimately, I am truly thankful to all my teachers, parents, seniors, and friends who helped and supported me a lot in my research journey.

REFERENCES

[1] S. Saravanan, N. Iniyamozhi, N. Nandhini and P. Vinotha "Improved Multiple Image Based Reflection Removal Algorithm Using Deep Neural Networks",2021

[2] R.AbikoandM.Ikehara, Singleimagere ectionremovalbasedonGAN with gradient constraint, IEEE Access, vol. 7, pp. 148790148799, 2019, doi: 10.1109/ACCESS.2019.2947266

[3] Christian Simon and In Kyu Park, "Reflection removal for in-vehicle black box videos," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4231–4239.

[4] Amit Agrawal, Ramesh Raskar, Shree K Nayar, and Yuanzhen Li, "Removing photography artifacts using gradient projection and flash exposure sampling," ACM Transactions on Graphics (TOG), vol. 24, no.3, pp. 828- 835,2005.

[5] Shree K Nayar, Xi-Sheng Fang, and Terrance Boult, "Separation of reflection components using color and polarization," International Journal of Computer Vision, vol. 21, no. 3, pp. 163–186, 1997.

[6] Yoav Y Schechner, Nahum Kiryati, and Ronen Basri, "Separation of transparent layers using focus," International Journal of Computer Vision, vol. 39, no. 1, pp. 25–39, 2000.

[7] Levin, A., Zomet, A., Weiss, Y.: 'Separating reflections from a single image using local features'. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, Washington, DC, USA, 2004, pp. 306–313

[8] Levin, A., Weiss, Y.: 'User assisted separation of reflections from a single image using a sparsity prior', IEEE Trans. Pattern Anal. Mach. Intell., 2004, 29, (9), pp. 1647–1654.

[9] F. Drago, K. Myszkowski, T. Annen and N. Chiba, "Adaptive Logarithmic Mapping For Displaying High Contrast Scenes", 2003

[10] R. Mostafiz, M. Rahman, A. Islam, and S. Belkasim, 'Focal Liver Lesion Detection in Ultrasound Image Using Deep Feature Fusions and Super Resolution', Mach. Learn. Knowl. Extr., vol. 2, no. 3, pp. 172–191, Jul. 2020, doi: 10.3390/make2030010.

[11] Zhang, K., Zuo, W., Chen, Y., et al.: 'Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising', IEEE Trans. Image Process., 2017, 26, (7), pp. 3142–3155

[12] Ioffe, S., Szegedy, C.: 'Batch normalization: accelerating deep network training by reducing internal covariate shift'. Proc. of the 32nd Int. Conf. on Machine Learning, Lille, France, 2015, pp. 448–456

[13] Zeiler, M.D., Krishnan, D., Taylor, G.W., et al.: 'Deconvolutional networks'.IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, Piscataway, NJ, 2010, pp. 2528–2535

[14] A. Hore and D. Ziou, ''Image quality metrics: PSNR vs. SSIM,'' in Proc. 20th Int. Conf. Pattern Recognit., Aug. 2010, pp. 2366–2369.

[15] Xue, T., Rubinstein, M., Liu, C., et al.: 'A computational approach for obstruction-free photography', ACM Trans. Graph., 2015, 34, (4), pp. 1–11

[16] YiChang Shih, Dilip Krishnan, Fredo Durand, and William T Freeman, "Reflection removal using ghosting cues," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3193–3201.

[17] Fan, Q., Yang, J., Hua, G., et al.:"A generic deep architecture for single image reflection removal and image smoothing'. IEEE Int. Conf. on Computer Vision, Venice, Italy, 2017, pp. 3258–3267

[18] R. Szeliski, S. Avidan, and P. Anandan, "Layer extraction from multiple images containing reflections and trans parency. In IEEE Conference on Computer Vision and Pat tern Recognition (CVPR), volume 1, pages 246–253, 2000. 2, 4

[19] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and

Andrew Zisserman, “The pascal visual object classes (voc) challenge,” International journal of computer vision, vol. 88, no. 2, pp. 303–338, 2010.

[20] R. Wan, B. Shi, L.-Y. Duan, A.-H. Tan, and A. C. Kot, ‘‘Benchmarking single-mage reflection removal algorithms,’’ in Proc. IEEE Int. Conf. Comput. Vis., Oct. 2017, pp. 3922–3930

[21] Q. Fan, J. Yang, G. Hua, B. Chen, and D. Wipf, ‘‘A generic deep architecture for single image reflection removal and image smoothing,’’ in Proc. IEEE Int. Conf. Comput. Vis., Oct. 2017, pp. 3238–3247.

[22] J. Yang, D. Gong, L. Liu, and Q. Shi, “Seeing deeply and bidirectionally: A deep learning approach for single image reflection removal,’’ in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 654–669.

[23] X. Zhang, R. Ng, and Q. Chen, ‘‘Single image reflection separation with perceptual losses,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4786–4794.

[24] K. Wei, J. Yang, Y. Fu, D. Wipf, and H. Huang, ‘‘Single image reflection removal exploiting misaligned training data and network enhancements,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2019, pp. 8178–8187.

ENHANCING LOSSLESS DATA COMPRESSION EFFICIENCY FOR DNA SEQUENCES AND AUDIO

Zaw Win Myat[1], and Thin Thin Naing[2]

[1,2] Faculty of Computing, University of Computer Studies (Mandalay)
zawwinmyat.dev@gmail.com, thinnaing05@gmail.com

ABSTRACT

Data compression plays an important role in eliminating storage requirements and transmission costs. This study presents an integrated approach to lossless text compression using Huffman coding and Canonical Huffman coding. Huffman coding assigns variable-length codes based on symbol frequencies, achieving compact encoding of the input data. Canonical Huffman coding, a variant of Huffman coding, improves storage and transmission efficiency by reducing the overhead of representing code trees. With this methodology, the proposed scheme balances statistical modelling and entropy coding to maximize compression performance. The results demonstrate that the application of Huffman and Canonical Huffman coding achieves significant improvements in compression ratio while maintaining computational efficiency.

KEYWORDS

Huffman Coding, Canonical Huffman Coding, Lossless Data Compression

1. INTRODUCTION

The rapid growth of digital data in areas such as communication, multimedia, and bioinformatics has intensified the demand for efficient compression techniques. Lossless compression methods are particularly important when exact data reconstruction is required, as in DNA sequences and audio files. Traditional approaches such as Huffman coding and arithmetic coding have provided strong foundations for entropy-based compression. Canonical Huffman coding, a refinement of Huffman coding, improves storage and transmission efficiency by minimizing the overhead of representing code structures. This paper explores a framework that leverages Huffman and Canonical Huffman coding to enhance lossless compression efficiency for DNA and audio data, demonstrating how these methods reduce redundancy while ensuring accurate recovery of the original information.

The efficiency of lossless compression with Huffman Coding was enhanced through the use of Canonical Huffman Coding, which improves both storage and decoding efficiency by eliminating the need to store the tree structure and representing codes by their lengths.

2. METHODOLOGY

2.1. Huffman Coding

In information theory, Huffman coding is a lossless data compression algorithm which is based on the principle of variable length coding, where the common symbols are encoded with shorter codewords. It reduces the number of bits required to represent each symbol in a given data sequence. For any symbol distribution *P*, the Huffman algorithm encodes a code with the shortest possible expected code word length for a single symbol [2][5].

The Huffman coding algorithm contains two main phases: frequency analysis and code word generation which can be regarded as tree construction. In the frequency analysis phase, the frequency of each symbol in the input data sequence is calculated. Then, a binary tree is

constructed by repeatedly combining the two least frequent symbols into a new node whose weight equals the sum of their frequencies [3][4]. This process continues until only one root node remains. Figure 1 shows the flow of Huffman coding.

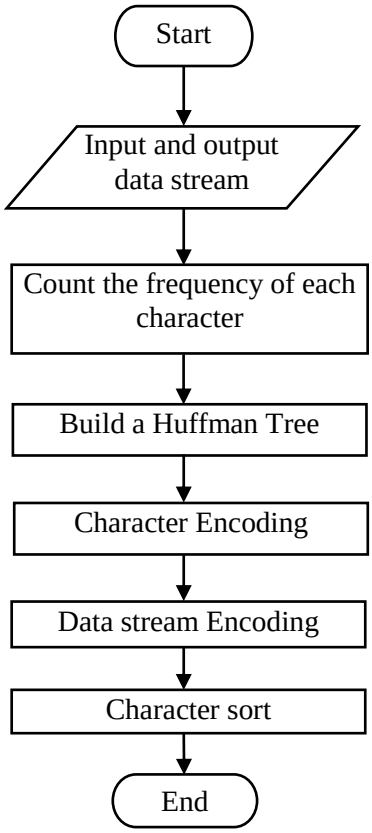


Figure 1. Huffman Coding Flowchart

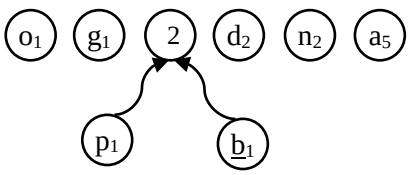
For instance, to encode input data, “ananda pagoda”, Assume that b is “ ” (Space).

Table 1. Frequency of Unique Symbols

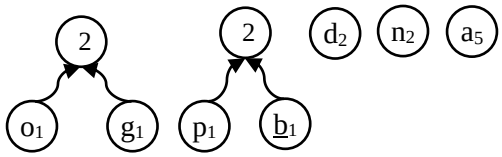
Symbol	a	n	d	p	g	o	b
Frequency	5	2	2	1	1	1	1

Then, sort in ascending order based on frequency and follow the algorithm the flow. Figure 2 shows the iteration steps of the Huffman coding algorithms.

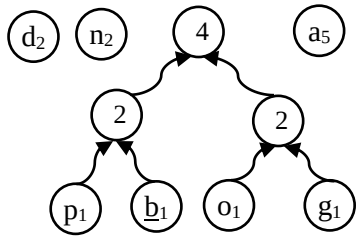
Iteration1



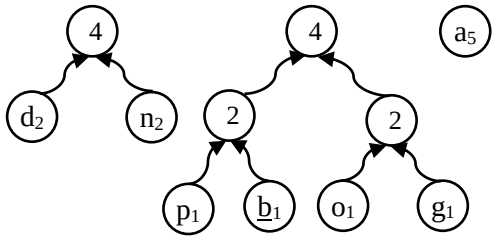
Iteration 2



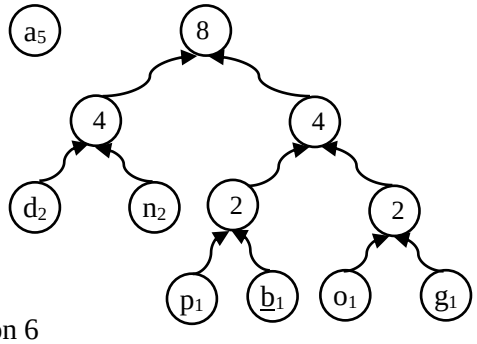
Iteration 3



Iteration 4



Iteration 5



Iteration 6

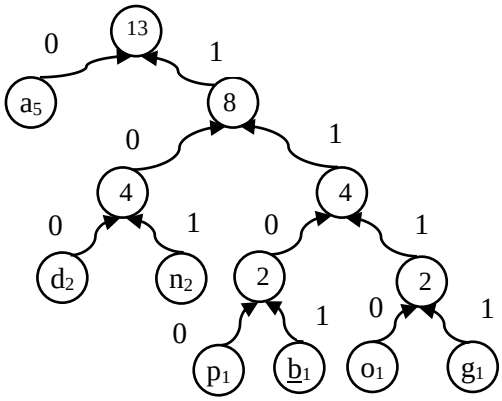


Figure 2. Encoding with Huffman Coding

A new node in the Huffman tree is formed by selecting the two nodes with the smallest frequencies and merging them into a single parent node. The frequency of this parent node is the sum of its two child nodes, and it represents the combined probability of their occurrence in the encoded data.

Table 2 shows the code assigned to each of the symbols presented in the tree. In the Huffman encoding, the symbols are replaced by the binary codes and transmit the codes instead of the symbols. Decoding at the receiver's end is relatively straightforward, requiring only the Huffman tree. The process begins by reading the encoded binary data stream sequentially from the root node of the tree and traversing downward until a leaf node is encountered. A binary 0 directs the traversal to the left child, whereas a binary 1 directs it to the right child. Upon reaching a leaf node, the corresponding character is identified and recorded. This process is repeated iteratively until the complete encoded sequence has been fully decoded [1].

Table 2. Code Words of "ananda pagoda"

Symbol	Code Word	Code Length
a	0	1
n	101	3
d	100	3
b	1101	4
p	1100	4
g	1111	4
o	1110	4

2.2 Canonical Huffman Coding

Canonical Huffman coding is a variant of traditional Huffman coding that provides a more compact and efficient representation of the codebook. While standard Huffman coding produces prefix-free codes, the structure of the resulting Huffman tree can vary depending on the way symbols with equal frequencies are arranged during construction. Canonical Huffman coding eliminates this variability by enforcing a consistent structure, which not only improves predictability but also reduces the storage overhead required for the codebook [4].

The steps involved in Canonical Huffman coding algorithm are stated as follow.

CanonicalHuffman(symbols, frequencies):

1. Build a Huffman tree from (symbols, frequencies)
2. For each symbol, record its code_length from the tree
3. Sort symbols by (code_length,symbol)
4. code = 0
5. prev_length = length of the firstsymbol
6. For each symbol in sorted order:
   If code_length > prev_length:
   code = code << (code_length – prev_length)

   Assign binary representation of 'code' (padded to code_length)

   Increment code by 1
   prev_length = code_length
7. Return mapping(symbol to canonical_code)

For instance, to encode for the sample data string, "ananda pagoda" with Canonical Huffman coding, sort the data according to bit length; then, sort the symbols lexicographically. Table 3 illustrates how it can be performed.

In the second step, assign the code of first symbol with the same number of '0' as the bit length. For symbol 'a', bit length is 1; therefore, a is 0. Afterward, next symbol, 'd', has bit length of 3 which is greater than the bit length of previous symbol 'a' which has length of 1; therefore, increase the code of previous symbol ,'a', by 1 (0 + 1 = 1) and append (3 -1 = 2 zeros) assign the code

to ‘d’. Afterward, d becomes 100. The workflow follows the algorithm, and the codewords illustrated in the Table 4 have been obtained.

Table 3. Sorted Symbols by bit length and lexicographic order

Symbol	Code Word	Code Length
a	0	1
d	100	3
n	101	3
b	1101	4
g	1111	4
o	1110	4
p	1100	4

3. DATASETS

3.1 DNA Sequence Datasets

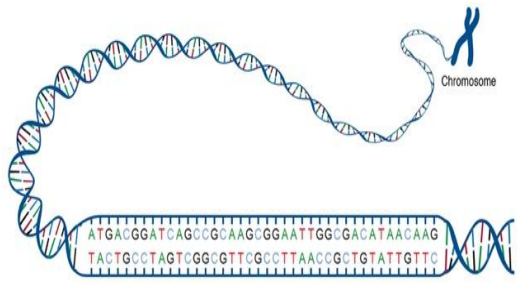


Figure 3. DNA Sequences

A DNA Sequence is the specific order of the four chemical bases: Adenine (A), Thymine (T), Cytosine (C), and Guanine (G) that form a DNA module. [4] DNA is a double helix made of two strands, and the bases are the building blocks of these strands. In the DNA double helix, adenine (A) always pairs with thymine (T), and cytosine (C) always pairs with guanine (G) [4].

Table 4. Code Words of Canonical Huffman Coding

Symbol	Code Word
a	0
d	100
n	101
b	1100
g	1101
o	1110
p	1111

The encoded data with Canonical Huffman is 010010111001101111011111. Therefore, the decoder only needs to know lengths and symbols, such as symbol ‘a’ has length of 1, symbol ‘d’, ‘n’ has length of 2 and symbol ‘b’, ‘g’, ‘o’, ‘p’ has length 4.

The following DNA Sequences for chimpanzee, dog and human is utilized.

Table 5. DNA Sequence Datasets

DNA Sequence File Names	Sizes
chimpanzee	3,258,260 bytes
dog	1,667,375 bytes
human	5,547,716 bytes

3.2 Audio Datasets

To compress audio files, Pulse Code Modulation, signed 16-bit little endian (pcm_s161e) is applied. All of the raw .wav audio files have sample rate of 44.1 kHz and both stereo and mono types are utilized.

In Figure 4, audio waveform of Burmese Harp is illustrated and it has duration of 6 minutes and 49.21 seconds with 1 channel, mono and bitrate of 705 KB/s. The uncompressed raw file size for burmese_harp.wav is 36,091,982 bytes.

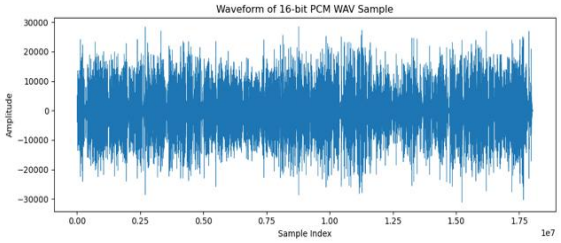


Figure 4. burmese_harp.wav's waveform

Also, in Figure 5, audio waveform of artist Htoo Eain Thin's song is illustrated and it has duration of 3 minutes and 25.47 seconds with 1 channel, mono and bitrate of 705 KB/s. The uncompressed raw file size for htoo_eain_thin.wav is 18,122,830 bytes.

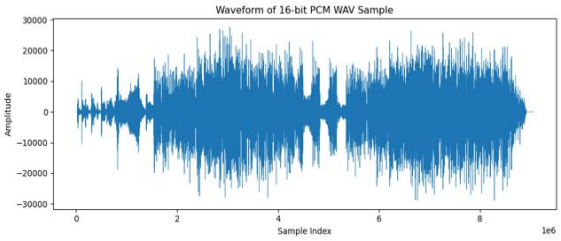


Figure 5. htoo_eain_thin.wav's waveform

In addition, in Figure 6, audio waveform of audio1_stereo.wav's is illustrated and it has duration of 2.94 seconds with 2 channels, stereo and bitrate of 1411 KB/s. The uncompressed raw file size for audio1_stereo.wav is 518,156 bytes.

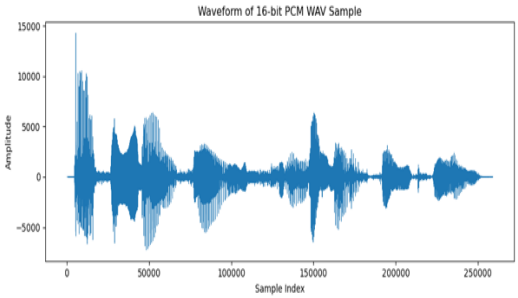


Figure 6. audio1_stereo.wav's waveform

And finally, in Figure 7, audio waveform of audio2_mono.wav's is illustrated and it has duration of 60 seconds with 1 channel, mono and bitrate of 705 KB/s. The uncompressed raw file size for audio2_mono.wav is 5,292,794 bytes.

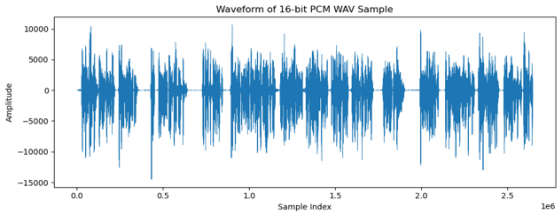


Figure 7. audio2_mono.wav's waveform

4. Experimental Setup And Performance Analysis

The experimental environment is uniformly configured as follow:

- Apple's MacBook Pro 2019 with macOS Sequoia Version 15.1.1 with Python 3.10.11
- 2.4 GHz Quad-Core Intel Core i5 Processor
- Memory of 8GB 2133 MHz LPDDR3 RAM

4.1 Performance Analysis between Huffman Coding and Canonical Huffman Coding on DNA Sequence

Tables 6 and 7 present the results of compressing DNA sequence data using Huffman Coding and Canonical Huffman Coding, respectively.

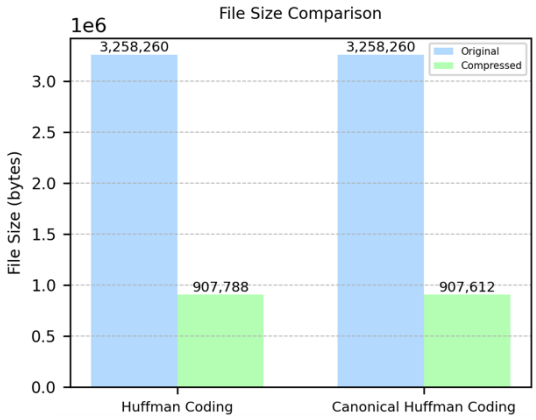


Figure 8. File Size Comparison for DNA Sequences Compression

For Huffman Coding, the compression ratios (without metadata) were consistent across all datasets, averaging around 0.278–

0.279, indicating that the method achieves nearly uniform compression efficiency regardless of sequence length. However, when metadata size (224 bytes) is considered, the effective compression ratio increased slightly (e.g., 0.27872 for human DNA), though the impact was negligible due to the relatively small metadata overhead compared to the large file sizes. The encoding and decoding times scaled proportionally with the dataset size, with the human DNA sequence requiring the longest encode (1.216 seconds) and decode (1.780 seconds) times.

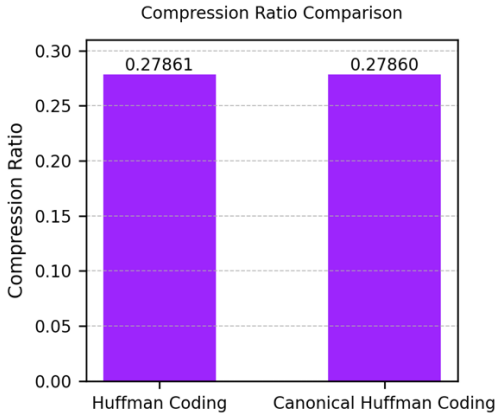


Figure 9. Compression Ratio Comparison for DNA Sequences

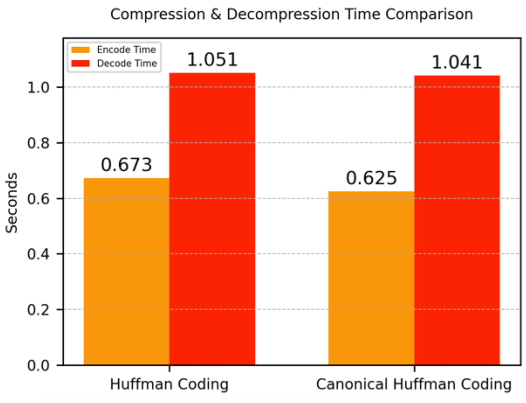


Figure 10. Time Comparison for DNA Sequences Compression

In contrast, Canonical Huffman Coding produced nearly identical compressed sizes and compression ratios to standard Huffman Coding, again stabilizing around 0.278–0.279 across all datasets. However, the metadata size was significantly reduced to 48 bytes, compared to 224 bytes in standard Huffman Coding. This smaller metadata footprint resulted in slightly better compression ratios with metadata included, though the improvement was marginal. The encoding times for Canonical Huffman were generally lower than those of standard Huffman for larger datasets (e.g., 1.019 seconds vs. 1.216 seconds for human DNA), suggesting computational efficiency during tree construction and symbol assignment. Interestingly, decoding times were mixed: while Canonical Huffman performed faster on smaller files (e.g., dog DNA: 0.561 seconds vs. 0.661 seconds), it required more time than standard Huffman for the human DNA dataset (1.851 seconds vs. 1.780 seconds).

Overall, these results confirm that Canonical Huffman Coding offers comparable compression efficiency to standard Huffman Coding while reducing metadata size and providing modest improvements in encoding speed. The trade-off lies in decoding performance, which can sometimes be slower for larger sequences. Nevertheless, both algorithms demonstrate stable compression performance for DNA sequences, achieving a compression ratio of approximately 0.28 across all datasets.

4.2 Performance Analysis between Huffman Coding and Canonical Huffman Coding on Audio

Table 8 shows the compression performance of the proposed method across different audio files. Smaller files such as *audio1_stereo* and *audio2_mono* achieve compression ratios of 0.759 and 0.775, respectively, indicating notable size reduction. Larger musical recordings like *burmese_harp* and *htoo_eain_thin* have higher compression ratios (0.927 and 0.918), reflecting less efficient compression for complex or large waveform content. Encoding and decoding times scale with file size, with *burmese_harp* requiring over 15 seconds to encode and 41 seconds to

decode, suggesting that computational cost is significant for large audio files.

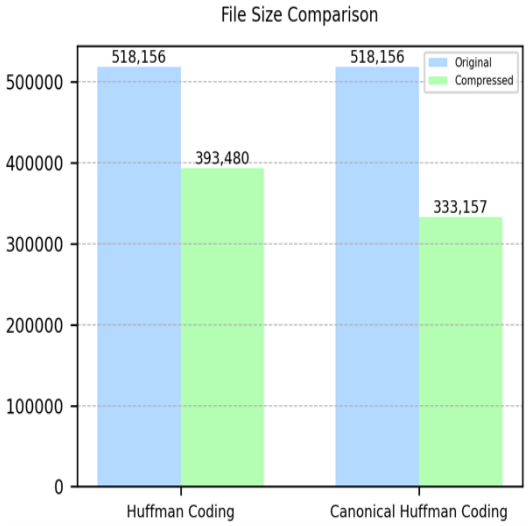


Figure 11. File Sizes Comparison for Audio Files Compression

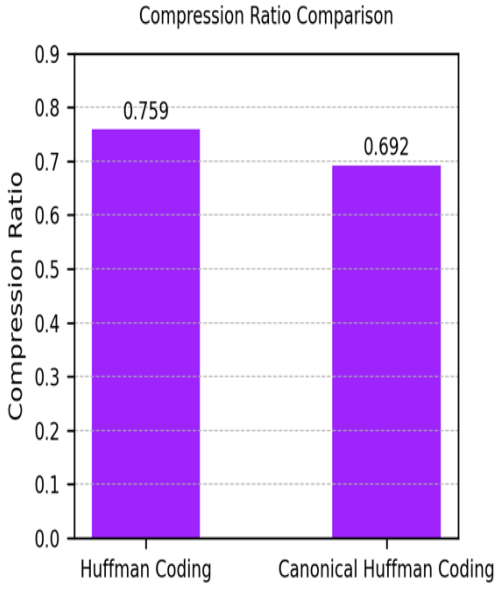


Figure 12. Compression Ratio Comparison for Audio Files

The term metadata stated across the analysis tables refers to the additional information needed to reconstruct the Huffman tree (or code lengths for Canonical Huffman), allowing the decoder to correctly decode the compressed data. Table 9 evaluates compression including metadata overhead, providing a more realistic assessment of storage efficiency. Without metadata, compression ratios improve, particularly for smaller files, with *audio1_stereo* at 0.643 and *audio2_mono* at 0.684.

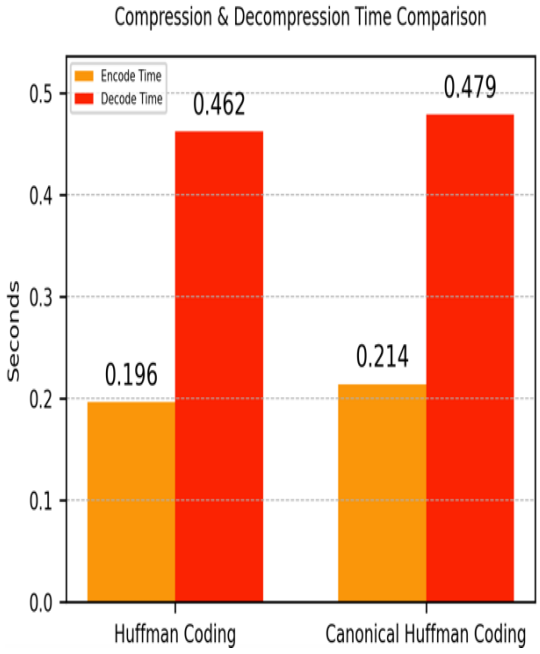


Figure 13. Time Comparison for Audio Files Compression

When metadata is included, the ratios increase to 0.692 and 0.693, highlighting the impact of metadata on overall storage savings. For larger audio files, metadata adds hundreds of kilobytes to several megabytes, raising effective compression ratios to 0.890 for *burmese_harp* and 0.897 for *htoo_eain_thin*. Encoding and decoding times remain consistent with Table 8, confirming algorithm stability while emphasizing the computational demand for complex audio data.

Files, as compression ratios remained close to the original size, and processing times increased substantially. This suggests that while Huffman-based methods are well-suited for text, they are not optimal for audio compression, where specialized algorithms (e.g., transform-based or perceptual coding) perform better.

4.3 Performance Analysis

Table 6. Results of Compressing DNA Sequences with Huffman Coding

File Names (.txt)	chimpanzee	dog	human
Original Size (bytes)	3,258,260	1,667,375	5,547,716
Compressed Size (bytes)	907,564	463,505	1,546,049
Metadata size (bytes)	224	224	224
Total Compressed Size (bytes)	907,788	463,729	1,546,273
Compression Ratio (without Metadata)	0.279	0.278	0.279
Compression Ratio (with Metadata)	0.27861	0.27811	0.27872
Encode Time (seconds)	0.673	0.329	1.216
Decode Time (seconds)	1.051	0.661	1.780

Table 7. Results of Compressing DNA Sequences with Canonical Huffman Coding

File Names (.txt)	chimpanzee	dog	human
Original Size (bytes)	3,258,260	1,667,375	5,547,716
Compressed Size (bytes)	907,564	463,505	1,546,049
Metadata size (bytes)	48	48	48
Total Compressed Size (bytes)	907,612	463,553	1,546,097
Compression Ratio (without Metadata)	0.279	0.278	0.279
Compression Ratio (with Metadata)	0.2786	0.27801	0.2786
Encode Time (seconds)	0.625	0.321	1.019
Decode Time (seconds)	1.041	0.561	1.851

Table 8. Results of Compressing Audio Files with Huffman Coding

File Names (.wav)	audio1_stereo	audio2_mono	burmese_harp	htoo_eain_thin
Original Size (bytes)	518,156	5,292,794	36,091,982	18,122,830
Compressed Size (bytes)	393,480	4,101,123	33,439,318	16,629,700
Compression Ratio	0.759	0.775	0.927	0.918
Encode Time (seconds)	0.196	1.974	15.352	7.767
Decode Time (seconds)	0.462	4.425	41.903	17.926

Table 9. Results of Compressing Audio Files with Canonical Huffman Coding

File Names (.wav)	audio1_stereo	audio2_mono	burmese_harp	htoo_eain_thin
Original Size (bytes)	518,156	5,292,794	36,091,982	18,122,830
Compressed Size (bytes)	333,157	3,619,693	31,018,464	15,340,163
Metadata size (bytes)	25,471	48,652	1,086,758	914,895
Total Compressed Size (bytes)	358,628	3,668,345	32,105,222	16,255,058
Compression Ratio (without Metadata)	0.643	0.684	0.859	0.846
Compression Ratio (with Metadata)	0.692	0.693	0.890	0.897
Encode Time (seconds)	0.214	1.889	16.059	7.390
Decode Time (seconds)	0.479	5.166	43.259	19.893

5. CONCLUSIONS

This study investigated the performance of Huffman Coding and Canonical Huffman Coding for text compression, as well as Canonical Huffman Coding for audio compression. The results demonstrate that algorithm choice strongly influences compression efficiency, metadata overhead, and processing speed.

For audio files, Canonical Huffman Coding achieved moderate compression ratios, but metadata size increased significantly for larger files, and encoding/decoding times scaled sharply with file size. These results indicate that Huffman-based methods, while effective for text compression, are less suitable for audio, where specialized or perceptual coding techniques are generally more efficient.

In conclusion, Canonical Huffman Coding provided better overall storage efficiency due to smaller metadata sizes compared to standard Huffman coding, while maintaining comparable compression ratios and fast encoding/decoding times. Huffman-based algorithms remain a reliable and efficient choice for text compression; however, for audio and other multimedia data, alternative algorithms are necessary to achieve meaningful compression and practical performance.

REFERENCES

[1] Athira Gopinath & Ravisankar. M (2020). Comparison of Lossless Data Compression Techniques, Proceedings of the Fifth International Conference on Inventive Computation Technologies (ICICT-2020), IEEE 2020, 628-633.

[2] Attiya Mahmood, Nazia Islam, Dawit Nigatu & Werner Henkel (2014). DNA Inspired Bi-Directional Lempel-Ziv-like Compression Algorithms, 2014 8th International Symposium on Turbo Codes and Iterative Information Processing (ISTC), IEEE 2014, 162-166.

[3] Gupta, Manasi (2021). Alternatives to Huffman Coding by Comparison to Other Algorithms, 2021 Innovations in Power and Advanced Computing Technologies (i-PACT), IEEE 2021, 1-4.

[4] Khalid Sayood (2005), Introduction to Data Compression, 3rd Edition.

[5] Sarkar, Subhra J. & Nabendu Kr. Sarkar (2016). A Novel Huffman Coding based Approach to Reduce the Size of Large Data Array, 2016 International Conference on Circuit Power and Computing Technologies [ICCPCT], IEEE 2016.

DETECTION OF CHEST X RAYS FOR PNEUMONIA USING RESNET50

Myint Zu Hmwe[1], and Thein Htay Zaw[2]

[1,2] Department of Information Technology Supporting and Maintenance, University of Computer Studies (Mandalay)
myintzuhmwe2000@gmail.com, theinhtayzaw.cu@gmail.com

ABSTRACT

Chest X-ray imaging technology enables the diagnosis of various lung diseases. Radiologists analyse chest X-ray images to identify lung diseases. Traditionally, this process can be time-consuming and costly. Implementing an automated system for pneumonia identification would make the process more accessible and efficient. Such a system allows individuals to detect pneumonia at an early stage and seek timely medical treatment. The chest X-ray image used Kaggle dataset in this paper. By applying machine learning techniques to medical image processing, healthcare professionals can achieve greater consistency and improved accuracy in diagnostic reporting. This paper used AI-based approaches for processing and analysing chest X-ray images to detect thoracic diseases using ResNet50 model. A total of 700 X-ray images is used in this study to identify pneumonia cases. The ResNet-50 architecture achieved the highest performance with a classification accuracy of 97.14%.

.

KEYWORDS

Pneumonia Detection, CNN Architecture, Resnet50

1. INTRODUCTION

Although artificial intelligence (AI) was acknowledged as a field of study as early as the 1950s, the scientific community did not investigate it extensively for a long time because of its poor viability in practice [1] The advent of massive data and the growing availability of computing power over the past few decades have made artificial intelligence (AI) a major topic of discussion in both industry and academics. The development of AI progressed through several stages, which are sometimes referred to as the "Seasons of AI." The years between the 1970s and the 2000s, which are commonly referred to as the "AI Winter," were until recently the most well-known AI season [2].

Pneumonia is a serious respiratory disease that causes inflammation in one or both lungs, resulting in symptoms like fever, cough, and difficulty breathing. It is especially dangerous for young children, as it accounts for approximately 15% of mortality in children under five years old [3]. The disease is more common in developing countries, where its effects are exacerbated by poor living conditions, pollution, overcrowding, and limited access to healthcare [4]. CNN forms the backbone of modern computer vision systems and has revolutionized the field of image processing. CNNs are extremely effective at image identification tasks because they are fundamentally built to automatically learn hierarchical patterns and features from input images without the requirement for manually created features. Convolutional, pooling, and fully linked layers are all combined in CNN models. By learning progressively more intricate features, from low-level edges and textures to high-level object representations, CNN is able to process images in a hierarchical manner [5]. For instance, CNNs are able to reliably differentiate between normal and pathological X-rays [6], COVID-19 [8], and lung cancer [7].

Convolutional Neural Networks (CNNs), in particular, have demonstrated their self-potential to extract valuable features in image classification tasks when Deep Learning (DL)

models are used [9], [10]. This feature-extraction process necessitates the use of transfer learning techniques, in which CNN models that have already been trained learn generic features on massive datasets such as ImageNet, which are then applied to the necessary job.

The proposed system consists of five sections. The related study of utilizing ResNet50 to detect pneumonia in chest X-rays is expressed in Section 2. The underpinning theory for identifying pneumonia in chest X-rays using ResNet50 is described in part three of this study, and section four shows the suggested system design. The experimental result is described in each of the five sections. Finally, the system's conclusion and potential developments are covered.

2. LITERATURE REVIEWS

Since chest X-ray (CXR) imaging can show important signs including increased lung opacity and consolidation, it is frequently utilized to diagnose pneumonia. Improving patient outcomes and starting therapy on time depend on a quick and precise diagnosis. Convolutional Neural Network (CNN) models that have been pretrained are used to extract features from images. Medical image analysis and pattern recognition are successfully accomplished using CNNs. This method helps medical professionals determine whether a patient has pneumonia or not. Timely diagnosis and treatment depend heavily on early and precise detection [11]. The author suggested a deep learning technique for identifying pneumonia based on convolutional neural networks. A processing model designed to address a classification challenge is created using a deep learning technique based on a Convolutional Neural Network (CNN). The model is intended to identify whether a chest X-ray exhibits symptoms typical of pneumonia and to categorize the images into two groups: those that are pneumonia-positive and those that are pneumonia-negative. Despite the model's comparatively high accuracy of almost 90%, overfitting is still a possibility because of the dataset's small size.[12].

The system talked about a deep learning model that uses VGG-16 and neural networks to diagnose pneumonia from chest X-ray pictures. Two datasets of chest X-ray (CXR) images are used to detect and categorize pneumonia using a deep learning (DL) model based on VGG16. Neural Networks (NN) in conjunction with the VGG16 model produced an accuracy of 92.15%, recall of 0.9308, precision of 0.9428, and F1-score of 0.937 for the first dataset. Furthermore, tests were performed on a second CXR dataset with 6,436 pictures that included COVID-19, normal, and pneumonia cases. The accuracy of the VGG16 with NN on this dataset was 95.4%, and its F1-score, recall, and precision were all 0.954 [13]. The author proposed reviewing the use of neural networks to detect pneumonia in chest X-ray pictures. In order to treat pneumonia effectively, early detection is essential. Because chest X-rays are more affordable than alternative techniques, they are frequently utilized for diagnosis. However, the process takes a long time because of the increasing amount of X-rays that need to be reviewed and the lack of skilled diagnosticians. Therefore, the demand for better diagnostic techniques is urgent. The author examined current studies that investigate the application of neural networks to the identification of pneumonia. Results from networks like VGG, InceptionNet, ResNet, and hybrid models are preferable to those from hierarchical CNNs, AG-CNN, and R-CNN[14]. The author offered a novel multi-scale transformer approach for accurate and efficient pneumonia detection. Following the model's use of the Chest X-ray Masks and Labels dataset, two datasets—the Kermany and Cohen datasets—are applied. Lung areas are isolated from the two datasets to increase the efficacy of ensuing classification tasks. The segmentation stage guarantees that the classification model concentrates on the most pertinent regions by separating lung areas. employed ResNet models that have already been trained (ResNet-50 and ResNet-101). The accuracy increases from 84.62% (baseline) to 91.23% while using the ResNet-50 backbone. Similarly, accuracy rises from 83.93% to 90.22% with the ResNet-101 backbone. [15].

Both data curation and deep learning research should focus more on clinically relevant uses of AI in CXR interpretation in order to better match future efforts with clinical priorities. Additionally, more accurate algorithm comparison and benchmarking would be made possible by the creation of standardized public

challenges that use well-annotated datasets for pertinent clinical tasks [16].

3. MODEL ARCHITECTURE

The In recent years, there has been growing interest in developing machine learning models to assist physicians in diagnosing pneumonia using chest X-ray images. These models have demonstrated promising results and offer the potential to improve the accuracy and efficiency of pneumonia diagnosis

3.1. RestNet50

ResNet-50 is a deep convolutional neural network that uses residual learning to enable the training of very deep networks. It is part of the ResNet family of architectures, which are designed to address the problem of vanishing gradients in deep networks. The main innovation in ResNet is the introduction of skip connections or residual connections, which allow gradients to flow more easily through the network during backpropagation. This makes it possible to train very deep models with hundreds or even thousands of layers without the degradation of performance due to training difficulties.

ResNet-50 strikes a balance between model depth and computational efficiency. It is widely used in image classification tasks and has achieved state-of-the-art results in several benchmark datasets, including the ImageNet challenge. In the context of pneumonia detection, ResNet-50 is particularly effective because it can learn complex features from chest X-ray images, such as lung opacities and other pneumonia-related abnormalities, enabling it to classify whether a given image contains signs of pneumonia.

For a more detailed explanation of how ResNet works, including the core concepts of residual connections, skip connections, and how the architecture is designed to overcome the limitations of traditional CNNs.

4. Methodology

The pneumonia chest X-ray image dataset is obtained from Kaggle.com. This dataset contains 700 Chest X-ray images. For preprocessing, X-ray images are resized. The original 2-D images are recorded in high-quality and come in various sizes, such as 2090 × 1858, 1422 × 1152, 1810 × 1434, 712 × 439 etc. All images are grayscale images. They are converted into colour images with a size of 224 × 224 × 3, representing height, width, and the number of channels, respectively. Figure 1 depicted the proposed system.

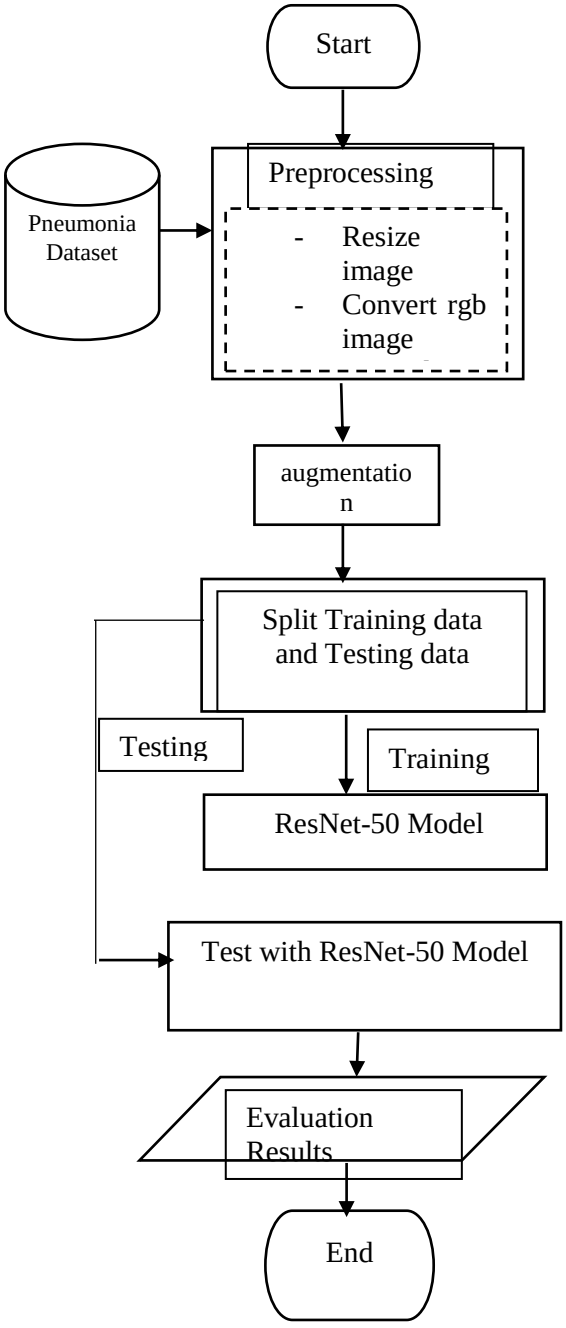


Figure 1: Proposed System Architecture

4.1. Dataset

This proposed methodology begins with the collection of required images from Kaggle.com. This dataset contains 5,856 Chest X-ray images. The first dataset consists of

5856 images of chest X-rays of which 4273 are pneumonia images and 1583 are normal chest X-ray images. It is divided into three folders, named train, val and test, that are used as training, validation and testing data. the training folder contains 5216 images, validation folder 16 images and testing folder 624 images. The training set contains 5,216 images, including 3,875 pneumonia images and 1,341 normal images. The testing set contains 624images, including 390 pneumonia images and 234 normal images. The validation set contains 16 images, including 8 pneumonia images and 8 normal images. Following figure2 and figure3 are displayed original normal gray scale image and original pneumonia image.

Figure 2: Original Normal Gray image

Figure 3: Original Pneumonia Gray image

4.2. Preprocessing

Image preprocessing plays a critical role in detecting pneumonia using deep learning with chest X-ray images. The original 2-D images are gray scale images and various sizes, such as 2090 × 1858, 1422 × 1152, 1810 × 1434, 712 × 439 etc. For the purpose of this experiment, they were converted to RGB images and resized to 224 x 224 x 3 pixels. Resized normal gray image and resized pneumonia gray image are shown in Figure 4 and Figure 5.

Figure 4: Resized Normal Gray image (224*224)

Figure 5: Resized Pneumonia Gray image (224*224)

After resizing the image and converting it to an RGB image, we apply normalization. The following Figures 6 and 7 present the converted RGB images.

Figure 6: Resized Normal RGB image

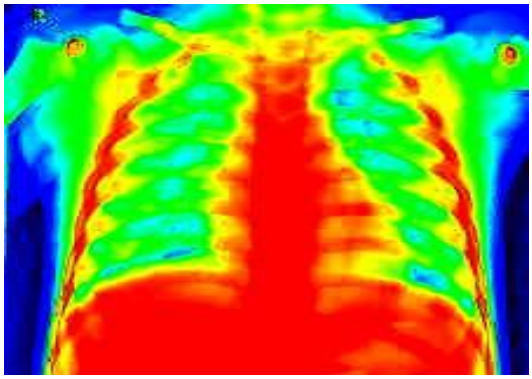


Figure 7: Normalized Normal RGB image

Normalize each image independently by subtracting the mean and dividing by the standard deviation RGB image.

$$\text{Normalization} = \frac{pixel\ vlue - \alpha}{\sigma}$$

∞ = mean of all pixel values in the binary image

δ= standard deviation

Example

image A= [1 2 3; 2 3 4; 3 3 4];

Flattened= [1, 2, 3, 2, 3, 4, 3, 3, 4]

Total pixels = 9

Sum = 25

mean (∞) = 25/ 9 ≈ 2.777

$$\text{Std deviation}(\delta) = \sqrt{\frac{1}{N} \sum (x-\mu)^2}$$

= 0.9163

Normalized _image =

[-1.940 -0.849 0.242;

[-0.849 0.242 1.333;

[0.242 0.242 1.333]

Image augmentation plays an important role in pneumonia detection using deep learning with chest X-ray images. The type of augmentation is applied: horizontal flip.

The images shown in Figures 8 is the horizontal flipped image.

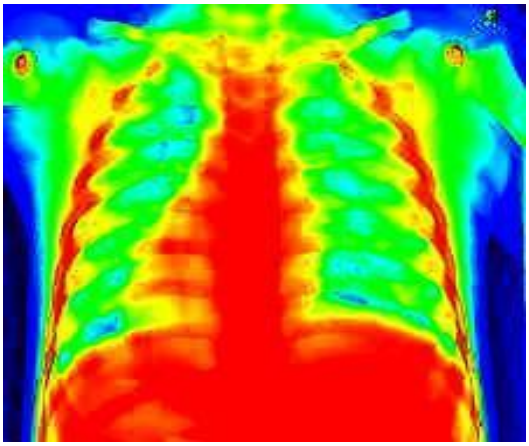


Figure 8: Horizontal Flip Normal image

4.3. **ResNet50** Architecture

ResNet50 architecture is essentially based on the residual block concept that uses input and output dimensions to training network. It consists of:

1.Input Layer: It accepts an image as input.

2.Convolutional Layers: Extract features from the input image.

3.Bottleneck Design: Each residual block contains three layers, a 1x1 convolution, a 3x3 convolution, a 1x1 convolution.

4. Residual Connections: Enable gradients to flow through the network.

5.Pooling Layers: Reduce the spatial dimensions of the feature maps.

6.Fully Connected Layers: Perform the final classification.

7.Output Layer: Outputs the class probabilities.

RESNET-50 is a convolution neural network that is 50 layers deep and here we are using the model whose layers are pre trained on ImageNet. The following Table1 is capable of doing classification of images in 1000 object categories.

5. Experimental Results and Discussion

The effectiveness of suggested system is analysed by measuring the evaluation criteria on the detection of Chest X Rays for Pneumonia. It is a great performing model. According to epochs, the accuracy result can change.

To visually demonstrate that the improved ResNet50 model using epochs 5, 7, 10 and14. All experiments also present the figures of training progress during training processes and confusion matrices of classification results, as shown in following figures:

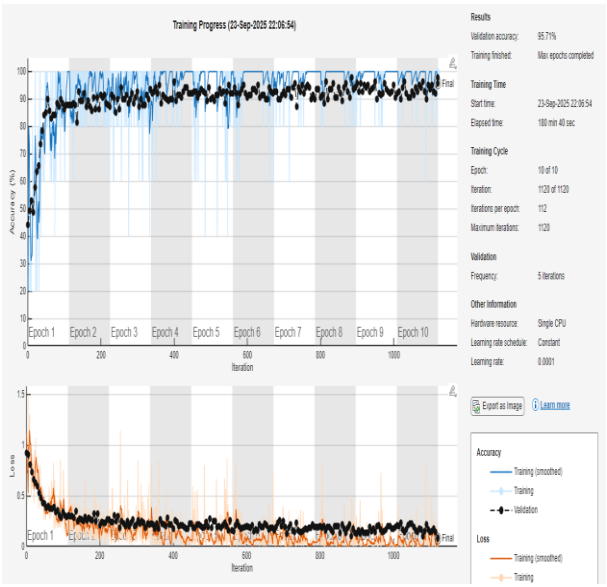


Figure 9: Training progress for image classification with epoch5

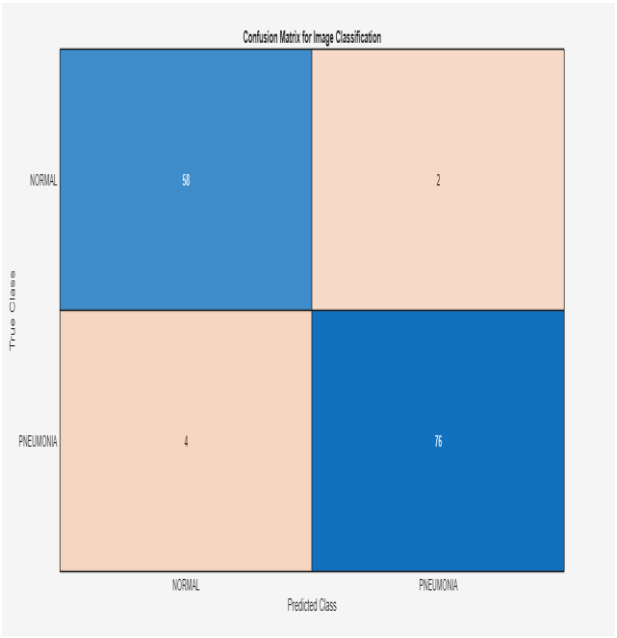


Figure 10: Confusion Matrix for image classification with epoch5

Training progress for image classification with epoch5 is shown in figure 9 and the confusion matrix of classification result in shown in figure 10. The result reached 95.0%.

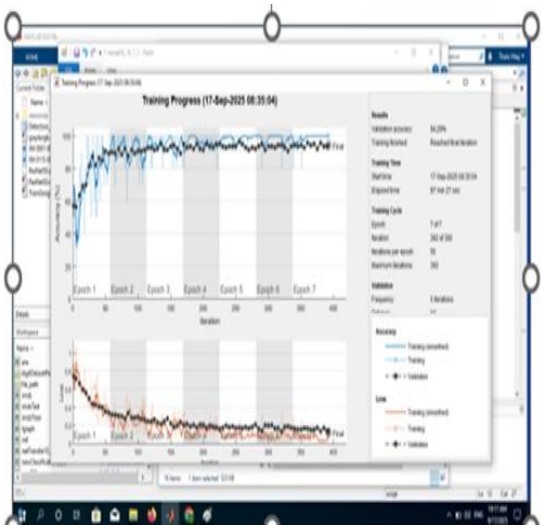


Figure 11: Training progress for image classification with epochs 7

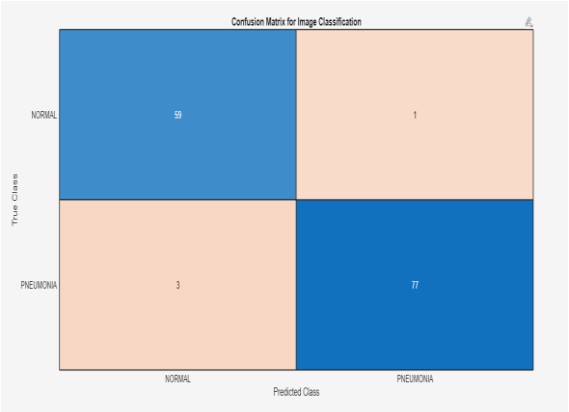


Figure 12: Confusion Matrix for image classification with epoch7

Figure 11 displays the training progress for image classification using epoch 7, and Figure 12 displays the classification result's confusion matrix. The outcome was 97.14%.

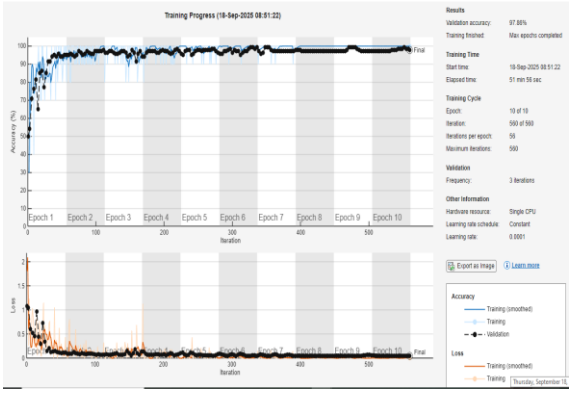


Figure 13: Training progress for image classification epoch 10

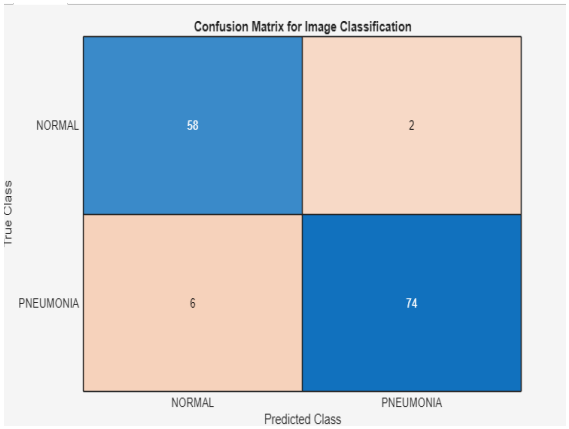


Figure 14: Confusion Matrix for image classification with epoch10

Figure 13 depicts the image classification training procedure using epoch 10, and Figure 14 displays the confusion matrix for the classification outcome. 94.29 percent was the outcome.

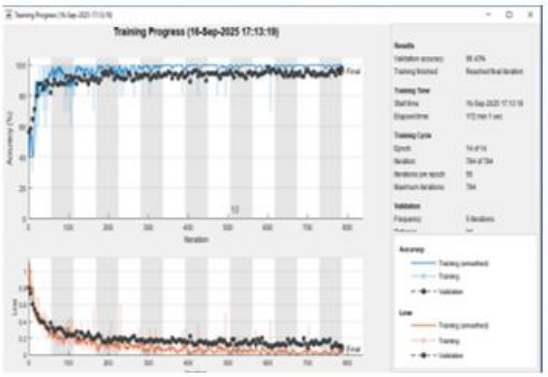


Figure 15: Training progress for image classification epoch 14

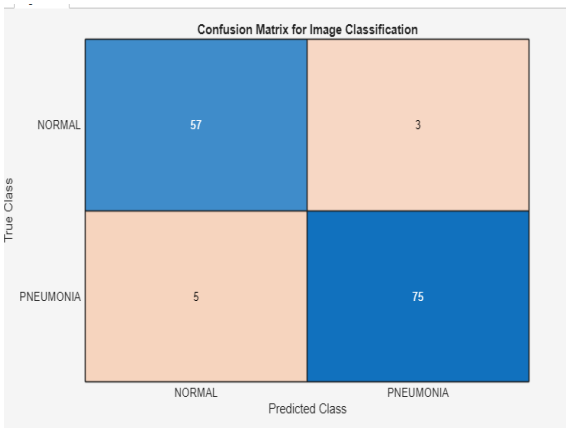


Figure 16: Confusion Matrix for image classification with epoch14

Figure 16 shows the confusion matrix for the classification result, and Figure 15 shows the training progress for image classification using epoch14. The result was 94.29 %.

The number of epochs significantly impacts the training and performance of any deep learning model. An epoch represents one complete pass through the entire training dataset.

If the number of epochs is too low, the model may not have enough opportunities to learn the underlying patterns in the data.

Training for too many epochs can lead to overfitting. In this scenario, the model starts to memorize the training data, including noise and specific details, instead of learning generalizable patterns.

As the number of epochs increases, the model learns more complex features and improves its performance on both the training and validation sets. There is an optimal range where the model achieves its best generalization capacity, meaning it performs well on unseen data. Therefore, by looking at these experiment results, epoch 7 is optimal range of epoch.

6. CONCLUSIONS

The system presents the effectiveness of deep learning technique for the classification tasks of pneumonia from medical imaging (x-ray chest). Resnet 50 pretrain model is used to detect pneumonia. This system also analysis the detection performance by changing the range of epochs during training process. The dataset used is Chest X-ray images from Kaggle.com. The system generates results with 95.0% accuracy for epochs 5, 97.14% accuracy for epochs 7, 94.29%accuracy for epochs 10 and 94.29% accuracy for epochs 14 on tested images. The accuracy performance slightly different change according to the epochs. Result reached the highest accuracy at epochs 7 in this system.

REFERENCES

[1] Haenlein, M. and Kaplan, A., 2019. A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. *California management review*, *61*(4), pp.5-14.

[2] Kaul, V., Enslin, S. and Gross, S.A., 2020. History of artificial intelligence in medicine. *Gastrointestinal endoscopy*, *92*(4), pp.807-812.

[3] Saber, A., Parhami, P., Siahkarzadeh, A., Fateh, M. and Fateh, A., 2024. Efficient and accurate pneumonia detection using a novel multi-scale transformer approach. *arXiv preprint arXiv:2408.04290*.

[4] Saber, A., Parhami, P., Siahkarzadeh, A., Fateh, M. and Fateh, A., 2024. Efficient and accurate pneumonia detection using a novel multi-scale transformer approach. *arXiv preprint arXiv:2408.04290*..

[5] Ibrokhimov, B. and Kang, J.Y., 2022. Deep learning model for COVID-19-infected pneumonia diagnosis using chest radiography images. *BioMedInformatics*, *2*(4), pp.654-670.

[6] Aktas, K., Ignjatovic, V., Ilic, D., Marjanovic, M. and Anbarjafari, G., 2023. Deep convolutional neural networks for detection of abnormalities in chest X-rays trained on the very large dataset. *Signal, Image and Video Processing*, *17*(4), pp.1035-1041.

[7] Shah, A.A., Malik, H.A.M., Muhammad, A., Alourani, A. and Butt, Z.A., 2023. Deep learning ensemble 2D CNN approach towards the detection of lung cancer. *Scientific reports*, *13*(1), p.2987.

[8] Reshi, A.A., Rustam, F., Mehmood, A., Alhossan, A., Alrabiah, Z., Ahmad, A., Alsuwailem, H. and Choi, G.S., 2021. Research Article An Efficient CNN Model for COVID-19 Disease Detection Based on X-Ray Image Classification.

[9] Sharif Razavian, A., Azizpour, H., Sullivan, J. and Carlsson, S., 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 806-813).

[10] Nijhawan, R., Verma, R., Bhushan, S., Dua, R. and Mittal, A., 2017, December. An integrated deep learning framework approach for nail disease identification. In *2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)* (pp. 197-202). IEEE.

[11] A.Pranaya, D.V.Sowmya, L.Poojitha, P.Grace, K.Bhavya, and Dr.N.V. Ganapathi Raju, "Pneumonia Detection Using Deep Learning", E3S Web of Conferences 391, 01067 (2023) https://doi.org/10.1051/e3sconf/202339101067, ICMED-ICMPC 2023.

[12] Račić, T. Popović, S. Čakić, and S. Šandi, "Pneumonia Detection Using Deep Learning Based on Convolutional Neural Network", 25th International Conference on Information Technology (IT), Zabljak, 17 – 20 February 2021.

[13] Shagun Sharma and Kalpna Guleria," A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks", International conference on machine learning and data engineering, 2023.

[14] Daniel Joseph Alapat , Malavika Venu Menon1, Sharmila Ashok, "A Review on Detection of Pneumonia in Chest X-Ray Images Using Neural Networks", J Biomed Phys Eng 2022; 12(6)

[15] Alireza Saber1 · Pouria Parhami1 · Alimohammad Siahkarzadeh1 · Mansoor Fateh1, Amirreza Fateh, "Efficient and Accurate Pneumonia Detection Using a Novel Multi-Scale Transformer Approach ", arXiv:2408.04290v2 [eess.IV] 3 Nov 2024.

[16] Erdi Çallı1 , ∗, Ecem Sogancioglu 1 , Bram van Ginneken, Kicky G. van Leeuwen, Keelin Murphy, "Deep learning for chest X-ray analysis: A survey", https://doi.org/10.1016/j.media.2021.102125.

Author's Biography

Myint Zu Hmwe received the Bachelor of Computer Science (B.C.Sc.) degree in 2023 from the University of Computer Studies, Mandalay, Myanmar. She is currently pursuing her Master of Computer Science (M.C.Sc.) degree at the same university.

COLOR FOLLOWING ROBOT USING KINEMATIC THEORY AND PID CONTROL

Htet Wai Lin[1], and May Mya Aung[2]

[1,2]Faculty of Computer Systems and Technologies, University of Computer Studies (Mandalay)

htatwailin22@gmail.com, maymyaaung@cumandalay.edu.mm

ABSTRACT

An autonomous color-following robot is designed using a structured approach that begins with robust color detection. The system employs HSV thresholding to isolate target colors, ensuring reliable performance across diverse lighting conditions.[2] Once the colored object is detected, its position relative to the frame center is calculated to determine tracking error. This error feeds into a PID controller, which continuously adjusts motor speeds using proportional, integral, and derivative components to minimize deviation and maintain smooth tracking. The corrected control signals are then translated into physical motion through a differential drive system, guided by kinematic equations that model the robot's movement. This integration of image processing, feedback control, and kinematic modeling enables precise and responsive object-following behavior using low-cost hardware.[1]

KEYWORDS

color following robot, kinematics theory, paranoid (right-left), PID control, RGB to HSV

1. INTRODUCTION

Autonomous mobile robots have become increasingly vital in modern applications ranging from industrial automation to intelligent surveillance. Among these, color-following robots offer a compelling solution for tasks involving object tracking, line following, and dynamic interaction with visual cues. The ability to detect and follow a specific color in real time enables robots to navigate complex environments without relying on predefined paths or external markers[1].

This paper explores the development of a color-following robot using a Raspberry Pi-based embedded system, integrating image processing, kinematic modeling, and PID control. The robot is designed to detect a target color using a camera module, interpret its position within the frame, and adjust its motion accordingly.[1] To achieve robust color detection, the system converts RGB input into HSV space, which provides greater resilience to lighting variations and simplifies thresholding [2]. By integrating affordable hardware components with sophisticated control algorithms, this system presents a cost-effective yet powerful solution for real-time color tracking in autonomous robotics. The design emphasizes scalability and efficiency, making it suitable for a wide range of applications from educational platforms to industrial automation. Through a combination of simulation-based validation and hands-on implementation, the system not only proves its functional reliability but also sheds light on the practical challenges encountered during autonomous visual navigation. These include issues such as environmental variability, sensor calibration, and real-time processing constraints. The project offers valuable insights into how intelligent control strategies can compensate for hardware limitations, ultimately contributing to the development of robust and adaptable vision-based tracking systems [6].

2. LITERATURE REVIEW

Several studies emphasize the importance of robust color-based image processing for object tracking and autonomous navigation. The JCSE article (Vol. 5, No. 1, Feb 2024) presents a human-following trolley robot that uses HSV-based color segmentation to detect and track human targets under varying lighting conditions. This approach demonstrates the effectiveness of HSV thresholding in real-time robotic vision systems.

PID control emerges as a central theme across multiple studies. Islam et al. (2014) propose an adaptive PID controller for differential drive robots, focusing on speed synchronization and minimizing drift during motion.

Their results show improved trajectory stability and responsiveness to dynamic changes.The ACTA Universitatis Cibiniensis (2019) article investigates position control using PID feedback, demonstrating how tuning PID parameters can significantly enhance tracking accuracy and reduce overshoot in mobile platforms.

Theoretical foundations for robot motion are addressed in Reza N. Jazar's *Theory of Applied Robotics* (2022), which provides comprehensive coverage of kinematics, dynamics, and control strategies. This text is essential for understanding how control signals translate into physical movement, especially in differential drive systems.

Ben-Ari and Mondada's *Elements of Robotics* complements this by offering practical insights into robot architecture, sensor integration, and motion planning. Their work bridges the gap between theoretical modeling and real-world implementation, making it highly relevant for scalable robotic systems.

A recurring theme appears in all of these sources: the foundation of successful object-following robots is made up of kinematic modeling, PID feedback control, and HSV-based color detection.

Few studies provide a completely integrated system that integrates all three with real-time performance indicators, despite the fact that many research validate these elements separately. Furthermore, dynamic thresholding and adaptive PID tuning in uncontrolled lighting are still unresolved issues.

3. METHODOLOGY

This section outlines the technical approach used to design and implement a color-following robot capable of autonomous navigation using image processing, kinematic modeling, and PID control.

The system integrates hardware and software components to detect a target color, compute positional error, and adjust motor commands in real time.

3.1. COLOR DETECTION

To achieve robust color tracking, the system converts captured frames from RGB to HSV color space. HSV allows for more reliable segmentation under varying lighting conditions. The target color is isolated using predefined threshold values for hue, saturation, and value.[1]

A binary mask is generated to highlight the detected region, and its centroid is computed to determine the position of the object within the frame.[2]

This positional data is used as input to the robot's steering control loop. The system operates in real time, with optimized frame resolution and processing latency to ensure responsive tracking.

While effective in controlled environments, the method is limited by static thresholding and sensitivity to background interference. [2]

The error position of the detected color is compared to the center of the camera frame to calculate the error:

$$\mathbf{Error = x\_center - x\_object} \quad (1)$$

This error value serves as input to the control algorithm, indicating how far the object is from the desired tracking path.[8]

3.2. PID CONTROL

To refine the robot's motion, a PID controller is implemented.[9] The control output u(t) is calculated as:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(t)dt + K_d \cdot \frac{de(t)}{dt} \quad (2)$$

where :

- u(t) is the control output
- e(t) is the error at time (t)
- K_p, K_i *and* K_d *the proportional, integral and differential gains*
- $\int_0^t e(t)dt$ *is the integral of the error over time*
- $\frac{de(t)}{dt}$ *is the rate of change of the error*

The robot uses a PID controller balancing proportional, integral, and differential terms to adjust its movement based on the object's position. This enables smooth and responsive tracking behavior. If no object is detected, the system loops back to continue reading video input, ensuring real-time adaptability.[1]

Figure 1 illustrates the inner workings of a PID control system applied to a DC motor, showcasing how precise regulation is achieved through feedback. The process begins with a set point representing the desired motor output, which is compared to the actual feedback signal to generate an error value. This error is processed by the PID controller, which calculates corrective actions based on proportional, integral, and derivative terms. The resulting control signal adjusts the motor input to minimize the error over time, ensuring stable and accurate performance. Through continuous feedback and adjustment, the system dynamically compensates for disturbances and maintains the desired output.

This error is then processed through three distinct components: the proportional block scales the error by a gain factor K_p, the integral block accumulates the error over time with gain K_i, and the derivative block predicts future error trends using gainK_d. These three terms are summed to produce a control signal that adjusts the motor's behavior, continuously minimizing the error. The feedback loop ensures that the system remains responsive and stable, making PID control a cornerstone of precision in robotics and automation.[3][5]

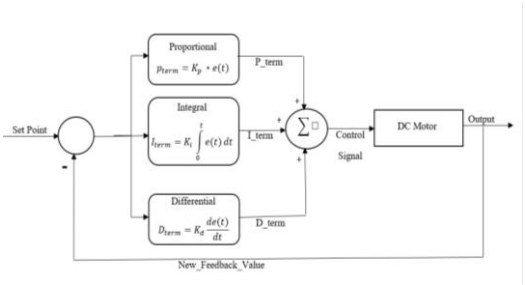


Figure 1. PID Control Block Diagram

3.3. Kinematic Modeling

The locomotion of the robot is governed by differential drive kinematics, a widely adopted model for wheeled mobile robots due to its simplicity and effectiveness in planar navigation scenarios.[9] In this framework, the robot's movement is characterized by its linear and angular velocities, which are derived from the rotational velocities of the left and right wheels over time.[10] These velocities are computed using the following equations:

$$x(t) = x_0 + \frac{1}{2}\int_0^t (v_r(t) + v_l(t))\cos(\theta(t))\,dt \quad (3)$$

$$y(t) = y_0 + \frac{1}{2}\int_0^t (v_r(t) + v_l(t))\sin(\theta(t))\,dt \quad (4)$$

Where:

- x , y = position
- θ = orientation
- x_0, y_0 = initial position
- $v_r(t), v_l(t)$ = velocities of wheels at time(t)
- L = distance between two wheel

These equations translate control signals into physical motion, allowing precise navigation.[9][10]

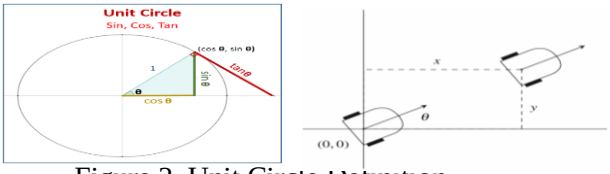


Figure 2. Unit Circle Definition

4. SYSTEM OVERVIEW

The robot system integrates a Raspberry Pi, RobotHat motor controller, camera module, and differential drive motors. The Raspberry Pi serves as the central processing unit, receiving visual input from the camera and executing control algorithms to regulate motor behavior.[11] The RobotHat interfaces with both DC motors and a servo motor, enabling precise actuation based on real-time feedback.

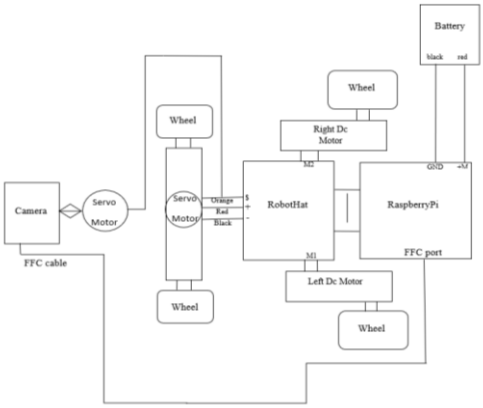


Figure 3: Block Diagram of a robot

This schematic illustrates the connections between the Raspberry Pi, RobotHat, camera, motors, and battery. It highlights the control flow from visual input to motor actuation. Connect the ROBOTHAT to the GPIO pins of the raspberry pi. And the camera is connected directly to the raspberry pi through the ffc port. Servo motors are connected to the digital pins which are on the ROBOTHAT to control the movement of the camera to track the object and to steer the robot's movement. Motor drivers which are used to drive the car is also connected to the PWM pins of the ROBOT-HAT. And also betteries are connected to the robothat and the raspberry pi is powered through the robot hat. When the power is on the target object is detected through the camera and delivered it to the raspberry pi to determine it is the color we want or not. If it is the color we want raspberry pi produce the PWM output which will use by motor drivers and follow the color object. In this paper camera will take a red color object as an input, image processing and the process to follow the input red color object and the output will be follow that red color object.[11]

Figure 4 illustrates the overall process flow of the object tracking system, which integrates computer vision with robotic control to achieve autonomous visual navigation. The system begins by continuously capturing video input from a camera, which is then processed to detect the object of interest using image analysis techniques. Upon successful detection, the system evaluates the presence of the object; if absent, it loops back to resume video acquisition. If the object is present, its coordinates are extracted and transmitted to the robot's control unit. These coordinates serve as input for a Proportional-Integral-Derivative (PID) controller, which dynamically adjusts the robot's movement to maintain alignment with the target. This closed-loop control enables the robot to follow the object in real time, completing the tracking cycle. The flowchart encapsulates the modular and reactive nature of the system, highlighting its ability to adapt to changing visual inputs and maintain robust tracking performance.

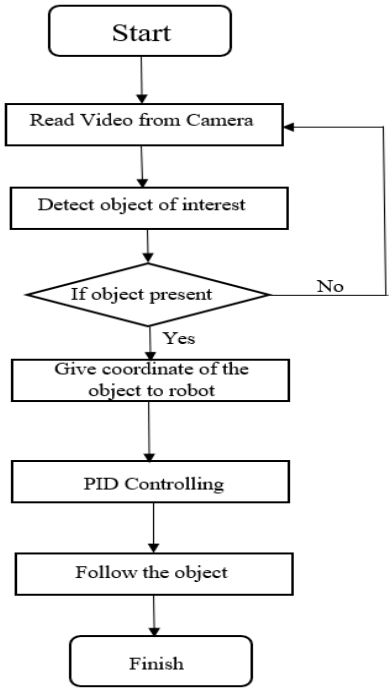


Figure 4. System Flow Chart

4.1 Hardware Integration

The Raspberry Pi interfaces with the camera module via the CSI (Camera Serial Interface), enabling efficient image capture for real-time processing. Motor control is handled through GPIO pins connected to a dedicated HAT, which simplifies wiring and expands functionality.[11] Power is supplied by a regulated battery pack, ensuring stable voltage for both the Pi and peripherals. The entire system is housed in a lightweight, mobile chassis designed for easy maneuverability and field deployment.

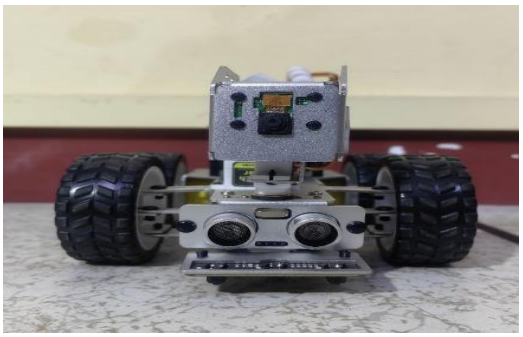


Figure 5. Front view of Robot

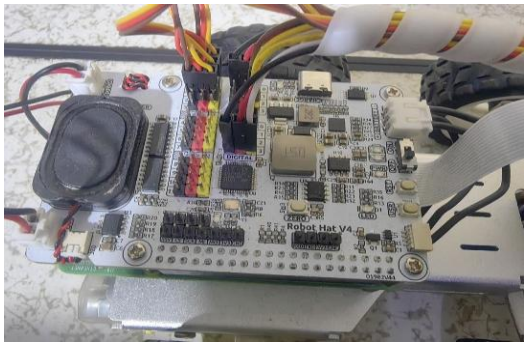


Figure 6. Front of Robot

5. Simulation and Results

This section presents the simulation studies and experimental results used to validate the performance of the color-following robot. The goal is to demonstrate the effectiveness of the integrated system including image processing, PID control, and kinematic modeling in achieving stable and responsive tracking behavior under real-world conditions.

5.1 Kinematic Simulation

The robot's motion was first simulated using MATLAB to verify the accuracy of the differential drive model. Given the wheel radius r=0.03r = 0.03 m and wheelbase L=0.15L = 0.15 m, the robot's linear and angular velocities were computed based on motor inputs. The simulation confirmed that when both motors rotate at equal speeds, the robot moves forward in a straight line. When one motor speed is increased relative to the other, the robot performs a smooth turn.

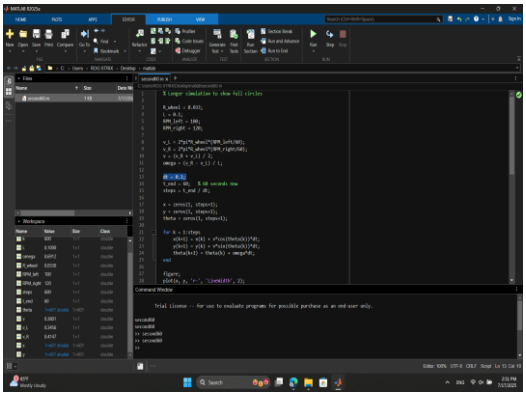


Figure 7. Matlab Simulation of Kinematic Theory

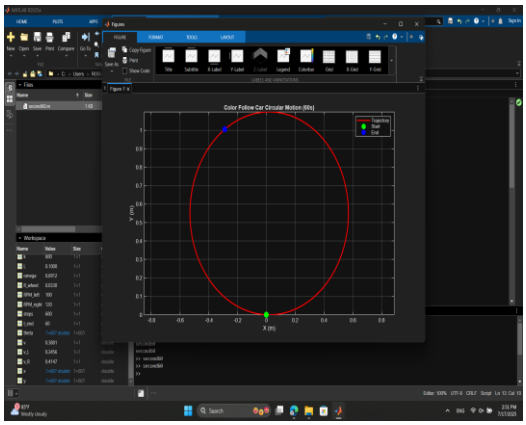


Figure 8. Matlab Simulation of Kinematic Theory

5.2 PID Controller Tuning

To achieve precise and stable tracking, the PID controller was tuned through iterative testing. Multiple gain sets were evaluated to observe their impact on error convergence and system stability. The optimal configuration was found to be:

- K_p = 0.6
- K_i = 0.1
- K_d = 0.05

Under this gain set, the robot responded quickly to positional error without overshooting or oscillating. The error curve showed a smooth decay, with the robot stabilizing its heading within 1.2 seconds of detecting a color shift. Comparative plots revealed that lower K_p values resulted in sluggish response, while higher K_d values introduced instability.

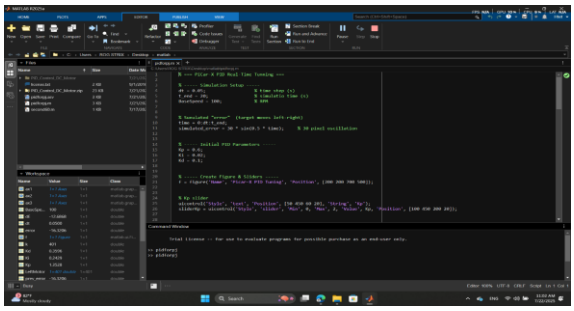


Figure 9. Matlab Simulation of PID Control

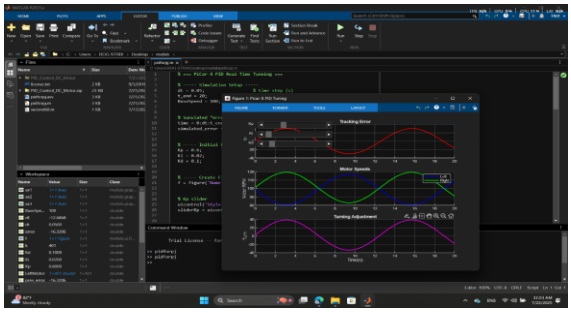


Figure 10. Matlab Simulation of PID Control

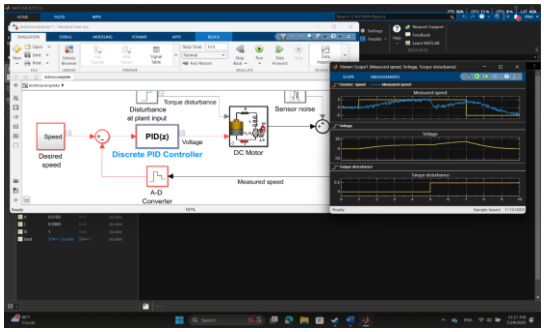


Figure 11. Matlab Simulation of PID Control on DC Motor

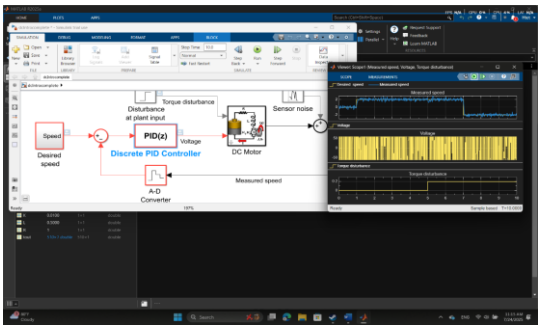


Figure 12. Matlab Simulation of PID Control on DC Motor

5.3. Real-Time Behavior and Field Testing

Color detection accuracy for the image processing pipeline was tested using both static and dynamic color targets. Frames were captured and converted from RGB to HSV, followed by thresholding to isolate the target color.[8] The system consistently detected the correct region, even under varying lighting conditions and background noise.

Centroid calculation was performed using contour analysis, and the positional error was computed relative to the center of the frame. The average detection latency was approximately 0.18 seconds, enabling near real-time responsiveness. False positives were minimal, and the system successfully ignored irrelevant colors outside the defined HSV range.

In live tests, the robot was placed in a controlled environment with a colored object (e.g., red ball) moving across its field of view. The robot demonstrated smooth tracking behavior, adjusting its heading and speed based on the object's position. When the object moved left, the robot turned left; when centered, it moved forward.

The paranoid algorithm ensured rapid decision-making, while the PID controller refined motor output to prevent abrupt or jerky movements. The robot was able to follow the object across a 2-meter test path with minimal deviation. When the object was removed, the robot paused and resumed tracking once the color reappeared demonstrating robust recovery behavior.

5.4. Real-Time Test Results

To validate the robot's distance control behavior under real-world conditions, a series of live tests were conducted and logged using onboard sensors.[10] The robot was placed in a controlled environment and tasked with maintaining a fixed distance from a colored target while tracking its position.

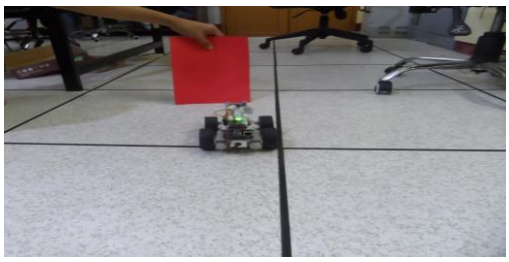


Figure 13. Testing With Red Color Paper

Figure 14. Testing With Red Color Paper

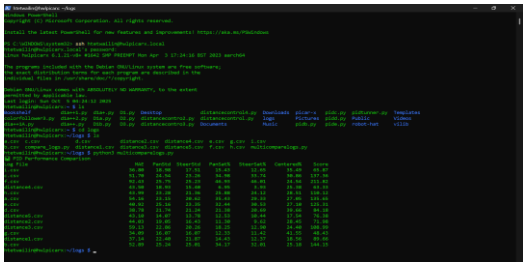


Figure 15. Logged Data From Robot

Figure 16. Logged Performance Metric form Robot

Figure 15 and 16 show the robot's tracking performance, a composite Score was calculated for each test run using a weighted combination of key metrics: Mean Absolute Error (MAE), servo standard deviations (PanStd and SteerStd), saturation percentages (PanSat% and SteerSat%), and Centered%. The Score formula penalizes high error, jitter, and saturation while rewarding consistent centering behavior.[12] As a result, lower Score values indicate better overall performance reflecting smoother control, more accurate tracking, and minimal servo strain. This scoring system enables direct comparison across multiple PID configurations and helps identify the most stable and efficient tuning parameters for real-world deployment.

6. Conclusion

This paper details the development of a color-following robot using Raspberry Pi, camera-based HSV image processing, kinematics, and PID control. The system detects a target color, computes positional error, and adjusts motion in real time. The PID tuning enable smooth, responsive tracking, while kinematic modeling translates control signals into physical movement. Simulations validated the control framework, and real-world tests confirmed reliable performance. Metric analysis showed that optimal PID parameters minimize error and jitter, improving centering. The design offers a scalable, low-cost solution for autonomous visual tracking, with future work aimed at adaptive control and complex environments.

REFERENCES

[1] "Color-Based Image Processing for Autonomous Human Following Trolley Robot Navigation with Camera Vision," J. Comput. Sci. Eng. (JCSE), vol. 5, no. 1, pp. 20-38, Feb. 2024. [Online]. Available: http://icsejournal.com/index.php

[2] "Color based Object Tracking Robot," ResearchGate, Jun. 2018. [Online]. Available: https://www.researchgate.net/publication/325661932_Color_based_Object_Tracking_Robot

[3] S. M. R. Islam, S. Shawkat, S. Bhattacharyya, and M. K. Hossain, "Differential Drive Adaptive Proportional-Integral-Derivative (PID) Controlled Mobile Robot Speed Synchronization," Int. J. Comput. Appl., vol. 108, pp. 5–8, 2014.

[4] R. N. Jazar, Theory of Applied Robotics: Kinematics, Dynamics, and Control, 3rd ed. Cham, Switzerland: Springer Nature, 2022.

[5] M. Ben-Ari and F. Mondada, Elements of Robotics. Cham, Switzerland: Springer Open, 2018.

[6] E. Yilmazlar and H. Kuscu, "Object Tracking by PID Control and Image Processing On Embedded System," Int. J. Eng. Sci., vol. 6, no. 9, pp. 33-37, Sep. 2017. [Online]. Available: https://www.researchgate.net/publication/321996553

[7] "Position Control of a Mobile Robot Through PID Controller," Acta Univ. Cibiniensis – Tech. Ser., vol. 71, 2019.

[8] W. Budiharto, E. Irwansyah, J. S. Suroso, and A. A. S. Gunawan, "Design of Object Tracking for Military Robot Using PID Controller and Computer Vision," ICIC Express Lett., vol. 14, no. 3, pp. 243-250, Mar. 2020.

[9] M. Ben-Ari and F. Mondada, Elements of Robotics. Cham, Switzerland: Springer Open, 2018.

[10] N. Correll, B. Hayes, C. Heckman, and A. Roncone, Introduction to Autonomous Robots: Mechanisms, Sensors, Actuators, and Algorithms, v3.0, Dec. 2021.

[11] Raspberry i (Trading) Ltd., "Raspberry Pi 4 Model B Datasheet," Release 1.1, Mar. 2024.

[12] "What Is Man Absolute Error In Machine Learning," Roots.net. [Online]. Available: https://robots.net/fintech/what-is-mean-absolute-error-in-machine-learning/**Authors Biography**

Htet Wai Lin received the Bachelor of Computer Technology (B.C.Tech.) degree in 2023 from the University of Computer Studies, Mandalay, Myanmar. He is currently pursuing her Master of Computer Technology (M.C.Tech.) degree at University of Computer Studies, Mandalay.