

# Hybrid Model Based on Recurrent Neural Network and Gated Recurrent Unit for Text Processing

Areej Fatime<sup>1</sup>, \*Muhammad Usama Yousaf<sup>2</sup>, Ayesha Saddiqua<sup>3</sup>, Mahnoor Izzat<sup>4</sup>, Aiman Sohaib<sup>5</sup>,  
Sana Hassan<sup>6</sup>, Muhammad Adnan Kaim Khani<sup>7</sup>

**Abstract:** This work proposes a hybrid deep learning model combining Recurrent Neural Networks (RNN) and Gated Recurrent Units (GRU) for improved text processing. While RNNs capture sequential dependencies and GRUs address vanishing gradient issues through gating mechanisms, each has limitations when used independently. The hybrid model leverages the strengths of both architectures to enhance memory retention, contextual understanding, and overall performance on text-related tasks. The proposed model is evaluated on text classification tasks and compared against standalone RNN, GRU, and LSTM models. Experimental results show that the hybrid approach achieves higher accuracy and more coherent, context-aware outputs while reducing training time. By combining RNN's sequence modeling capability with GRU's efficient long-range dependency handling, the model overcomes common challenges in memorizing long sequences. The improved performance demonstrates practical benefits for applications such as chatbots, automated summarization, and sentiment analysis, where accurate and contextually relevant text processing is critical. The hybrid model also shows better efficiency in managing sequential data across multiple domains compared to traditional sequential models. Although transformer-based models currently lead in state-of-the-art benchmarks, this hybrid approach offers a computationally lighter alternative with competitive results, making it suitable for scenarios with limited resources. The dataset was split into multiple training and testing ratios to validate robustness, and evaluation across several performance metrics confirms that the proposed model outperforms individual RNN, GRU, and LSTM baselines. The results indicate that combining complementary recurrent architectures can yield a more effective and efficient solution for natural language understanding tasks.

**Keywords:** Recurrent neural network, Gated recurrent unit, Long Short Term Memory, Text classification, Sentiment analysis

## I. INTRODUCTION

Text processing encompasses automating text evaluation to extract structured data for analysis, text generation, and manipulation [21]. One prominent task within text processing is text classification, which involves categorizing text into different groups based on various criteria such as content, sentiment, or topic. Deep learning models classify text from a series of inputs [1].

In sentiment analysis, evaluate people's thoughts in textual data and extract their points of view. Sentiment analysis is divided into two classifiers. Binary classifiers deal with two values, positive or negative, while multi-classifiers classify text into multiple labels and multiple levels. Now discuss another form of text classification, as news is an important source of information news classifier helps the user to classify news into different categories, like business news, entertainment news, emergency news, politics news, and so on [2]. Text preprocessing is an essential step to building a machine learning model, and the quality of the model depends on how well the data has been preprocessed. Text documents are molded as transaction that holds noise in numerous forms like punctuation, emotions, and text in different circumstances. The purpose of the text preprocessing is to represent each document individually as a feature vector that splits the text into distinct words. The main text preprocessing steps are (tokenization, lower casing, stemming, lemmatization, stop words removal [3].

In text processing, the application of standard models often falls short of delivering optimal performance. In such cases, there arises a compelling need for enhancements or hybridizations to bolster their capabilities. Previous models encountered several challenges with recurrent neural networks (RNNs) [4]. One of the central issues that plagued RNNs is their inherent short-term memory problem. They struggle to effectively retain historical data in time series, leading to frequent data loss. To address this limitation, we find ourselves in search of a solution that can imbue our models with a form of "memory" that extends beyond their usual constraints. One of the immediate benefits of resolving this problem is the elimination of gradient issues that had previously plagued us. With the advent of the gated recurrent unit (GRU) [5], we find that it possesses the capacity to maintain long-term information dependencies. Despite its somewhat sluggish convergence, modest learning efficiency, and occasional issues with under-fitting, the GRU proves to be a valuable tool in tackling this recurrent neural network dilemma. It becomes increasingly evident that we require a model that can effectively address these challenges and provide us with the means to capture intricate and distant connections within our data.

## II. RELATED WORK

Basu, C. Watters, and M. Shepherd's research study to identify text in news articles. Compare text analyzers for news items using artificial neural networks and support vector machines

<sup>1</sup>Department of Information Technology, The University of Haripur

<sup>2,4,7</sup> Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman

<sup>6</sup> Faculty of Business and Finance, Universiti Tunku Abdul Rahman

<sup>3,5</sup> Institute of Computing, Kohat University of Science and Technology, Pakistan Country: Pakistan, Malaysia

Email: \*its.usamayousaf@gmail.com

first. 82% of the 600 test files were accurate [6]. According to Raghavendra Kumar Dwivedi's research publication, the precision, recall, and F1 score of the rule-based model (RBM). Our rule-based model (RBM) approach is based on these five rules and performs better than other solutions available in terms of precision (0.99), recall (0.78), and F1 score (0.83). [7].

Wei Zhang, in their research paper Increased classification accuracy is achieved by adjusting the corresponding conditional probability when naive Bayes is used. In terms of probability, the naive Bayes classifier is the best since it places the text in the class with the highest likelihood [8].

In this study, Fazli Wahid, Ramla Sheik, et al. used an enhanced convolutional neural network (eCNN) based on L1 regularization to autonomously divide coronary arteries. It has been noted that implementing L1 regularization with various values for coefficients has unique effects on the CNN architecture's operation, thereby improving the system's efficiency [9].

A stock trend prediction model that incorporates the Gate Recurrent Unit (GRU) and Attention mechanism was recently proposed by Ming-Che Lee et al. The search result reveals that the proposed model and labels created by this research are capable of successfully using financial commodities trading techniques and projecting major stock price movements. The average precision of predictions made with the GRU-Attention method may be about 70% [10].

In their research work, Murthy et al. suggested a method for classifying sentiments in text data based on long short-term memory (LSTM). In the vast majority of investigations from the published literature, the sentiment analysis model's accuracy is 85% [11], [20].

In their research article, M Aamir et al. suggested a model that was tested using four MNIST standard variant datasets. Four popular methods for multiclass classification—standard CAE, RBM, Alex Net, and artificial neural networks have been contrasted with the results. The performance of the suggested model has shown an apparent boost when compared to alternative categorization approaches.[12], [22].

In their study work, Combining KNN and the fine-grained algorithm, Jasmir, Siti Nurmaini, and Bambang Tutu Ko state that KNN research contributed to a reduction in the system throughput of KNN in clinical research texts, ranging from 388,274 s to 260,641 s, utilizing a dataset of clinical trial text with a data range of 1,000,000. The highest accuracy recall, precision, and F1score values were produced by the KNN with  $k = 5$  [13], [23].

Aamir Muhammad. Wahid Fazli et al proposed an enhanced version of CAE based on a layered architecture and a feed-forward mechanism called deep CAE in that research study. The outcomes have been evaluated in terms of training error, assessment error, and execution time with standard SAE, DAE, RBM, SCAE, ScatNet, and PCANet. The outcomes showed that the suggested model performed better than the previously described models [14], [24].

Harsh H. Patel and Purvi Prajapati proposed in their research paper that the accuracy and cost-effectiveness of various decision tree algorithms have been taken into consideration. There was only one dataset on which CART, ID3, C4.5, and other decision tree algorithms were applied. Trees of Decisions perform better than other algorithms in terms of precision, speed, and accuracy. According to this study, the most precise and accurate method for this dataset is CART [15], [25].

[IRFTC], or improved random forest for text classification, is a method that Nasir Jalal introduced in their work. It combines bootstrapping and heuristic search approaches. IRFTC outperforms classical RF by 6.28% for binary classification and by 4.98% in comparison to the best performer NB. IRFTC, on the other hand, performs 1.50 percent better than conventional RF and 3.3 percent higher than the second-best DT [16], [26].

Wahid, F, Ghazali, R., et al., in their present study, address the intended problem by utilizing genetic algorithm (GA) operators. A few common benchmark minimization and maximization functions have been subjected to the suggested approach, and the outcomes have been compared with traditional FA and GA. It is possible to see a notable increase in the convergence rate, which leads to excellent solutions for optimization issues [17], [27].

Merin Thomas and Latha C. A set of tweets in the Malayalam language of South India was subjected to sentimental analysis. Accuracy goals were found to rise with dataset depth and quality. RNN-LSTM modeling techniques are used to simulate sentiment classification. This model achieves an average accuracy of 80%. Having deeper datasets can still increase the accuracy of the results. Categorization method that uses exceptions [18], [28].

The classification performance of FRS-RNN is compared with several newly created models and state-of-the-art classical models such as SVM, KNN, Naive Bayes, DNN, CNN, and RNN. The suggested FRS-RNN outperforms all of the previously described classification models, according to experimental findings and empirical evaluation [19], [29].

### III. PROPOSED METHODOLOGY

Here's the proposed methodology for constructing and training this GRU-based model. The approach builds on two closely related recurrent architectures: the standard recurrent neural network (RNN) and the gated recurrent unit (GRU), a gated variant of the RNN cell. We compare their performance to find the best solution, aiming to improve the model's capacity to represent complicated and long-term dependencies in sequential data.

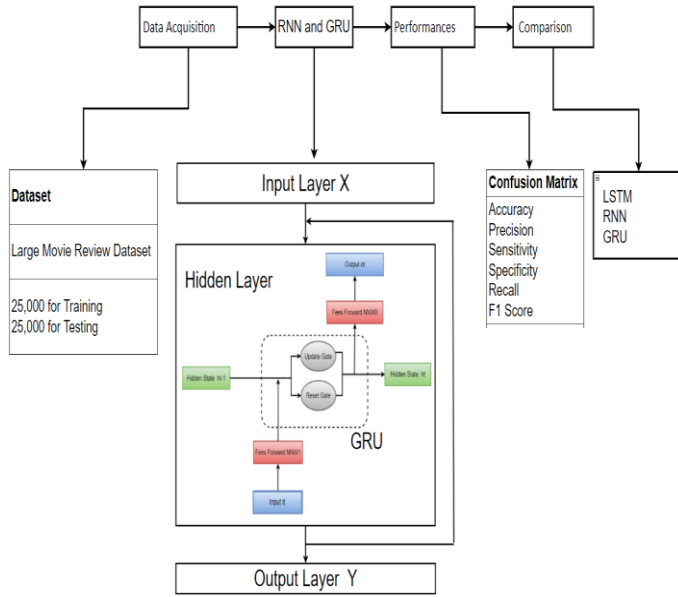


Figure 1 Proposed Methodology

### A. Data Acquisition

Data collection is one of the most crucial phases of a research project since it enables the researcher to address research questions. Data collection is the process of gathering information in order to understand more about the subject of the study. Different data collection techniques are suited for different sorts of data. However, depending on the type of data that will be utilized in the research.

### B. Preprocessing

Preprocessing data is a crucial step in enhancing the accuracy and value of the data for machine learning models. A hyperparameter is a parameter whose value is predetermined before the learning process begins. But training the data determines the values of other parameters.

### C. Model Design

First, we describe how the training and testing data are acquired and prepared. The second stage builds the recurrent classifier itself. Rather than treating RNN and GRU as two separate networks that are fused together, our model builds on the standard recurrent neural network (RNN) cell structure and replaces its plain hidden-state update with the gated recurrent unit (GRU) mechanism: an update gate and a reset gate that control how much of the previous hidden state is kept or overwritten at each time step. This gating is what lets the model retain longer-range dependencies than a plain RNN cell while remaining lighter than an LSTM. As shown in Figure 2, the resulting network still has the standard three-layer recurrent structure -- an input layer, a gated hidden layer, and an output layer -- but the hidden layer's internal update rule is GRU's, not a plain RNN's. In the third stage, performance evaluation parameters are used to analyze the proposed model's performance (recall, sensitivity, specificity, accuracy, precision, and F1 score) via the confusion matrix. In the fourth stage, we compare this GRU-based model against plain RNN, LSTM, and other baselines.

### Hybrid RNN and GRU

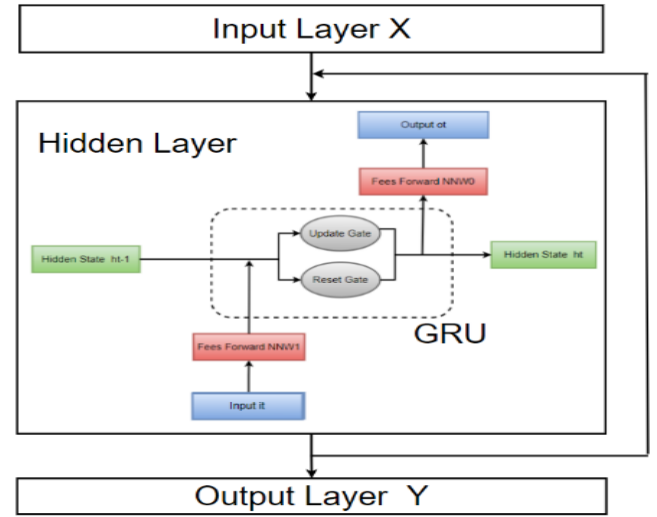


Figure 2: GRU cell structure (gated hidden layer)

## IV. PROPOSED ALGORITHM

This algorithm and associated code provide a structured approach to building a hybrid RNN and GRU model, leveraging the assets of both architectures to handle complex sequential data competently.

### A. Proposed Algorithm

Table 1: Proposed Hybrid Algorithm Stages

|                                                                                                                                                                                                                                                                                         |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Start                                                                                                                                                                                                                                                                                   |
| Stage 1: Input Gate Weights ( $W_i, b_i$ ) processing the input-                                                                                                                                                                                                                        |
| Stage 2: Update Gate Weights ( $W_z, b_z$ ), the update gate                                                                                                                                                                                                                            |
| $\setminus (z(t) \setminus)$ , which determines how much of the previous hidden state $\setminus (h(t-1) \setminus)$ and the new candidate hidden state $\setminus (h'(t) \setminus)$ should be combined to form the new hidden state $\setminus (h(t) \setminus)$ . –                  |
| Stage 3: Reset Gate Weights ( $W_r, b_r$ ) calculate the reset gate $\setminus (r(t) \setminus)$ , which decides how much of the previous hidden state $\setminus (h(t-1) \setminus)$ should be forgotten when calculating the candidate hidden state $\setminus (h'(t) \setminus)$ . – |
| Stage 4: Hidden State Weights ( $W_h, b_h$ ) compute the candidate hidden state $\setminus (h'(t) \setminus)$ by considering the current input $\setminus (x(t) \setminus)$ and the reset-modified previous hidden state $\setminus (r(t) \setminus \cdot h(t-1) \setminus)$ .          |
| Stage 5: output                                                                                                                                                                                                                                                                         |
| End                                                                                                                                                                                                                                                                                     |

TABLE 2: Pseudo Code

|                                                          |
|----------------------------------------------------------|
| Start                                                    |
| Initialize:                                              |
| - Weights and biases for the input gate ( $W_i, b_i$ )   |
| - Weights and biases for the update gate ( $W_z, b_z$ )  |
| - Weights and biases for the reset gate ( $W_r, b_r$ )   |
| - Weights and biases for the hidden state ( $W_h, b_h$ ) |
|                                                          |
| For each time step $t$ :                                 |
| Input:                                                   |
| - Previous hidden state $h(t-1)$                         |
| - Current input $x(t)$                                   |
|                                                          |
| Calculate reset gate:                                    |
| $r(t) = \text{sigmoid}(W_r * [h(t-1), x(t)] + b_r)$      |
|                                                          |
| Calculate update gate:                                   |
| $z(t) = \text{sigmoid}(W_z * [h(t-1), x(t)] + b_z)$      |
|                                                          |
| Calculate candidate hidden state:                        |
| $h'(t) = \tanh(W_h * [r(t) * h(t-1), x(t)] + b_h)$       |
| Update hidden state:                                     |
| $h(t) = (1 - z(t)) * h(t-1) + z(t) * h'(t)$              |
| END                                                      |

## V. EXPERIMENTAL RESULT

### A. System Specification

The system we're replicating has 4GB of RAM, an Intel(R) Core(TM) i3-5010U processor running at 2.10 GHz, and a 64-bit operating system, x64-based CPU. Python and Anaconda are the tools that are utilized for simulation-related tasks. TensorFlow and Keras' libraries for recurrent neural networks and gated recurrent units were employed.

### B. Dataset Description

<https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews>

There are 50,000 reviews of movies in this dataset, acquired from Kaggle, split evenly into 25,000 for training and 25,000 for testing, and each labeled as either positive or unfavorable. A set of word indexes has been used to encode each preprocessed

review (integers). The dataset's overall frequency of each phrase is used to index terms for convenience of usage. For instance, the third most frequent phrase in the dataset is encoded by the integer "3". As a result, simple filtering operations can be used, like "only consider the top 10,000 most frequent phrases but omit the top 20 most frequent words."

### C. Result Discussions

This section provides a brief discussion of the results of our two suggested models. Provides a quick overview of all the experiments conducted in relation to the hybrid paradigm for text processing founded on gated recurrent units (GRU) and recurrent neural networks (RNN). Our first experiment on a large movie review data set for the hybrid paradigm for text processing based on gated recurrent units and recurrent neural networks was trained on 80% of the randomly selected data and is evaluated on the remaining 20%. Compared to the unusual model, our accuracy was greater.

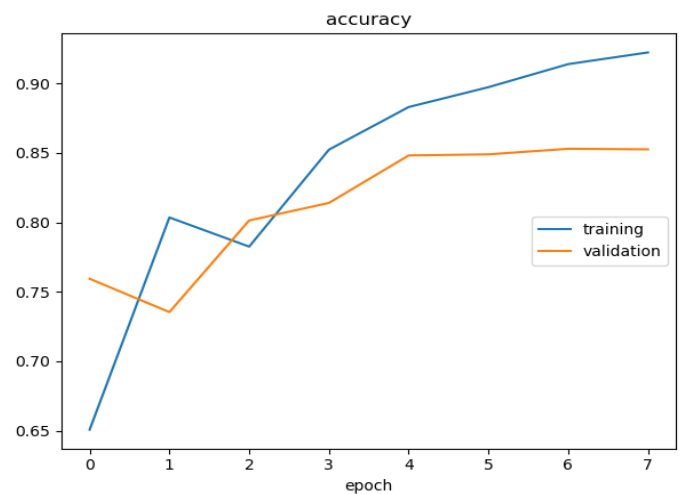


Figure 3. Training and Validation accuracy

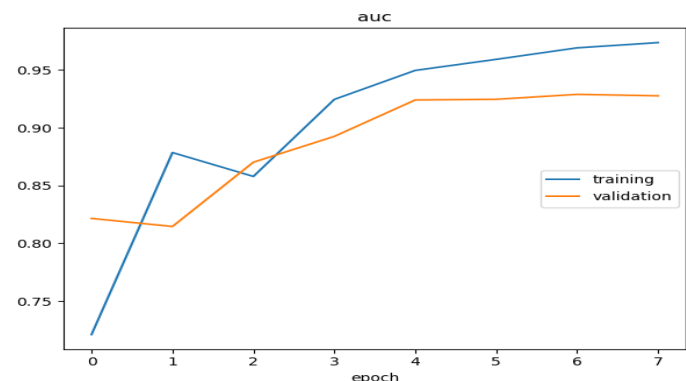


Figure 4. Training and Validation AUC

Figures 3 and 4 show the hybrid model's accuracy and AUC for dataset division. It took 7 epochs for the model to converge on a hybrid model. However, as the number of epochs increases over time, the outcomes progressively more accurate. The model provides the best results for training and testing data, indicating the highest performance, according to the accuracy and AUC

graphs. The training accuracy of our model throughout training is 92%, and validation is 85%.

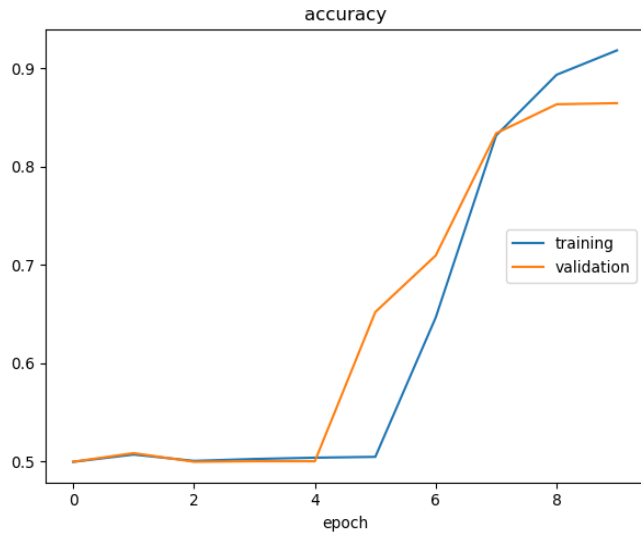


Figure 5: Training and Validation Accuracy

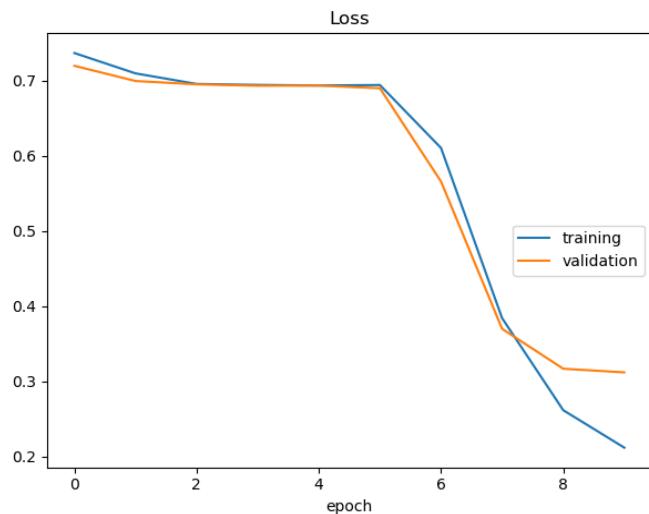


Figure 6: Training and Validation Loss

Figures 5 and 6 show the hybrid model's accuracy and loss for the second data split. Convergence took 8 epochs, with accuracy improving and loss dropping steadily as training progressed. The model provides the best results for training and testing data, indicating the highest performance, according to the accuracy and loss graphs. The hybrid model's validation accuracy 85%, and training accuracy is both 91%.

## VI. EVALUATION PARAMETERS

Scores of performance evaluation parameters for a large movie review dataset classification with 80-20%, 70-30%, 60-40% ratios are given below:

TABLE 3: Evaluation Parameters Across Train/Test Splits

| Dataset division | Training accuracy | Validation accuracy | Sensitivity | Specificity | precision | recall | F1 score |
|------------------|-------------------|---------------------|-------------|-------------|-----------|--------|----------|
| 80-20%           | 92%               | 85%                 | 92%         | 92%         | 92%       | 92%    | 92%      |
| 70-30%           | 91%               | 85%                 | 92%         | 92%         | 95%       | 92%    | 92%      |
| 60-40%           | 91%               | 85%                 | 92%         | 92%         | 95%       | 92%    | 92%      |

## VII. COMPARISON CHART

Comparative analysis of the proposed model with the previous model. The proposed method increased hybrid model accuracy and reduced training time. Our proposed model improved the training accuracy to 92%, and on unseen data, the accuracy is up to 86%, which was greater than other models. As compared with other techniques, such as RNN and GRU, our proposed model achieved the best performance.

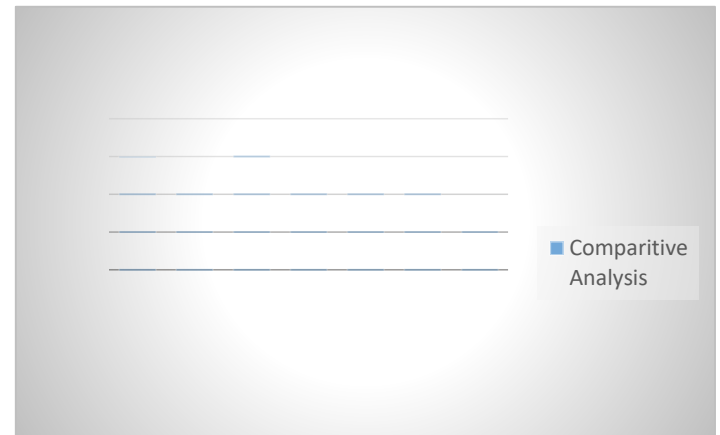


Figure 7: Comparison chart

## VIII. CONCLUSION AND FUTURE DIRECTION

In this study, a recurrent neural network (RNN) and gated recurrent unit (GRU)-based hybrid model for text processing using deep learning techniques has been proposed, which is a classified hybrid model. The project's goal was to raise performance related to the hybrid model of the training algorithm. Comparison of proposed work with other deep learning techniques, so we can see which is the best deep learning model for the memorization problem and gives the best performance. Our proposed model enhances the functionality and capabilities of chatbots, automated summarization, and sentiment analysis applications. Their combined strengths address limitations inherent in using either architecture alone, producing results that are more accurate, coherent, and contextually aware outputs. Better user experiences and more efficient management of sequential data across multiple domains are the outcomes of this. The dataset is divided into different training and testing ratios, and results are analyzed by passing through several performance attributes, which is why we were able to achieve a better accuracy rate in our research work and reduce training time. Our proposed model gives the best

performance, which was greater than that of other models. As compared with other techniques, such as LSTM, RNN, and GRU, making them more powerful and versatile for an extensive range of sequence-demonstrating tasks.

## REFERENCES

- [1] J. Mita, K. Dalal, and M. Zaveri, "Automatic Text Classification: A Technical Review," doi: 10.5120/3358-4633.
- [2] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep Learning-based Text Classification: A Comprehensive Review," *ACM Computing Surveys*, vol. 54, no. 3, Article 62, Apr. 2021, 40 pages, doi: 10.1145/3439726.
- [3] A. Bachayo, Z. Ahmed, & S. Affrah. (2022). A model design for smart home security system using (IoT) with CCTV camera," *International Journal of Computing and Related Technologies*, vol. 3, no. 2, pp. 29–42, 2022.
- [4] A. I. Kadhim, "An Evaluation of Preprocessing Techniques for Text Classification," *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 16, no. 6, June 2018.
- [5] M. A. Bamboat, A. A. Laghari, H. Li, A. A. Khan, and M. A. Qaimkhani, "Quality of experience assessment of over the top services," in *Proc. 2024 4th Int. Conf. Blockchain Technology and Information Security (ICBCTIS)*, Aug. 2024, pp. 145–149.
- [6] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.
- [7] A. S. Shah, A. Dinesh, A. Shah, M. Farooq, A. Maqsood, and M. A. K. Khani, "Smart gas leak detection and emergency response system using IoT for homes," *Journal of ICT, Design, Engineering and Technological Science*, vol. 8, no. 1, pp. 18–23, 2024.
- [8] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," in *Proc. 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.
- [9] H. Ali, M. A. K. Khani, A. S. Shah, A. A. Laghari, and M. Minam, "Smart class analytics for assessing group understanding via facial expressions," *Journal of Information Communication Technologies and Robotic Applications*, vol. 16, no. 1, 2025.
- [10] A. Basu, C. Watters, and M. Shepherd, "Support Vector Machines for Text Categorization," comparison of artificial-neural-network and support-vector-machine classifiers on news articles (82% accuracy on 600 test files).
- [11] M. A. K. Khani, A. A. Khan, A. B. Brohi, and Z. A. Shaikh, "Designing mobile learning smart education system architecture for big data management using fog computing technology," *International Journal of Imaging and Sensing Technologies and Applications (IJISTA)*, vol. 1, no. 1, pp. 1–23, 2022.
- [12] R. K. Dwivedi and M. Aggarwal, "Sentiment Analysis and Feature Extraction Using a Rule-Based Model (RBM)," doi: 10.1007/978-981-13-2354-6\_7.
- [13] A. Dashnyam, A. Peljidi, B. Natsag, A. Tseden-Ish, Z. Ali, M. A. K. Khani, and S. Hassan, "Evaluating block chain based voting systems: An empirical study on public administration adoption and challenges," *Pakistan Journal of Life & Social Sciences*, vol. 23, no. 2, p. 56, 2025.
- [14] W. Zhang and F. Gao, "An Improvement to Naive Bayes for Text Classification," doi: 10.1016/j.proeng.2011.08.404.
- [15] M. A. K. Khani, M. Usama, A. S. Shah, A. Shah, S. H. Abbas, A. Maqsood, and A. A. Laghari, "Intelligent vehicle number plate recognition system using YOLO for enhanced security in smart buildings," *Journal of ICT, Design, Engineering and Technological Science*, vol. 8, no. 2, pp. 11–17, 2024.
- [16] F. Wahid, Y. Ma, R. Sheikh, and M. Aamir, "An Enhanced Convolutional Neural Network Based on L1 Regularization for Segmentation of Coronary Arteries in X-Ray Angiograms," doi: 10.1109/ICACS60934.2024.10473261.
- [17] A. A. Laghari, H. Li, A. A. Khan, Y. Shoulin, S. Karim, and M. A. K. Khani, "Internet of Things (IoT) applications security trends and challenges," *Discover Internet of Things*, vol. 4, no. 1, Art. no. 36, 2024.
- [18] M.-C. Lee, "Research on the Feasibility of Applying GRU and Attention Mechanism Combined with Technical Indicators in Stock Trading Strategies," *Applied Sciences*, vol. 12, no. 3, p. 1007, Jan. 2022, doi: 10.3390/app12031007.
- [19] A. S. Shah, A. Maqsood, A. Shah, M. A. K. Khani, J. Anjum, and S. Zafar, "Enhanced airport operations: Automated baggage drop-off and boarding pass generation for travelers," *Journal of Advances in Technology and Engineering Research*, vol. 10, no. 2, pp. 1–6, 2024.
- [20] G. S. N. Murthy, S. R. Allu, B. Andhavarapu, M. Bagadi, and M. Belusonti, "Text-based Sentiment Analysis using LSTM," *International Journal of Engineering Research and Technology*, vol. 9, no. 5, May 2020, doi: 10.17577/ijertv9is050290.
- [21] N. Malik, M. A. Kaimkhani, A. A. Shaikh, A. B. Brohi, and Luhrani, "MCQ's evaluation using Python OCR: An algorithmic implementation and design approach," *International Journal of Artificial Intelligence & Mathematical Sciences*, vol. 2, no. 1, pp. 37–52, 2023.
- [22] M. Aamir, N. M. Nawi, F. Wahid, and H. Mahdin, "A Deep Contractive Autoencoder for Solving Multiclass Classification Problems," doi: 10.1007/s12065-020-00424-6.
- [23] S. Jasmir, S. Nurmaini, and B. Tutuko, "Fine-grained algorithm for improving KNN computational performance on clinical trials text classification," *Big Data and Cognitive Computing*, vol. 5, no. 4, Art. no. 60, 2021.
- [24] M. Aamir, N. M. Nawi, F. Wahid, M. S. H. Zada, M. Z. Rehman, and M. Zulqarnain, "Hybrid Contractive Auto-Encoder with Restricted Boltzmann Machine for Multiclass Classification."

- [25] H. H. Patel and P. Prajapati, "Study and Analysis of Decision Tree Based Classification Algorithms."
- [26] N. Jalal, A. Mehmood, and G. S. Choi, "A Novel Improved Random Forest for Text Classification Using Feature Ranking and Optimal Number of Trees."
- [27] F. Wahid and R. Ghazali, "A Hybrid Approach of Firefly and Genetic Algorithm for Solving Optimisation Problems," *International Journal of Computer Aided Engineering and Technology*, vol. 14, no. 1, pp. 62–79, 2021.
- [28] M. Thomas and C. A. Latha, "Sentimental Analysis Using Recurrent Neural Network," doi: 10.14419/ijet.v7i2.27.12635.
- [29] B. Behera and G. Kumaravelan, "Text Document Classification Using Fuzzy Rough Set Based on Robust Nearest Neighbor," doi: 10.1007/s00500-020-05410-9.