A Comparative Analysis of Machine Learning and Neural Network Models for Text Classification

Dibya Ayishwarja Mallick
BRAC University
Dhaka, Bangladesh.
[1]dibya.aishwarja.mallick@g.bracu.ac.bd

Fayruz Tahania Haseen
BRAC University
Dhaka, Bangladesh.
[1]fayruz.tahania.haseen@g.bracu.ac.bd

MD. Faiyaz
BRAC University
Dhaka, Bangladesh.
[1]md.faiyaz@g.bracu.ac.bd

***Abstract*— This project presents a comparative analysis of machine learning and deep learning models for multi-class text classification on a Question-Answer dataset. A range of feature representation techniques were evaluated, including Bag-of-Words (BoW), TF-IDF, custom-trained Skip-gram embeddings, and pre-trained GloVe vectors. These features were used to train ten distinct models, from classical algorithms like Logistic Regression to advanced architectures such as LSTMs and GRUs. Models were evaluated on accuracy and F1-score. The empirical results demonstrate that the Bidirectional GRU (BIGRU) model combined with GloVe embeddings achieved the highest performance, yielding a weighted F1-score of 63.02%. This finding highlights the effectiveness of pre-trained contextual embeddings and bidirectional recurrent architectures for capturing the semantic complexities inherent in QA text classification.**

***Keywords*— Deep Learning, Machine Learning, Natural Language Processing (NLP), Recurrent Neural Networks (RNN), Text Classification, Word Embeddings.**

I. Introduction

In the current digital era, the exponential growth of unstructured text data necessitates advanced techniques for its management and analysis. A fundamental task within the field of Natural Language Processing (-NLP) is text classification, which involves assigning predefined labels to textual content. This process is crucial for transforming unstructured data into a structured format, enabling scalable analysis and information retrieval. The applications of text classification are diverse, ranging from automated spam detection and sentiment analysis to content moderation and topic categorization.

The motivation for this research is rooted in the practical challenge of organizing vast repositories of user-generated content on Question-Answer (QA) platforms. Effective categorization of QA pairs is essential for enhancing user experience, facilitating efficient information discovery, and enabling analytics on user engagement and content gaps. This study aims to develop and systematically evaluate a robust text classification system by comparing a comprehensive suite of machine learning (ML) and deep learning (DL) models. The objective is to move beyond rudimentary keyword-based methods and leverage algorithms capable of understanding the semantic context of the text.

The dataset utilized in this work consists of question-answer pairs, pre-partitioned into training and testing sets. Each data point comprises a text block with a 'Question Title', 'Question Content', and a 'Best Answer', along with a corresponding class label. The dataset is multi-class, containing distinct categories. An initial exploratory data analysis revealed a notable imbalance in the class distribution, as depicted in Fig. 1.

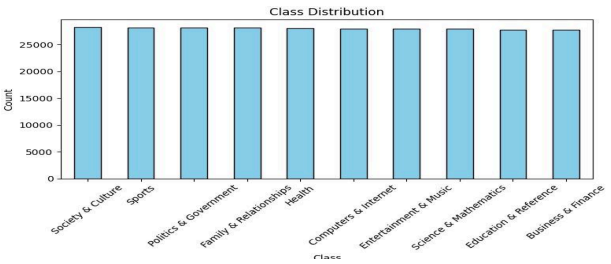


Fig. 1. Distribution of classes within the training dataset.

This observed class imbalance, where certain categories are significantly more frequent than others, is a critical factor influencing model design and evaluation. Consequently, metrics that account for this imbalance, such as the weighted F1-score, are essential for a fair and accurate assessment of model performance. The primary goal of this research is to determine the most effective combination of word representation techniques and classification models to accurately predict the class of a given QA text.

II. Methodology

This section details the systematic approach for data processing, feature engineering, and model implementation. The methodology is organized into four main stages: exploratory data analysis, data preprocessing, word representation, and model architectures.

A. Exploratory Data Analysis (EDA)

An initial exploratory data analysis was conducted to ascertain the intrinsic properties of the dataset. This analysis focused on two primary aspects: class distribution and text length. The class distribution analysis, as shown in the Introduction (Fig. 1), revealed a balanced distribution among the categories. This finding confirmed that standard accuracy can be used as a reliable primary evaluation metric alongside the F1-score.

B. Data Preprocessing

A rigorous preprocessing pipeline was applied to the raw text to generate a clean and normalized corpus. The pipeline comprised the following steps:

1. Normalization: All text was converted to lowercase, and recurring textual artifacts such as "Question Title" were programmatically removed.

2. Character Filtering: All non-alphabetic characters, including punctuation and numerical digits, were removed.

3. Tokenization: The cleaned text was segmented into individual tokens.

4. Stopword Removal: Common English stopwords were removed using the NLTK library's standard list.

5. Lemmatization: Tokens were reduced to their dictionary base form using the WordNet Lemmatizer to consolidate word variations into a single feature.

6. Stemming: Cut the length of a word to make the word in its base form.

C. Word Representation Techniques

Four distinct techniques were implemented to convert the preprocessed text into numerical vectors for model training.

1. Bag-of-Words (BoW): A document-term matrix was created based on word frequencies. The vocabulary was limited to the 5,000 most frequent terms.

2. Term Frequency-Inverse Document Frequency (TF-IDF): A weighted representation was generated that emphasizes words that are frequent in a document but rare in the overall corpus. The feature space was also limited to 5,000 unigram and bigram features.
3. Word2Vec (Skip-gram): A custom Word2Vec model was trained on the project's corpus to learn 100-dimensional dense vector embeddings that capture semantic relationships between words.

4. GloVe (Global Vectors for Word Representation): Pre-trained, 100-dimensional GloVe embeddings, trained on a large external corpus, were utilized to provide a robust semantic foundation.

D. Model Architectures

Ten models were implemented, spanning classical machine learning and deep learning paradigms.

1. Machine Learning Models: Three algorithms were trained using both BoW and TF-IDF feature sets: Logistic Regression, Multinomial Naive Bayes, and Random Forest.

2. Neural Network Models: Seven neural network architectures were developed. The common structure included an Embedding Layer (initialized with Word2Vec or GloVe vectors and kept non-trainable), a recurrent layer with 64 units (SimpleRNN, GRU, LSTM, or their Bidirectional variants), a Dense layer with ReLU activation, and a final Softmax output layer. To mitigate overfitting, Dropout regularization and an Early Stopping mechanism based on validation loss were employed.

E. Hyperparameters Tuning

In this study, hyperparameter tuning was performed to optimize both traditional machine-learning models and neural network–based architectures for the text-classification task. For the machine-learning models, empirical tuning focused primarily on adjusting the feature-extraction configurations, specifically Bag-of-Words (BoW) and TF-IDF representations. Different vocabulary sizes and n-gram ranges were explored to improve the discriminative capacity of Logistic Regression, Naive Bayes, and Random Forest classifiers. Logistic Regression performance improved by refining TF-IDF parameters, while Naive Bayes benefited from changes to BoW and TF-IDF tokenization settings. Similarly, Random Forest showed performance variation depending on the quality of handcrafted features derived from BoW and TF-IDF. For the neural network models including RNN, BiRNN, LSTM, BiLSTM, GRU, BiGRU, and deep feed-forward models trained on GloVe, skip-gram, and TF-IDF/BoW embeddings. Multiple hyperparameters were iteratively tuned to enhance learning stability and generalization. Experiments evaluated different epoch counts, batch sizes, dropout rates, and the number of neurons per layer. Early tests demonstrated sensitivity to overfitting, leading to the introduction and gradual adjustment of dropout; dropout values were varied before converging on 0.5. Similarly, the number of neurons was optimized and ultimately standardized at 128 to balance model complexity and computational cost. Batch sizes were tested across multiple settings, and 32 was selected as the most stable. Epochs were initially increased to observe performance trends but plateaued, so the training duration was fixed at 5 epochs to prevent overfitting while maintaining efficiency. These systematic adjustments provided a consistent hyperparameter configuration across neural models, ensuring fair comparison while retaining strong performance.

III. Results

TThis section presents the empirical results of the classification experiments. The performance of each model is evaluated using accuracy and the F1-score. Given the balanced nature of the dataset, both metrics provide a reliable measure of model performance.

A. Overall Performance Summary

The consolidated results for all experimental configurations are presented in TABLE I. The models are ranked in descending order based on their F1-score to facilitate a clear comparison between the different word representation techniques and model architecture

TABLE I

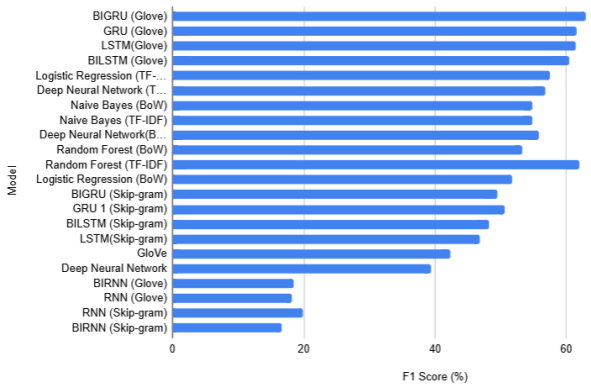
COMPARATIVE PERFORMANCE OF ALL TRAINED MODEL

Model	Accuracy (%)	F1 Score (%)
BIGRU (Glove)	62.66	63.02
GRU (Glove)	61.02	61.54
LSTM (Glove)	61.06	61.5
BILSTM (Glove)	60.88	61.18
Logistic Regression (TF-IDF)	56.79	57.54
Deep Neural Network (TF-IDF)	56.11	56.73
Naive Bayes (BoW)	55.19	54.88
Naive Bayes (TF-IDF)	54.82	54.81

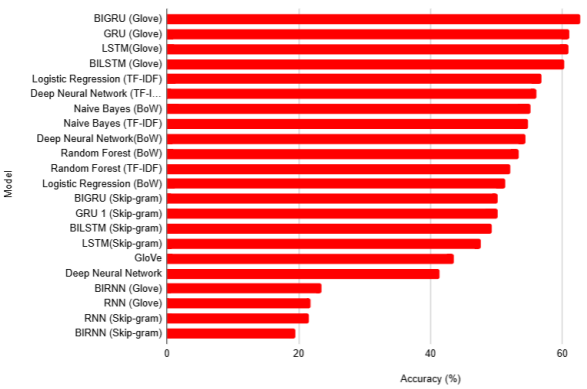
Deep Neural Network (BoW)	55.29	5698
Random Forest (BoW)	53.43	53.21
Random Forest (TF-IDF)	52.13	62.04
Logistic Regression (BoW)	51.37	51.8
BIGRU (Skip-gram)	50.17	49.42
GRU (Skip-gram)	50.63	50.84
BILSTM (Skip-gram)	49.25	48.21
LSTM (Skip-gram)	50.49	50.98
Deep Neural Network (Glove)	43.54	42.37
Deep Neural Network (Skip-gram)	41.3	39.42
BIRNN (Glove)	23.44	18.5
RNN (Glove)	25.64	21.83
RNN (Skip-gram)	21.52	19.82
BIRNN (Skip-gram)	25.15	24.17

From the table, a clear performance hierarchy emerges. The top-performing models are dominated by neural network architectures using GloVe embeddings, with the BIGRU (Glove) model achieving the highest F1-score of 63.02%. Notably, the Random Forest (TF-IDF) model also performed exceptionally well, securing the second-highest F1-score at 62.04%, proving to be highly competitive with the more complex deep learning models. At the lower end of the spectrum, the SimpleRNN-based models (RNN and BIRNN) performed poorly regardless of the embedding used, with F1-scores below 20%.

B. Performance Visualization



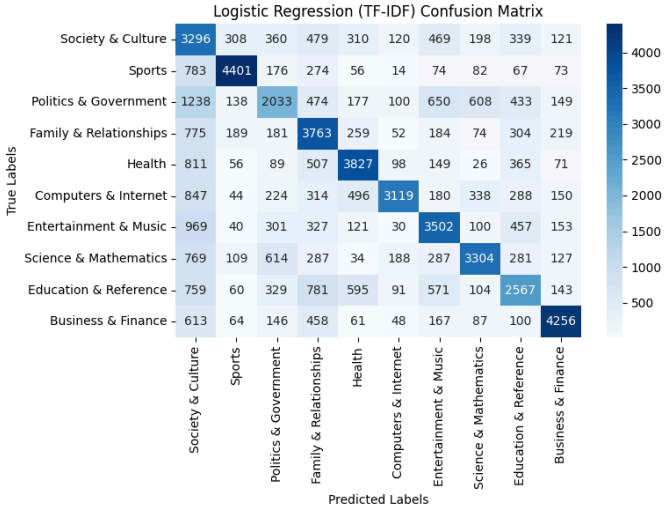
"Fig. 2. Comparison of F1-Scores Across All Model Configurations."

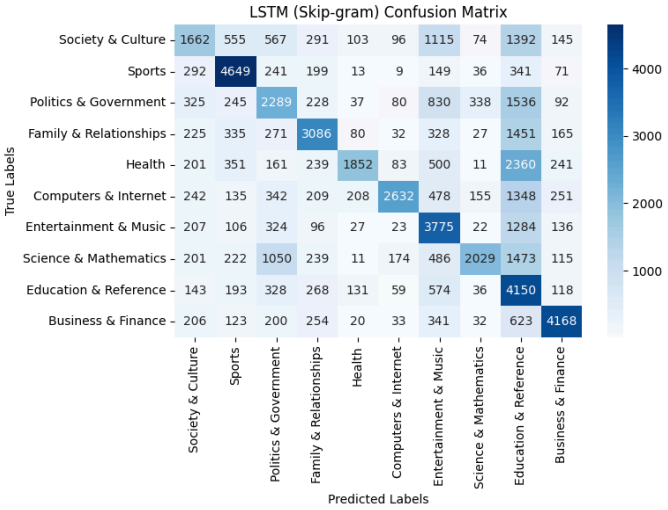


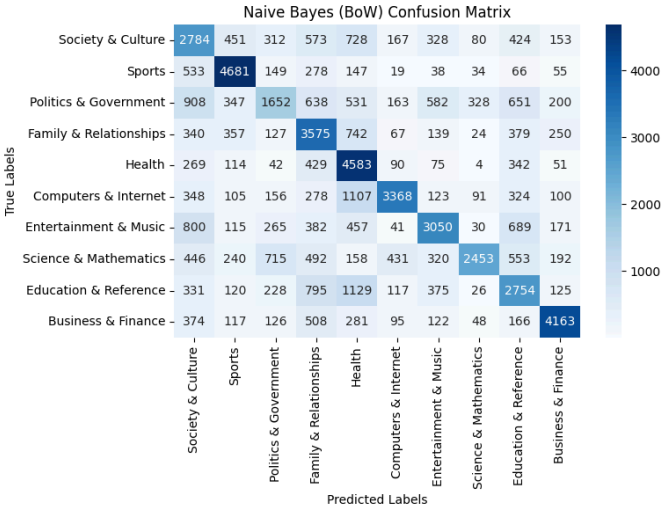
"Fig. 3. Comparison of Accuracy Across All Model Configurations."

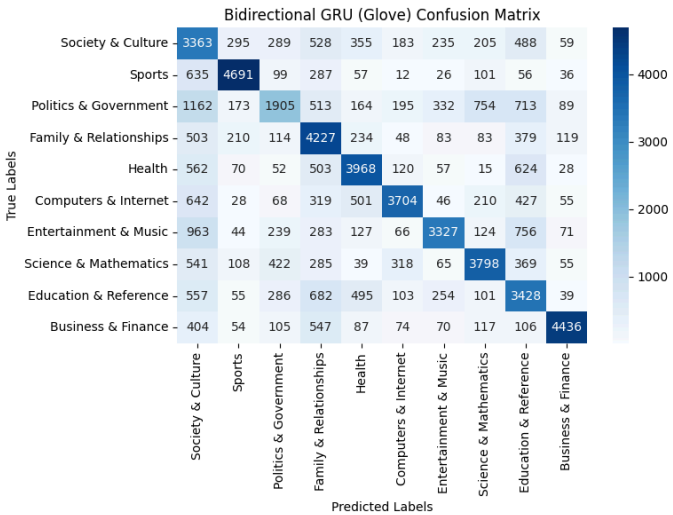
The visualization distinctly shows three tiers of performance. The top tier consists of GRU and LSTM-based models with GloVe embeddings and the Random Forest (TF-IDF) model. The middle tier is composed of classical ML models using BoW, Naive Bayes, and the DNN models. The bottom tier is clearly occupied by the SimpleRNN-based architectures, which failed to learn effective patterns from the data.

To further interpret model performance beyond aggregate metrics, confusion matrices were generated for the best-performing models in each representation category: Logistic Regression (TF-IDF), Naive Bayes (BoW), BiGRU (GloVe), LSTM (Skip-gram). For the machine-learning models, confusion matrices showed strong overall accuracy, though misclassifications were more common among semantically related categories such as *Society & Culture* and *Family & Relationships*. The BiGRU with GloVe exhibited the most consistent class-wise performance, while the LSTM with Skip-gram also demonstrated solid contextual discrimination despite occasional overlap between closely related topics. Overall, the confusion matrices provided a clear visualization of class-wise strengths and weaknesses, helping highlight category-specific misclassification patterns across model types.









C. Comparative Analysis

ML Models (BoW vs. TF-IDF): Among the classical machine learning models, those trained on TF-IDF vectors consistently outperformed their Bag-of-Words counterparts. For instance, Logistic Regression with TF-IDF achieved an F1-score of 57.54%, a notable improvement over the 51.80% achieved with BoW. This suggests that the TF-IDF weighting scheme, which emphasizes more discriminative words, was more effective for this classification task than raw term frequency.

NN Models (Embeddings and Architectures): The deep learning models demonstrated a strong dependency on the quality of word embeddings. Models using pre-trained GloVe embeddings consistently and significantly yielded better results than those using the custom-trained Skip-gram embeddings. For example, the BIGRU with GloVe scored 63.02% on the F1-metric, whereas with Skip-gram it scored only 49.42%. This wide margin indicates that the broader semantic knowledge captured in the pre-trained GloVe vectors was highly beneficial and that the project's training corpus was likely insufficient to train powerful custom embeddings from scratch.

Furthermore, architectural complexity played a critical role. The SimpleRNN and BIRNN models were the worst-performing of all, indicating their inability to handle long-range dependencies in the text. In contrast, LSTM and GRU models, which are designed to mitigate this issue, performed substantially better. The use of bidirectional

processing (e.g., BIGRU) provided a slight but consistent advantage over unidirectional architectures (e.g., GRU), confirming the value of processing text in both forward and backward directions.

IV. Conclusion

This study conducted a comprehensive comparative analysis of ten different machine learning and deep learning models for the task of multi-class text classification on a Question-Answer dataset. The experiments systematically evaluated four distinct word representation techniques: Bag-of-Words, TF-IDF, custom-trained Word2Vec (Skip-gram), and pre-trained GloVe embeddings.

The key findings indicate that sophisticated neural network architectures, when paired with high-quality pre-trained embeddings, are highly effective for this task. The Bidirectional GRU (BIGRU) model using GloVe embeddings emerged as the top-performing model, achieving a weighted F1-score of 63.02%. This highlights the significant advantage of leveraging contextual information from both forward and backward text sequences. Notably, the classical Random Forest model with TF-IDF features also demonstrated remarkable performance, securing an F1-score of 62.04%, proving to be a highly competitive and computationally less expensive alternative. The results consistently showed that pre-trained GloVe embeddings outperformed the custom-trained Word2Vec model, and TF-IDF was superior to Bag-of-Words for the classical algorithms.

However, a significant challenge was encountered in pushing the model accuracies higher. Initial experiments utilizing the full scope of the dataset led to a drop in performance, which may be attributed to overfitting on noisy or complex examples within the training data. To mitigate this and achieve the best possible outcomes, various hyperparameters were tuned and different feature space limitations were explored. Despite these efforts, the overall accuracy indicates that the task remains complex.

Future work could focus on several promising areas. First, implementing more advanced, state-of-the-art architectures like Transformers (e.g., BERT) could potentially yield a significant performance boost by capturing deeper contextual relationships. Second, more sophisticated regularization techniques and a more extensive, automated hyperparameter search could help to better control overfitting. Finally, exploring domain-specific pre-trained embeddings or fine-tuning existing ones on the target dataset may provide a stronger foundation for the neural network models.

References

[1] T. E. Oliphant, NumPy: A Guide to NumPy. USA: Trelgol Publishing, 2006. [Online]. Available: https://numpy.org/

[2] P. Virtanen et al., "SciPy 1.0: fundamental algorithms for scientific computing in Python," Nature Methods, vol. 17, pp. 261–272, 2020. [Online]. Available: https://scipy.org/

[3] R. Řehůřek and P. Sojka, "Gensim – Statistical Semantics in Python," Natural Language Processing Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 2010. [Online]. Available: https://radimrehurek.com/gensim/

[4] W. McKinney, pandas: a foundational Python library for data analysis and statistics, Python Software Foundation, 2010. [Online]. Available: https://pandas.pydata.org/

[5] J. D. Hunter, "Matplotlib: A 2D graphics environment," Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007. [Online]. Available: https://matplotlib.org/

[6] M. Waskom, "Seaborn: statistical data visualization," Journal of Open Source Software, vol. 6, no. 60, p. 3021, 2021. [Online]. Available: https://seaborn.pydata.org/

[7] A. Mueller, "Wordcloud for Python," GitHub repository, 2012. [Online]. Available: https://github.com/amueller/word_cloud

[8] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python. O'Reilly Media Inc., 2009. [Online]. Available: https://www.nltk.org/

[9] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://scikit-learn.org/

[10] F. Chollet et al., Keras. GitHub, 2015. [Online]. Available: https://keras.io/

[11] TensorFlow Developers, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. [Online]. Available: https://www.tensorflow.org/

[12] Python Software Foundation, Python: A programming language, 1991. [Online]. Available: https://www.python.org/