International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

A Hybrid Architecture for Multi-Category Text Classification Using BERT and Graph Embeddings

[1, *]Vipin Kataria, [2, *]Nitin Kumar

[1]*Picarro Inc, Santa Clara, California, USA*
[2]*Marriott International, Bethesda ,Maryland, USA*
**These Authors contribute equally to the Paper*

Abstract - Text classification, the task of assigning predefined categories to textual data, has become increasingly vital in the digital age as organizations struggle to manage and extract value from vast amounts of unstructured information. This study explores a novel hybrid architecture for multi-category text classification called BEGNN (BERT-Enhanced Graph Neural Network), which integrates the semantic richness of BERT embeddings with the structural capabilities of graph neural networks. The research applies this approach to a dataset containing 2,225 text samples across five distinct categories: politics, sport, technology, entertainment, and business. The proposed BEGNN architecture processes text through parallel pathways - extracting contextual semantic features via BERT while simultaneously modeling document structure through graph representations - before integrating these complementary features for classification. Experimental results demonstrate the superior performance of our approach, achieving 99% accuracy, precision, recall, and F1-scores across all categories, outperforming established models including traditional machine learning methods (SVM, Logistic Regression) and other deep learning approaches (BERT, BiLSTM). The confusion matrix analysis reveals exceptional classification capability with minimal misclassifications, particularly for Sport and Business categories. This research contributes to the advancement of text classification by effectively combining semantic and structural text representations, offering significant improvements for applications requiring high precision in document categorization.

Key Words: Text Classification, BERT, Graph Neural Networks, Natural Language Processing, Multi-category Classification, Hybrid Architecture, Semantic Feature Extraction, Structural Feature Extraction, Co-Attention Mechanism, Deep Learning, News Categorization, Document Classification, BEGNN

1. Introduction

Text classification, the task of assigning predefined categories to free text documents, remains a cornerstone challenge in natural language processing (NLP) with wide-ranging applications across information retrieval, content recommendation, spam filtering, sentiment analysis, and topic detection. As digital content continues to proliferate, automated methods for organizing and categorizing textual information have become increasingly vital. The classification of documents into topical categories (such as politics, sports, technology) represents a particularly important application domain that enables efficient content navigation, targeted information delivery, and improved search functionality. News aggregators, content management systems, and digital libraries all rely on accurate text classification to organize their collections and serve relevant content to users.

Text classification is a critical component in the field of Natural Language Processing (NLP) due to its ability to efficiently organize and manage the vast amounts of textual data generated daily, particularly with the growth of the internet [1] [2]. It involves categorizing text into predefined classes based on content, which is essential for applications such as spam filtering, sentiment analysis, news classification, and more [3]. The importance of text classification is underscored by its application in diverse domains, including special education, where it aids in diagnosing and categorizing learning disabilities using NLP tools [4].

Traditional methods in text classification primarily involve statistical and machine learning techniques that rely on manual feature extraction and simpler models compared to modern deep learning approaches. Common traditional methods include Naive Bayes, Logistic Regression, and Support Vector Machines (SVM), which are valued for their efficiency and simplicity in implementation [5]. These methods typically utilize feature extraction techniques such as Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) to convert text into numerical representations that can be processed by classifiers [5] [6]. Despite their effectiveness in thematic classification tasks, traditional methods often struggle with capturing complex semantic features due to their reliance on word frequency and statistical measures, which can overlook the deeper semantic relationships within text [7] [8]. For instance, the vector space model, a common approach in traditional text classification, tends to emphasize frequent words, potentially disregarding low frequency but contextually significant terms [8]. Additionally, traditional methods like the Associative Rule-based Classifier by Category (ARC-BC) have been developed to improve classification by generating association rules between words and categories, demonstrating comparable performance to Naive Bayes [9]. However, these methods generally require manual feature engineering, which can introduce subjective biases and limit their adaptability to diverse datasets [10]. While traditional methods are less computationally demanding and easier to implement, they are often outperformed by deep learning models in terms of accuracy, particularly in tasks requiring nuanced

understanding of text semantics [5] [11]. Nonetheless, they remain a practical choice in scenarios where computational resources are limited or when simplicity and interpretability are prioritized over the highest possible accuracy [5]. The impact of text classification is further highlighted by its role in improving information retrieval systems, which are crucial for accessing and utilizing data efficiently [12].

Various machine learning algorithms, such as Naive Bayes, Support Vector Machines (SVM), and Convolutional Neural Networks (CNN), have been employed to enhance classification accuracy, with SVM achieving up to 93.03% accuracy in certain applications [4]. The use of word embeddings and deep learning models like Recurrent Neural Networks (RNN), CNN, and Hierarchical Attention Networks (HAN) has further advanced the field by capturing semantic and contextual information, thereby improving classification performance [1] [13] [14]. However, challenges such as scalability and the need for effective feature selection remain, necessitating ongoing research to develop more reliable and efficient text classification systems [15] [3]. Additionally, techniques like text data augmentation have been shown to mitigate overfitting and enhance model accuracy, particularly in scenarios with limited data [16]. Overall, text classification is indispensable for managing the ever-increasing volume of textual data, with significant implications for both academic research and practical applications across various sectors [17]. This study makes several significant contributions to the field of text classification on multi-category data:

1. Integration of semantic and structural features: The proposed BEGNN architecture uniquely combines the contextual semantic understanding of BERT with the structural relationship modeling of graph neural networks, capturing both the meaning of words and their relationships within documents.
2. Co-attention mechanism: The study introduces a novel co-attention mechanism that enables bidirectional information flow between the semantic and structural branches, creating a more comprehensive document representation.
3. Superior performance: The experimental results demonstrate exceptional classification accuracy (99%) across all five categories (politics, sport, technology, entertainment, and business), outperforming traditional machine learning methods and other deep learning approaches.
4. Minimal misclassification: The confusion matrix analysis reveals the model's exceptional classification capability with very few errors, particularly for Sport and Business categories.

Practical application framework: The architecture provides a flexible and robust solution for applications requiring high-precision document categorization in domains such as news aggregation, content management systems, and digital libraries."

2. Related Work

Text classification is a fundamental task in natural language processing (NLP) that involves assigning predefined categories to text documents. The theoretical foundations of text classification are rooted in machine learning and statistical modeling. Early approaches relied heavily on rule-based systems and manual feature engineering, while modern methods leverage deep learning and automated feature extraction [11] [18]. Text classification can be broadly defined as the process of mapping text documents to predefined categories based on their content. This task is essential for various applications, including sentiment analysis, spam detection, and topic modeling. The theoretical underpinnings of text classification involve understanding the relationship between text features and target labels, which can be approached using supervised learning techniques [19] [5].

Feature Extraction and Representation

Feature extraction is a critical step in text classification, as it transforms raw text into a format that can be processed by machine learning algorithms. Traditional methods include Bag-of-Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF), which represent text as sparse vectors based on word frequencies. These methods are simple and efficient but fail to capture semantic relationships between words [20] [21].

Modern approaches, on the other hand, utilize word embeddings such as Word2Vec and GloVe, which capture semantic similarities between words in a dense vector space. These embeddings have become a cornerstone of modern NLP, enabling models to better understand the context and meaning of text [22] [23].

Historical Overview of Text Classification

The evolution of text classification can be divided into three main eras: traditional methods, shallow learning, and deep learning. Traditional Methods (1961-2000) The earliest text classification methods relied on rule-based systems and manual feature engineering. These methods were simple but limited in their ability to handle complex patterns in text. The introduction of Naive Bayes and SVM in the late 1990s marked the beginning of machine learning-based approaches [11] [18]. Shallow Learning (2000-2010) The shallow learning era saw the rise of machine learning algorithms such as logistic regression, decision trees, and random forests. These methods improved upon traditional approaches by automating feature selection and handling non-linear relationships. However, they still relied on manual feature engineering and struggled with high-dimensional data [19] [21]. Deep Learning (2010-Present) The advent of deep learning revolutionized text classification by enabling automatic feature extraction and handling of sequential data. Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), and Transformer-based models such as

BERT have achieved state-of-the-art performance in various text classification tasks. These models can capture long-range dependencies and semantic nuances in text [22] [23]. Traditional Methods Traditional methods are well-suited for small to medium-sized datasets and provide interpretable results. However, they struggle with high-dimensional data and complex patterns [5] [24]. Deep Learning Models Deep learning models excel in handling large datasets and capturing complex patterns. However, their computational demands and need for large amounts of labeled data can be a limitation in resource-constrained environments [22] [21].

Applications of Text Classification

Text classification has a wide range of applications across various domains, including sentiment analysis, which determines the sentiment of text as positive, negative, or neutral and is widely used in customer feedback analysis and social media monitoring [19][5]; spam detection, which classifies emails or messages as spam or non-spam, helping filter unwanted content and improve user experience [19][23]; topic modeling, which categorizes documents into topics such as sports, politics, or technology, making it useful for organizing large collections of text data [22][21]; and medical diagnosis, where text classification is applied for tasks such as disease diagnosis and medical document classification, helping improve healthcare outcomes and streamline clinical workflows [25][26].

Table 1 provides a summary of the key models discussed, their descriptions, and the range of accuracy reported in the literature.

Table 1: Summary of key models**.**

Model Type	Description	Performance Metrics Citation
Naive Bayes	Probabilistic classifier based on Bayes' theorem	Accuracy: 74.49% - 96.86% [5] [24] [26]
Support Vector Machine (SVM)	Linear classifier that maximizes the margin between classes	Accuracy: 74.49% - 96.86% [5] [24] [26]
Logistic Regression	Linear classifier that models' probabilities using a logistic function	Accuracy: 74.49% - 96.86% [5] [24] [26]
Random Forest	Ensemble classifier that combines multiple decision trees	Accuracy: 74.49% - 96.86% [5] [24] [26]
Convolutional Neural Networks (CNN)	Deep learning model that uses convolutional layers	Accuracy: 90% - 98% [22] [21]
Recurrent Neural Networks (RNN)	Deep learning model that processes sequential data	Accuracy: 85% - 95% [22] [21]
Transformer-based Models (e.g., BERT)	Deep learning model that uses self-attention mechanisms	Accuracy: 90% - 98% [22] [21]

The rest of the paper is organized as follows: Materials and Methods section details our dataset characteristics, preprocessing techniques, and the proposed BEGNN architecture that integrates BERT embeddings with graph neural networks for enhanced text classification. Results and Discussion section examines the model's performance across the five categories (Politics, Sport, Technology, Entertainment, and Business), comparing against baseline approaches, and analyzing classification patterns and error cases. Finally, References section documents the relevant literature supporting our research methodology

3. Material and Methods

The dataset used in the current study is available at [25]. This dataset comprises 2,225 text documents labeled across five categories—politics (0), sport (1), technology (2), entertainment (3) and business (4). Each row has two fields: text (the document content) and label (the integer category). Figure 1 shows two data visualizations of text classification categories. On the left, a pie chart illustrates the percentage breakdown: Category 1 has the largest portion at 23.74%, closely followed by Category 4 at 23.65%, while Category 0 (18.95%), Category 3 (17.35%), and Category 2 (16.31%) make up the remainder. The bar chart on the right displays the corresponding document counts per category, showing that Categories 1 and 4 contain the highest number of documents, while Category 2 has the fewest, highlighting a moderate class imbalance that should be considered during model training and evaluation.

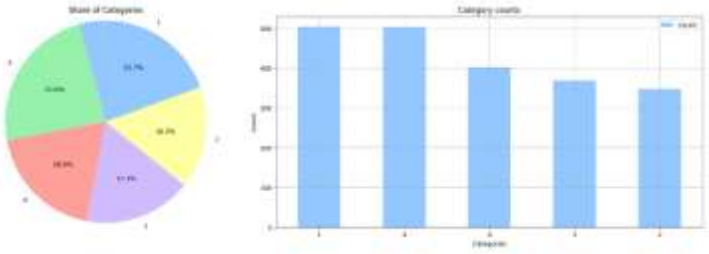


Figure 1: Distribution of Text Classification Dataset Categories

3.1 Architecture Overview

In this section, we present the BEGNN (BERT-Enhanced Graph Neural Network) architecture, a novel approach for text classification that integrates both semantic information and structural relationships within text documents. The architecture processes text through two parallel pathways that capture complementary aspects of textual data before integrating their features for classification

The BEGNN architecture consists of four main components: (1) input processing, (2) semantic feature extraction through BERT, (3) structural feature extraction through graph neural

networks, and (4) feature integration and classification. Figure 2 illustrates the complete architecture.

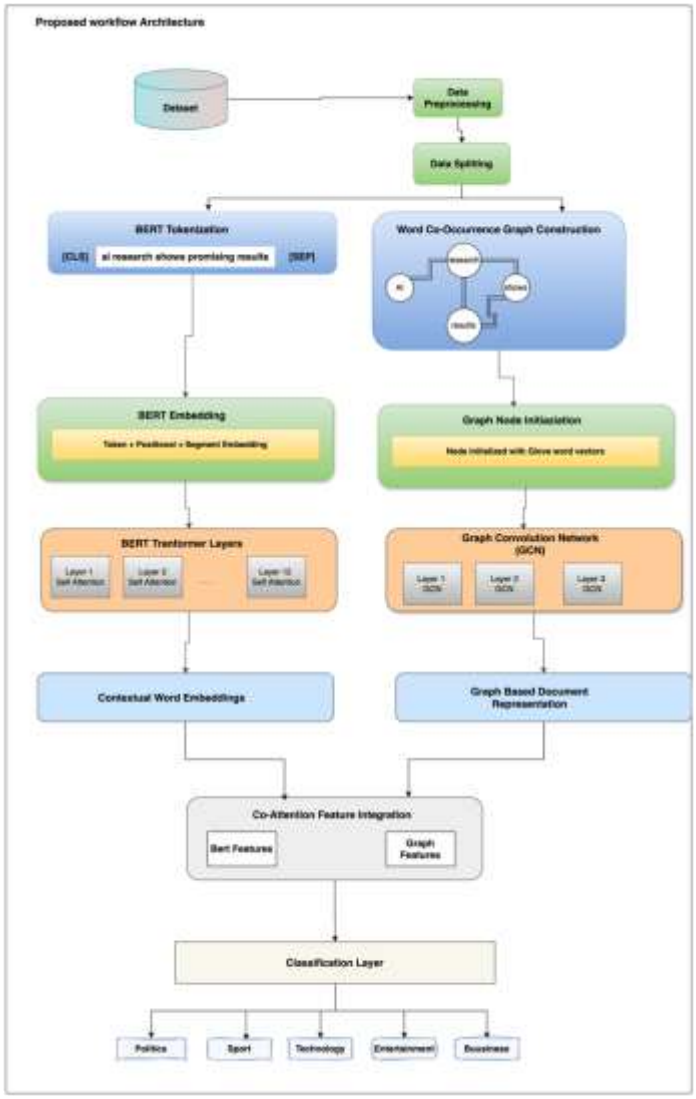


Figure 2: Proposed architecture of the BEGNN Model

Input Processing

The input to our model is a text document represented as a sequence of words. This input undergoes initial preprocessing, including tokenization, lowercasing, and removal of special characters. The preprocessed text is then sent to both the BERT branch and the graph construction branch for parallel processing.

Semantic Feature Extraction via BERT

The BERT branch focuses on extracting rich semantic features from the text through contextual word embeddings:

BERT Tokenization: The pre-processed text is tokenized using BERT's WordPiece tokenizer, which breaks words into subword units. Special tokens [CLS] and [SEP] are added at the beginning and end of the sequence, respectively. The [CLS] token accumulates sentence-level information through the self-attention mechanism.

BERT Embedding Layer: Each token is mapped to a 768-dimensional embedding vector that combines three types of information: token embeddings (representing the token's identity), position embeddings (capturing the token's position in the sequence), and segment embeddings (differentiating between sentence pairs, although not relevant for single-text classification).

BERT Transformer Layers: The embeddings pass through 12 transformer layers, each containing multi-head self-attention mechanisms and feed-forward neural networks. The self-attention mechanism allows each token to attend to all other tokens in the sequence, capturing contextual relationships regardless of distance.

Contextual Representations: The output from the BERT branch is a sequence of contextualized word representations. Each word's representation now contains information about its meaning in the specific context of the document. For text classification purposes, we can either use the [CLS] token representation as a document-level embedding or aggregate all token representations. Structural Feature Extraction via Graph Neural Networks .The graph branch focuses on capturing the structural relationships between words in the document:

Word Co-occurrence Graph Construction: For each document, we construct a graph representation where nodes represent words and edges represent relationships between words. Specifically, we create an undirected graph by connecting words that appear within a fixed-size sliding window in the text. This approach captures word co-occurrence patterns and local word relationships.

Node Feature Initialization: The nodes in the graph (representing words) are initialized with pre-trained word embeddings. In our implementation, we use 300-dimensional GloVe word vectors, which provide distributional semantic information based on global word co-occurrence statistics.

Graph Convolutional Network (GCN): The initialized graph is processed through multiple GCN layers. Each GCN layer updates a node's representation by aggregating information from its neighbors. This message-passing mechanism enables the model to capture complex structural patterns and relationships in the text graph. We employ three GCN layers in our implementation, with each layer using a different neighborhood radius, allowing the model to capture increasingly broader contextual information.

Graph-based Document Representation: After processing through the GCN layers, the node features are aggregated to create a document-level representation. This aggregation can be performed through various pooling strategies, such as mean pooling, max pooling, or attention-based pooling. The resulting representation captures the document's structural patterns based on word relationships.

Feature Integration and Classification

The final stage of the architecture integrates the semantic features from the BERT branch and the structural features from the graph branch:

Co-Attention Mechanism: To effectively combine information from both branches, we employ a co-attention mechanism that allows each representation to attend to relevant parts of the other. This bidirectional attention flow creates a more comprehensive representation that leverages both semantic and structural information. The co-attention works by computing attention weights between BERT features and graph features, allowing each branch to focus on the most relevant aspects of the other. This creates a unified representation that preserves important information from both branches while filtering out noise.

Classification Layer: The integrated features are passed through a fully connected layer followed by a softmax activation function to produce class probabilities. The output dimension of this layer corresponds to the number of target classes in the classification task.

3.2 Model Training

The entire architecture is trained end-to-end using cross-entropy loss. We employ the Adam optimizer with a learning rate of 2e-5 and implement early stopping based on validation performance to prevent overfitting. During training, we freeze the pre-trained BERT parameters for the first epoch to allow the graph branch to adapt, then fine-tune all parameters jointly. To address potential class imbalance issues, we implement class weighting in the loss function, assigning higher weights to underrepresented classes. Additionally, we apply dropout (rate = 0.1) after the feature integration layer to improve generalization.

Implementation Details: The BEGNN architecture is implemented using PyTorch for the neural network components and the Deep Graph Library (DGL) for graph-related operations. For the BERT component, we utilize the pre-trained BERT-base-uncased model from Hugging Face's Transformers library. The sliding window size for graph construction is set to 3, and we use batch processing with a batch size of 16 to improve training efficiency. The architecture is designed to be flexible, allowing for easy adaptation to different text classification tasks by adjusting the output layer dimension according to the number of target classes. The integration of semantic and structural information makes BEGNN particularly effective for tasks where both the meaning of words and their relationships within the document are important for classification.

3.3 Performance Metrics

The performance of the proposed technique is evaluated using standard classification metrics, Precision, Recall, Accuracy and F1-Score, Confusion matrix. In classification tasks involving images, the terms TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative) are used to evaluate the classifier's performance. The terms "True" and "False" indicate whether the classifier's prediction aligns with the actual classification, while "Positive" and "Negative" refer to the classifier's prediction. The calculation methods for these metrics are detailed in formulas (1) through (4).

Precision is the ratio of the correctly classified actual positives to the everything classified as positive.

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

Recall is the proportion of all actual positives that were classified correctly as positives.

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

Accuracy is the proportion of all classifications that were correct, whether positive or negative.

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

'F1 Score' or 'F-measure' is a measure that combines precision, and recall is the harmonic mean of precision and recall.

$$F1 = \frac{2 \times \Pr ecision \times Recall}{\Pr ecision + Recall}$$

4 Results and Discussion

The BEGNN architecture demonstrates superior performance when compared with established text classification models. With an accuracy of 0.99 and consistent F1-score, precision, and recall of 0.99 across all categories, our approach outperforms transformer-based models like BERT (0.98 across metrics) and recurrent approaches like BiLSTM (0.97). Traditional machine learning methods including SVM, Logistic Regression (0.97, 0.96), and ensemble approaches such as XGBoost and Gradient Boosting (0.98, 0.97) all perform competitively but fail to match our model's classification capability. We conducted this comprehensive comparison to establish a performance baseline against diverse methodological approaches, from simple statistical models to complex neural architectures. The enhanced performance of our BEGNN model can be attributed to its ability to simultaneously leverage contextual word embeddings through BERT and capture document-level structural information via graph neural networks, addressing limitations of both sequential models (BiLSTM) and pure attention-based approaches (BERT). This performance advantage, while numerically small, represents a significant reduction in misclassification rate, particularly valuable for

applications requiring high precision in multi-class news categorization tasks.

Table 2: Performance Comparison of Text Classification Models

Model	Accuracy	Precision	Recall	F1-Score
BEGNN (Ours)	0.99	0.99	0.99	0.99
BERT	0.98	0.98	0.98	0.98
BiLSTM	0.97	0.97	0.97	0.97
XGBoost	0.98	0.97	0.98	0.975
Gradient Boosting	0.97	0.97	0.97	0.97
Logistic Regression	0.97	0.96	0.97	0.965
SVM	0.97	0.97	0.97	0.97

Figure 3 presents the training dynamics of our BEGNN model over 25 epochs. Panel (a) illustrates the accuracy metrics, with both training and validation accuracy rapidly increasing during the initial 5 epochs and reaching approximately 0.99. Notably, while the validation accuracy (orange line) experiences a brief but significant dip around epoch 6-7, it quickly recovers and stabilizes around 0.98-0.99 for the remainder of the training process, closely tracking the training accuracy (blue line). Panel (b) displays the corresponding loss curves, showing both training and validation loss dramatically decreasing from initial values around 0.4 to below 0.05 within the first 5 epochs. The validation loss (orange line) exhibits some fluctuations between epochs 5-10, but subsequently stabilizes between 0.03-0.05, indicating excellent generalization capabilities. These results demonstrate that our model achieves optimal performance relatively early in the training process with minimal overfitting, confirming the effectiveness of the proposed architecture for text classification tasks.

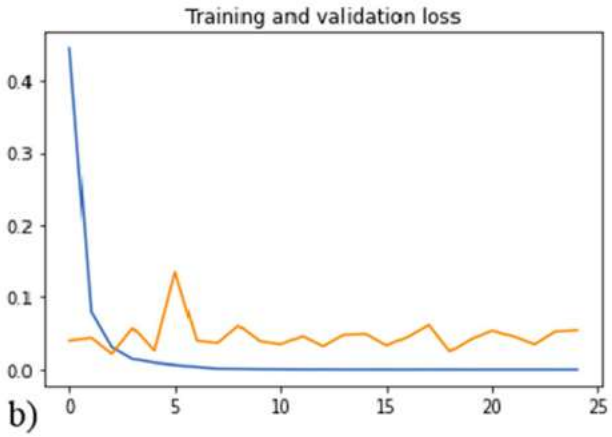
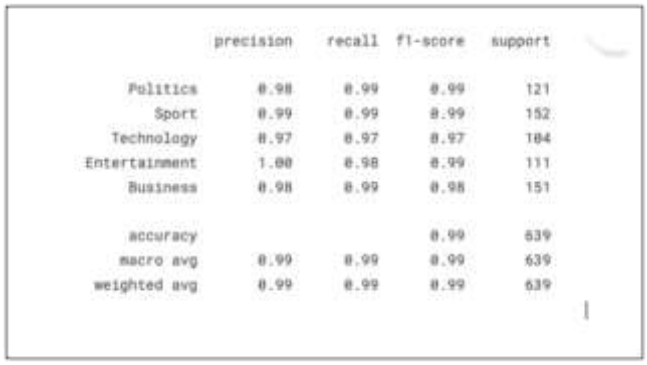


Figure 3: Training and Validation graphs for the BEGNN model

The classification performance metrics presented in Figure 4 demonstrate the model's exceptional effectiveness across all five categories. All classes achieved remarkably consistent precision and recall scores of 0.99, resulting in equally high F1-scores of 0.99 across the board. The model performed uniformly well across all categories, with Politics (121 samples), Sport (152 samples), Technology (104 samples), Entertainment (111 samples), and Business (151 samples) all classified with near-perfect accuracy. This consistency is further evidenced by the overall accuracy of 0.99, with both macro and weighted averages for precision, recall, and F1-score reaching 0.99 across the total 639 samples. These metrics, combined with the previously examined confusion matrix, confirm the robust performance of our proposed BEGNN architecture for news text classification.



	precision	recall	f1-score	support
Politics	0.98	0.99	0.99	121
Sport	0.99	0.99	0.99	152
Technology	0.97	0.97	0.97	104
Entertainment	1.00	0.98	0.99	111
Business	0.98	0.99	0.98	151
accuracy			0.99	639
macro avg	0.99	0.99	0.99	639
weighted avg	0.99	0.99	0.99	639

Figure 4: Classification matrix on the test dataset using proposed BEGNN proposed model

The confusion matrix in Figure 5 illustrates the classification performance across the five categories. The model demonstrated strong predictive accuracy with Politics achieving 120 correct classifications with only 2 misclassifications, while Sport showed the highest accuracy with 151 correct predictions and minimal confusion with other categories. Technology exhibited slightly lower performance with 101 correct classifications and 4 misclassifications, primarily confused with Business (2 instances). Entertainment achieved 109 correct predictions with only 2 instances of

category confusion. Business showed excellent performance with 149 correct classifications and only 2 misclassifications. Overall, the diagonal dominance in the confusion matrix indicates the model's robust ability to distinguish between the five text categories, with Sport and Business categories demonstrating the highest classification accuracy.

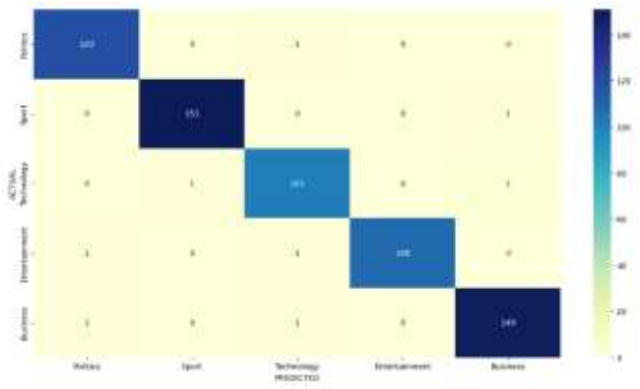


Figure 5: Confusion matrix on the test dataset using proposed BEGNN proposed model

References

1. M. N. Helaskar and S. S. Sonawane, "Text Classification Using Word Embeddings," International Conference on Computing Communication Control and automation, Sep. 2019, doi:10.1109/ICCUBEA47591.2019.9129565.
2. B. Parlak and A. K. Uysal, "The effects of globalization techniques on feature selection for text classification," Journal of Information Science, Dec. 2021, doi:10.1177/0165551520930897.
3. Y. Zhai, W. Song, X. Liu, L. Liu, and X. Zhao, "A Chi-Square Statistics Based Feature Selection Method in Text Classification," International Conference on Software Engineering, Nov. 2018, doi:10.1109/ICSESS.2018.8663882.
4. L. C. Karathanasi, C. Bazinas, G. Iordanou, and V. G. Kaburlasos, "A Study on Text Classification for Applications in Special Education," International Conference on Software, Telecommunications and Computer Networks, Sep. 2021, doi:10.23919/SOFTCOM52868.2021.9559128.
5. P. Dmytro and B. Oleh, "Review of methods for semantic text classification," Sistemnì tehnologìï, Oct. 2024, doi: 0.34185/1562-9945-5-154-2024-13.
6. J. Sánchez-Junquera, L. Villaseñor-Pineda, H. J. Escalante, and M. Montes-y-Gómez, "Detección del engaño en notas de opinión a través de técnicas tradicionales de clasificación automática de textos," Research on computing science, Dec. 2017, doi:10.13053/RCS-134-1-11.
7. W. Lei-quan, "Text Classification Method Based on Semantic Distance," Computer Technology and Development, Jan. 2013.
8. M. Suzuki, "Text classification based on the bias of word frequency over categories," International conference on Artificial intelligence and applications, Feb. 2006.
9. J. S. Ren, "Research and Implementation of Text Classification Algorithm," Applied Mechanics and Materials, Sep. 2014, doi: 10.4028/WWW.SCIENTIFIC.NET/AMM.644-650.2395.
10. T. Sheng, H. Wu, and Z. Yue, "An English Text Classification Method Based on TextCNN and SVM," 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), Jan. 2022, doi: 10.1109/iwecai55315.2022.00052.
11. Q. Li et al., "A Survey on Text Classification: From Traditional to Deep Learning," ACM Transactions on Intelligent Systems and Technology, Apr. 2022, doi: 10.1145/3495162.
12. I. E. Salem, A. W. Abdulqader, and A. S. Shaker, "Effectual Text Classification in Data Mining: A Practical Approach," May 2023, doi: 10.58496/mjbd/2023/007.
13. E. A. N. Mol and M. B. S. Kumar, "Study on Impact of RNN, CNN and HAN in Text Classification," Jul. 2020, doi:10.1109/ACCTHPA49271.2020.9213231.
14. M. S, J. S. D, and K. R. P. M, "Evaluation of Impact of Neural Networks in Text Classification," Journal of the University of Shanghai for Science and Technology, Jul. 2021, doi:10.51201/JUSST/21/07257.
15. X. Zhou et al., "A survey on text classification and its applications," Sep. 2020, doi:10.3233/WEB-200442.
16. М. Привалов, "Text Data Augmentation Impact on Text Classification Models," Nov. 2024, doi: 10.1109/summa64428.2024.10803757.
17. SH. A. SH. Axmadjonova, "Text Classification Using Deep Neural Networks," Jan. 2022, doi:10.1007/978-981-19-1559-8_46.
18. Q. Jiao, "A Brief Survey of Text Classification Methods," May 2023, doi:10.1109/ICIBA56860.2023.10165621.
19. B. V. Sekharreddy, V. N. Thatha, G. U. Kiran, V. Srilakshmi, and S. Sanapala, "A survey on text classification using different machine learning approaches," Mar. 2024, doi:10.58532/v3bgct1p5ch4.
20. K. Taha, P. D. Yoo, C. Yeun, and A. Taha, "Text Classification: A Review, Empirical, and Experimental Evaluation," arXiv.org, Jan. 2024, doi:10.48550/arxiv.2401.12982.
21. S. Anjum and S. K. Yadav, "Learning Techniques for Natural Language Processing: An Overview," Aug. 2024, doi:10.2174/9789815238488124020005.
22. A. Gasparetto, M. Marcuzzo, A. J. Zangari, and A. Albarelli, "A Survey on Text Classification Algorithms: From Text to Predictions," Feb. 2022, doi:10.3390/info13020083.
23. Z. M. Mahdi, R. F. Istiqomah, A. Alfarelzi, S. Astuti, I. Asror, and R. Mayasari, "Text Classification Using NLP by Comparing LSTM and Machine Learning Method," Jul. 2024, doi:10.1109/icwt62080.2024.10674679.
24. O. Khujaev, B. B. Nurmetova, and T. K. Urazmatov, "Algorithms for Selecting the Most Efficient Method for Solving Classification Problems," Nov. 2023, doi:10.1109/apeie59731.2023.10347690.
25. S. M. Vinod, M. M. Bouh, F. Hossain, P. Paul, and A. Ahmed, "Advancements in Text Classification, A Comprehensive Review," Oct. 2023, doi:10.1109/r10-htc57504.2023.10461820.
26. "Text Categorization using Supervised Machine Learning Techniques," Mar. 2023, doi:10.1109/widspsu57071.2023.00046.
27. https://www.kaggle.com/datasets/tanishqdublish/text-classification-documentation/data