International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 11-No.4 (2025) pp. 9415-9421
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Arabic text classification using graphs and deep learning

Mohammed Benhammouda[1]*, Abdelkader Khobzaoui[2], Nadir Mahammed[3]

[1]EEDIS laboratory, University of Djillali Liabes, Sidi Bel Abbes, Algeria
* **Corresponding Author Email:** mohammed.benhammouda@univ-sba.dz - **ORCID:** 0000-0003-2429-8852

[2]Laboratory of Mathematic, University of Djillali Liabes, Sidi Bel Abbes, Algeria
Email: akhobzaoui@yahoo.fr - **ORCID:** 0000-0002-0910-1794

[3]LabRI-SBA Laboratory, Ecole Superieure en Informatique, Sidi Bel Abbes, Algeria
Email: n.mahammed@esi-sba.dz - **ORCID:** 0000-0001-7865-5937

Article Info:

DOI: 10.22399/ijcesen.4402
Received : 11 July 2025
Revised : 11 October 2025
Accepted : 20 October 2025

Keywords

Arabic NLP
Text classification
Graph Neural Networks
AraBERT
Deep learning

Abstract: This paper proposes a novel approach to Arabic text classification that integrates Graph Convolutional Networks (GCNs) with AraBERT embeddings. Unlike traditional sequence-based methods, our framework constructs document-level graphs where words are represented as nodes and edges encode semantic and co-occurrence relations. AraBERT provides rich contextual embeddings for each node, enabling the GCN to capture both local and global dependencies. Experiments on the SANAD–Khaleej dataset (45,500 news articles across seven balanced categories) show that our model achieves 97.25% accuracy, 97.26% macro-F1, and 97.27% recall, significantly outperforming baseline models such as CNNs (95.89% accuracy) and LSTMs (95.23% accuracy). The results confirm the effectiveness of combining graph-based architectures with pre-trained language models for morphologically rich languages such as Arabic, and demonstrates scalability for large-scale text processing.

1. Introduction

The exponential growth of Arabic digital content across social media, news platforms, and academic repositories has created an urgent demand for effective text classification systems. Unlike Latin-based languages, Arabic presents unique challenges such as complex morphology, rich inflection, and significant lexical variation. These characteristics make traditional natural language processing (NLP) techniques less effective for Arabic.

While deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transformer-based architectures have achieved strong results in English text classification, progress in Arabic remains limited. Sequence-based models often fail to capture long-range dependencies and semantic relationships that are essential for handling morphologically rich languages.

Recent advances in Graph Neural Networks (GNNs) offer a promising alternative. By representing text as graphs—where words and documents are nodes connected by semantic or co-occurrence relations—GNNs can naturally capture structural dependencies beyond sequential context [1,2]. Graph Convolutional Networks (GCNs), in particular, have shown effectiveness in text classification tasks by integrating local and global contextual information [3]. However, their potential remains underexplored for Arabic.

In this paper, we introduce GCN+AraBERT a novel Arabic text classification framework, that combines GCNs with AraBERT embeddings [4]. By leveraging pre-trained contextual embeddings, our method enriches the representation of graph nodes and allows the GCN to better capture semantic dependencies in Arabic text.

The main contributions of this work are as follows:

1. A graph-based framework for Arabic text classification that represents documents as word-document graphs enriched with contextual embeddings.
2. Integration of AraBERT embeddings into GCNs, enabling effective modeling of Arabic's morphological and semantic complexity.
3. Extensive evaluation on the SANAD–Khaleej dataset [5], showing significant improvements over CNN, LSTM, and AraBERT-only baselines.

This study demonstrates that combining graph-based architectures with pre-trained language models substantially improves performance on Arabic text classification tasks and provides a scalable approach for morphologically rich languages.

2. Background and Related Work

Text classification has evolved from traditional feature engineering methods to deep learning [6] and, more recently, to graph-based approaches. Early approaches relied on Bag-of-Words and TF-IDF, which represent documents as sparse vectors but fail to capture semantic and structural dependencies [7]. Word embeddings such as Word2Vec [8] and GloVe [9] improved semantic representation, but remain static and insensitive to context [10].Deep learning models, including CNNs [5,11] and LSTMs [12], brought significant improvements by learning contextual features directly from text sequences. More recently, transformer-based architectures such as BERT [13] and AraBERT [4] achieved state-of-the-art results by producing contextualized embeddings. However, these sequence-based models are limited in capturing long-range dependencies and higher-order relationships between words and documents—an issue particularly challenging in morphologically rich languages like Arabic.Graph Neural Networks (GNNs) have emerged as a promising paradigm for text classification [14,15]. By representing words and documents as nodes connected by semantic or co-occurrence relations, GNNs can effectively model structural dependencies beyond linear sequences. Graph Convolutional Networks (GCNs) are particularly suited for this task, as they aggregate information across neighboring nodes and learn rich relational embeddings. Prior work has shown the effectiveness of GCNs for English text classification [2,3], but their application to Arabic remains underexplored.Recent studies on Arabic NLP highlight the potential of combining contextual embeddings with advanced architectures. AraBERT [4] has become a strong baseline for Arabic understanding, but its integration with graph-based models has not been sufficiently investigated. Our work fills this gap by combining GCNs with AraBERT embeddings for Arabic text classification, aiming to capture both contextual semantics and structural dependencies.

3. Methodology

Our methodology integrates graph-based modeling with pre-trained contextual embeddings to improve Arabic text classification. The framework consists of four main components: graph construction, embeddings, GCN architecture, and dataset preparation.

3.1 Graph Construction

Documents are represented as heterogeneous graphs. Each unique word and each document are modeled as a node. Edges capture relations such as:

- Word–document edges, linking words to the documents in which they appear.
- Word–word edges, based on co-occurrence within a sliding window or semantic similarity.

3.2 Embeddings (AraBERT)

We leverage AraBERT, a transformer-based model pretrained on large Arabic corpora, to initialize node embeddings. AraBERT provides contextual representations that capture morphology and semantics more effectively than static embeddings. Each word node is thus enriched with a vector embedding that reflects its contextual meaning in the document.

3.3 GCN Architecture

The classification model is based on a two-layer Graph Convolutional Network. The first layer aggregates neighborhood information and applies ReLU activation, while the second layer outputs class-specific scores. Dropout is applied to mitigate overfitting, and cross-entropy loss is used for optimization. Hyperparameters include: hidden dimension = 1024, dropout = 0.5, learning rate = 0.001, and batch size = 64. Training is performed using PyTorch Geometric.

3.4 Dataset and Preprocessing

We evaluate our approach on the Khaleej subset of the SANAD corpus [5], a large-scale freely available benchmark designed for Arabic text categorization tasks, composed of news articles collected from three major Arabic news portals: alarabiya.net, alkhaleej.ae, and akhbarona.com. The Khaleej subset was constructed from alkhaleej.ae, consisting of 45,500 news articles across seven balanced categories: Finance, Religion, Sports, Culture, Medical, Technology and Politics. All articles are written in Modern Standard Arabic, ensuring the absence of dialectal variation.

The SANAD–Khaleej dataset was selected for its balanced distribution, linguistic quality, and large scale, making it a suitable benchmark for deep learning and graph-based models.

4. The proposed approach

To create a graph suitable for classification purposes, it is essential to accurately define the nodes and edges. In this context, the words in a document are represented as nodes in the graph. Each unique word becomes a distinct node, which allows the model to effectively capture the relationships among different terms. This representation can be further improved by incorporating word embeddings from pre-trained models like BERT or AraBERT, which provide contextualized vectors for each word node. The edges in the graph represent the relationships between these word nodes and also include document nodes to link words with their respective documents. This approach accommodates both local connections—where words closely interact within a sentence—and global connections—where words relate across multiple documents or wider contexts. By utilizing both word-node and document-node frameworks within a heterogeneous graph structure, Graph Convolutional Networks (GCNs) can effectively leverage these relationships during training. For instance, edges might be established based on co-occurrence statistics, where higher co-occurrence frequencies signify stronger relationships. Once this graphical representation is established, it is integrated into a GCN architecture that learns from these interconnected nodes and edges to identify patterns relevant to text classification tasks. The inclusion of various types of relationships enables GCNs to better understand the complex dependencies present in textual data compared to traditional methods that treat all interactions uniformly. See references: [15,1,14,16,3,2].

4.1 Extracting Word Embeddings with Arabert

To derive effective word embeddings for text classification, we utilize AraBERT [4], a BERT [13] variant designed for Arabic. This approach leverages transfer learning with pre-trained models fine-tuned on extensive Arabic literature. AraBERT generates contextual embeddings that capture the semantic relationships between words based on their context. The embedding process involves tokenizing input text and feeding these tokens into the model, resulting in rich vector representations where each word's meaning is context-dependent. This is particularly beneficial for Arabic, which has complex morphological features and ambiguity. By using AraBERT, we overcome limitations of traditional methods like Word2Vec [8] and GloVe [9], which produce static embeddings lacking contextual sensitivity. After extracting word embeddings from AraBERT, we create a graph representation with nodes representing words or documents. Connections between nodes denote co-occurrences that illustrate linguistic relationships, which can be effectively processed using Graph Convolutional Networks (GCNs). This integration enhances performance in text classification by allowing models to leverage insights from both local and broader contexts. Additionally, AraBERT enriches our feature set and provide a nuanced modeling of the unique linguistic characteristics of Arabic texts.

Prior to use, the dataset was pre-processed to remove Latin characters, punctuation marks, diacritics, elongation characters, and extraneous spaces,

4.2 Model evaluation metrics

The performance of the text classification model is evaluated using four standard metrics: accuracy, precision, recall, and F1-score [17].

- Accuracy is defined as the ratio of correctly predicted instances to the total number of instances, and is computed using Equation (4).
- Precision measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It is calculated using Equation (5).
- Recall represents the proportion of correctly predicted positive instances out of all actual positive instances, as defined by Equation (6).
- F1-score is the harmonic mean of precision and recall, and is especially useful when a balance between the two is required. It is calculated using Equation (7).

$$TF(i,j) = \frac{Frequency\ of\ term\ i\ in\ document\ j}{Total\ number\ of\ terms\ in\ document\ j} \quad (1)$$

$$IDF(i,j) = log\ \frac{Total\ number\ of\ docs\ in\ the\ dataset}{Number\ of\ docs\ which\ include\ term\ i} \quad (2)$$

$$W(i,j) = TF(i,j)\ \times\ IDF(i,j) \quad (3)$$

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP\ + FN)} \quad (4)$$

$$Precision\ = \frac{TP}{TP + FP} \quad (5)$$

$$Recall\ = \frac{TP}{TP + FN} \quad (6)$$

$$F1 - score\ = \frac{2 * Precision * Recall}{Precision + Recall} \quad (7)$$

4.3 Definitions

- TP (True Positive): Number of positive samples correctly predicted as positive. The model successfully identified them as positive.
- TN (True Negative): Number of negative samples correctly predicted as negative. The model successfully identified them as negative.
- FP (False Positive): Number of negative samples incorrectly predicted as positive. The model classified them as positive, although they are actually negative.
- FN (False Negative): Number of positive samples incorrectly predicted as negative. The model classified them as negative, although they are actually positive.

Following the preprocessing of the Arabic document datasets, the data was divided into a training set (80%) and a test set (20%) using Python's train_test_split library function.

4.4 Data Collection and Preprocessing

The stages of data gathering and preprocessing are essential for preparing text data for classification tasks with Graph Convolutional Networks (GCNs). During preprocessing, we enhance data quality by removing stop words, punctuation, and numerical figures to reduce noise. Techniques like stemming or lemmatization standardize word forms for consistent textual representation. Next, tokenization segments documents into smaller units called tokens, which serve as nodes in the graph, while extraneous elements such as URLs or emojis are discarded to focus on critical content. To further refine the dataset for GCN applications, we exclude infrequent words that may not significantly impact classification. The outcome is a cleaned and processed dataset conducive to efficient graph construction, allowing relationships between tokens to be represented as edges in the graph. When building text graphs, we ensure that the semantic relationships and contextual meanings of each token are maintained through appropriate edge weighting techniques based on affinities among the tokens.

4.5 GCN+AraBERT implementation details

To effectively deploy Graph Convolutional Networks (GCNs) for classification, a methodical strategy is essential for model construction and training. The process begins with creating a text graph from the document corpus, where words and documents act as nodes, and edges represent relationships based on co-occurrence or semantic similarity. A distinct word list from the dataset must be compiled to map words to identifiers, forming adjacency matrices that outline node connections. Once the graph is established, word embeddings are crucial for representing textual data accurately. Utilizing pre-trained embeddings like BERT or Arabert enhances GCN performance by providing contextual representations that capture semantic nuances. Optimal model training requires careful tuning of parameters such as learning rate, batch size, and embedding dimensions, alongside incorporating dropout layers to prevent overfitting through regularization. PyTorch is a flexible framework for building GCNs, especially with libraries like PyTorch Geometric, which offers various GCN architectures and advanced techniques like multi-head attention and residual connections. Experimenting with different graph convolutional layers can yield insights into information propagation across the network. Finally, defining relevant performance metrics—such as accuracy, precision, recall, and F1-score—is critical for evaluating the model's classification effectiveness.

5. Experimental results

5.1. Experimental Setup

The proposed model was implemented in PyTorch Geometric. Training was performed on an NVIDIA RTX 3060 GPU with 12GB memory. Hyperparameters were set as follows: learning rate = 0.001, batch size = 64, hidden dimension = 1024, dropout = 0.5, and number of GCN layers = 2. The dataset was split into 80% for training and 20% for testing, ensuring class balance.

5.2 Baseline Models

To validate the effectiveness of our approach, we compared it against several widely used baselines [12]:

- BIGRU
- BILSTM
- CGRU
- CLSTM
- CNN
- GRU
- HANGRU
- HANLSTM
- LSTM

These baselines were selected to represent traditional machine learning, deep sequence models, and transformer-based approaches.

5.3 Overall Performance

Table 1 reports the performance across all models on the SANAD–Khaleej dataset.

Table 1. *Performance comparison on SANAD–Khaleej dataset*

Model	**Accuracy (%)**
BIGRU	95.00%
BILSTM	93.91%
CGRU	94.23%
CLSTM	94.57%
CNN	95.89%
GRU	93.86%
HANGRU	96.94%
HANLSTM	95.49%
LSTM	95.23%
GCN+AraBERT (ours)	97.25%

Our model consistently outperforms all baselines, achieving the highest accuracy. The improvement over GCN+AraBERT (+0.31% accuracy) confirms the added value of graph-based representation.

5.4 Confusion Matrix Analysis

Table 2 shows the confusion matrix for the best-performing model. Most misclassifications occur between Culture and Religion, which share overlapping vocabulary. Other categories are classified with near-perfect accuracy.

Table 2. *Confusion matrix of GCN+Arabert classification model of SANAD–Khaleej dataset*

		Predicted						
	Class	0	1	2	3	4	5	6
Actual	0	594	0	0	0	0	5	3
	1	0	603	2	24	3	1	3
	2	0	2	625	0	1	0	7
	3	2	16	0	631	0	4	3
	4	2	4	1	0	625	4	2
	5	9	1	0	0	9	658	5
	6	1	1	3	2	3	1	646

5.5 Training Dynamics

Fig. 1 and Fig. 2 display training and validation loss and accuracy. The curves show stable convergence without significant overfitting, demonstrating the effectiveness of dropout regularization and balanced dataset construction.

6. Discussion

The results validate our hypothesis that graph-based architectures enhance Arabic text classification.

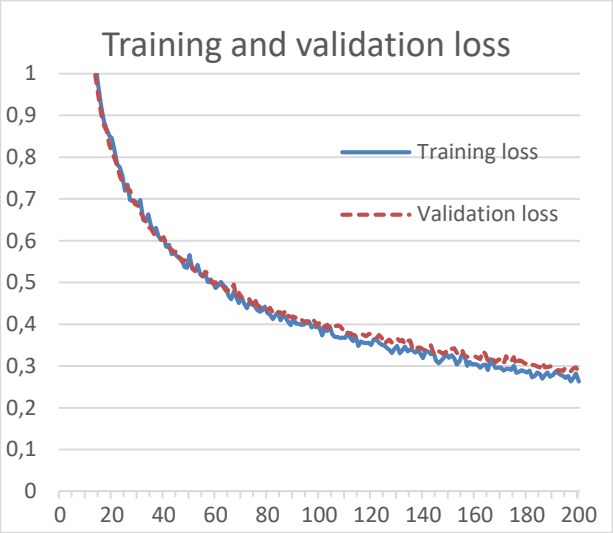

Figure 1. *Training vs validation loss of GCN+AraBERT classification model of SANAD–Khaleej dataset*

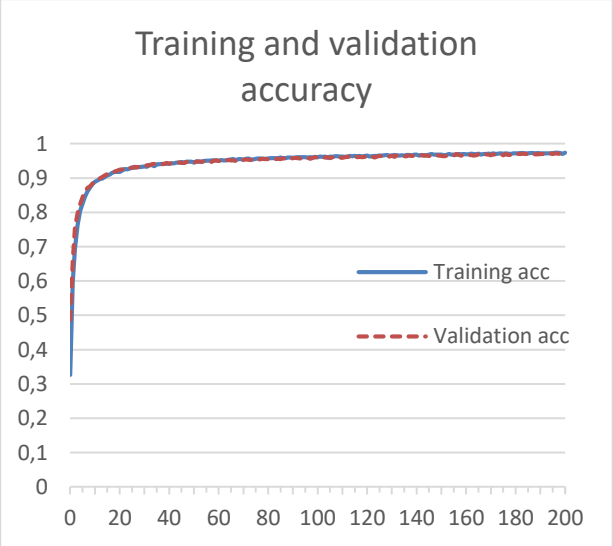

Figure 2. *Training vs validation accuracy of GCN+AraBERT classification model of SANAD–Khaleej dataset*

Table 3. *GCN+Arabert results of SANAD–Khaleej dataset*

Class	Name	Precision	Recall	F1-score
0	Finance	0.9770	0.9867	0.9818
1	Religion	0.9617	0.9481	0.9549
2	Sports	0.9905	0.9843	0.9874
3	Culture	0.9604	0.9619	0.9612
4	Medical	0.9750	0.9796	0.9773
5	Tech	0.9777	0.9648	0.9712
6	Politics	0.9656	0.9833	0.9744

The proposed method captures long-range dependencies and semantic relationships that sequence models cannot.

The experimental results confirm the effectiveness of the proposed model that combines Graph

Convolutional Networks (GCNs) with AraBERT embeddings for Arabic text classification. This hybrid architecture outperforms traditional sequence-based models (CNNs, LSTMs, GRUs) by capturing both local and global dependencies within documents.

Representing texts as graphs provides a significant advantage: it enables the modeling of semantic relations between words beyond their linear order in the text. By integrating contextual embeddings from AraBERT, each word node is enriched with a representation that reflects its morphological and semantic context, which is particularly important for a morphologically rich language like Arabic. Consequently, the GCN can leverage complex relational structures that sequence models cannot effectively capture.

The confusion matrix analysis indicates that most categories are accurately classified, with minor confusion between Culture and Religion, which share overlapping vocabulary and thematic proximity. This suggests that the model successfully captures lexical distinctions and contextual nuances among classes while remaining sensitive to inherent semantic similarities between topics.

Furthermore, the stable convergence of training and validation curves demonstrates the robustness and generalization capability of the model, aided by dropout regularization and a balanced dataset. These results indicate that the proposed framework can be generalized to other Arabic NLP tasks such as sentiment analysis, topic detection, and information retrieval.

Overall, the integration of heterogeneous graph structures with contextual embeddings shows high potential for processing morphologically rich languages. This study highlights the relevance of graph-based architectures for modeling semantic relationships in Arabic text and paves the way for future research incorporating syntactic and discourse-level features into graph representations.

7. Conclusion

This work introduced a graph-based framework for Arabic text classification, GCN+AraBERT, by combining Graph Convolutional Networks with AraBERT embeddings. The proposed method represents documents as graphs, where AraBERT provides contextualized embeddings and GCNs capture semantic and structural dependencies.

Experiments on the SANAD–Khaleej dataset demonstrated that our model outperforms conventional baselines, achieving 97.25% accuracy, 97.26% macro-F1, and 97.27% recall, compared to 95.89% for CNNs and 95.23% for LSTMs. These results highlight the effectiveness of graph-based architectures in handling the morphological richness and long-range dependencies of Arabic.

Beyond accuracy, the framework improves interpretability through graph visualization and scales efficiently to large datasets. This makes it a strong candidate for real-world applications such as news classification, sentiment analysis, and information retrieval in Arabic.

Future work will focus on extending the approach to dialectal Arabic, integrating syntactic and discourse features into the graph, and exploring domain adaptation for multilingual text classification.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1]Kipf, T. N., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks (No. arXiv:1609.02907). arXiv. https://doi.org/10.48550/arXiv.1609.02907

[2]Yao, L., Mao, C., & Luo, Y. (2019). Graph Convolutional Networks for Text Classification. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 7370–7377. https://doi.org/10.1609/aaai.v33i01.33017370

[3]Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph Attention Networks (No. arXiv:1710.10903). arXiv. https://doi.org/10.48550/arXiv.1710.10903

[4]Antoun, W., Baly, F., & Hajj, H. (2021). AraBERT: Transformer-based Model for Arabic Language Understanding (No. arXiv:2003.00104). arXiv. https://doi.org/10.48550/arXiv.2003.00104

[5]Elnagar, A., Al-Debsi, R., & Einea, O. (2020). Arabic text classification using deep learning models. Information Processing & Management, 57(1), 102121. https://doi.org/10.1016/j.ipm.2019.102121

[6]Sundus, K., Al-Haj, F., & Hammo, B. (2019). A Deep Learning Approach for Arabic Text Classification. 2019 2nd International Conference on New Trends in Computing Sciences (ICTCS), 1–7. https://doi.org/10.1109/ICTCS.2019.8923083

[7]El Rifai, H., Al Qadi, L., & Elnagar, A. (2022). Arabic text classification: The need for multi-labeling systems. Neural Computing and Applications, 34(2), 1135–1159. https://doi.org/10.1007/s00521-021-06390-z

[8]Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space (No. arXiv:1301.3781). arXiv. https://doi.org/10.48550/arXiv.1301.3781

[9]Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. In A. Moschitti, B. Pang, & W. Daelemans (Eds.), Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1532–1543). Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1162

[10]Sabri, T., Beggar, O. E., & Kissi, M. (2022). Comparative study of Arabic text classification using feature vectorization methods. Procedia Computer Science, 198, 269–275. https://doi.org/10.1016/j.procs.2021.12.239

[11]Aggarwal, C. C. (2023). Neural Networks and Deep Learning: A Textbook. Springer International Publishing. https://doi.org/10.1007/978-3-031-29642-0

[12]Elnagar, A., Omar Einea, & Al-Debsi, R. (2019). Automatic Text Tagging of Arabic News Articles Using Ensemble Deep Learning Models. In M. Abbas & A. A. Freihat (Eds.), Proceedings of the 3rd International Conference on Natural Language and Speech Processing (pp. 59–66). Association for Computational Linguistics. https://aclanthology.org/W19-7409/

[13]Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In J. Burstein, C. Doran, & T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

[14]Labonne, M. (2023). Hands-on graph neural networks using Python. Packt Publishing Birmingham, UK.

[15]Hamilton, W. L. (2020). Graph representation learning. Morgan & Claypool Publishers

[16]Pal, A., Selvakumar, M., & Sankarasubbu, M. (2020). Multi-Label Text Classification using Attention-based Graph Neural Network. Proceedings of the 12th International Conference on Agents and Artificial Intelligence, 494–505. https://doi.org/10.5220/0008940304940505

[17]Powers, D. M. W. (2020). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation (No. arXiv:2010.16061). arXiv. https://doi.org/10.48550/arXiv.2010.16061