BALKAN JOURNAL OF ELECTRICAL & COMPUTER ENGINEERING, Vol. 12, No. 1, March 2024
Research Article

# Document Classification with Contextually Enriched Word Embeddings

Raad Saadi Mahmood, Gokhan Bakal and Ayhan Akbas

***Abstract*—The text classification task has a wide range of application domains for distinct purposes, such as the classification of articles, social media posts, and sentiments. As a natural language processing application, machine learning and deep learning techniques are intensively utilized in solving such challenges. One common approach is employing the discriminative word features comprising Bag-of-Words and n-grams to conduct text classification experiments. The other powerful approach is exploiting neural network-based (specifically deep learning models) through either sentence, word, or character levels. In this study, we proposed a novel approach to classify documents with contextually enriched word embeddings powered by the neighbor words accessible through the trigram word series. In the experiments, a well-known web of science dataset is exploited to demonstrate the novelty of the models. Consequently, we built various models constructed with and without the proposed approach to monitor the models' performances. The experimental models showed that the proposed neighborhood-based word embedding enrichment has decent potential to be used in further studies.**

***Index Terms*—Text classification, Deep Learning, LSTM, Word2Vec, N-grams**

## I. Introduction

THE complexities of human language and the ambiguous nature of word meanings within various contexts pose a significant challenge for machines attempting to learn the precise meanings of words and accurately extract important classifications. However, with the aid of context and textual data, or through human intervention in creating the necessary contexts, models can be trained to process text more effectively, especially through the use of deep learning or neural networks in natural language processing. These techniques have proven to be useful in a range of language classification applications, including spam detection, question classification, and news classification.

To enable accurate predictive development in language and text processing, algorithms have been developed to identify specific words with unique meaning scopes and to establish relationships between words within text sentences. By developing specialized lists of text and relationships between words, models can be trained to more accurately predict classification outcomes and improve their usefulness.

In the field of natural language processing, deep learning techniques such as recurrent neural networks have demonstrated high accuracy in predicting text. These models are able to memorize and recall previous words and information by storing them in hidden layers and maintaining communication between them to recognize relationships. To achieve this, words are converted into vectors and the text is represented by a bag of words. The text length and units are relatively small and closely related within the model.

This article proposes a method for classifying documents, as illustrated in Fig. 1, by following a series of steps.

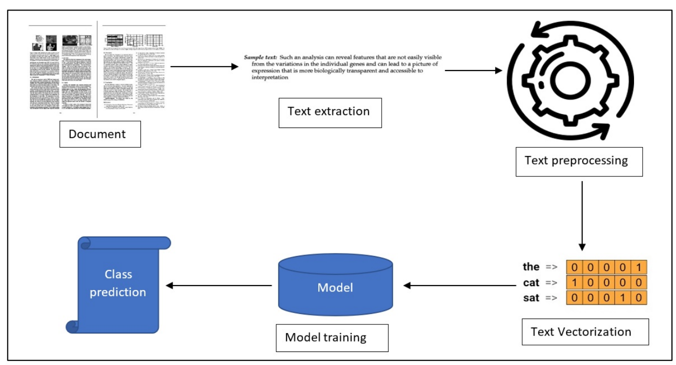


Fig. 1: Document classification model training method

## II. Related work

This article presents a review of previous studies on the use of the single word2vec with n-grams and LSTM models, either independently or in combination, followed by a comparison of their results. Some of these studies include Trappey, Hsu, Trappey and Lin, who utilized a Bayes-based approach with ML and n-grams to classify patent documents with a small dataset of 114 documents achieving a precision rate of approximately 90%, but the model was not suitable for large datasets with 200 words per document. They employed the Back-propagation Network (BPN) algorithm for this purpose [1]. Aghila also employed a small dataset that used feature selection in document classification between single-page documents and 400-page documents. Their approach demonstrated an

**RAAD SAADI MAHMOOD** is with the Department of Computer Engineering, Engineering Faculty, Cankiri Karatekin University, Cankiri, Turkey e-mail: (irqrsm82@gmail.com ) https://orcid.org/0000-0003-0879-1989
**GOKHAN BAKAL** is with the Department of Computer Engineering, Engineering Faculty, Abdullah Gul University, Kayseri, Turkey e-mail: (gokhan.bakal@agu.edu.tr) https://orcid.org/0000-0003-2897-3894
**AYHAN AKBAS** is with the Insttite for Communication Systems, University of Surrey, Guildford, UK e-mail: (a.akbas@surrey.ac.uk) https://orcid.org/0000-0002-6425-104X
Manuscript received Sep 26, 2023; accepted Oct 28, 2024.
DOI: 10.17694/bajece.1366812

accuracy rate of 87% to 98% on the multi-dataset, and they recommended using the combined method for Naïve Bayes and other ML algorithms [2].

Regarding word2vec, Joulin, Grave, Bojanowski, and Mikolov proposed the use of Word2vec in document processing, utilizing images of documents, tags, titles, and captions with 60 and 200 hidden layers. They employed the fastText model with bigram and word vectors, resulting in 97% accuracy with trigram while increasing the model's speed by 600x. However, this study trained with small text documents [3]. Chen and Sokolova detailed embedding algorithms in text processing and analysis of medical and scientific texts, including a dataset of 1000 texts categorized as positive and negative. Their study demonstrated that the Word2vec method is more reliable than Doc2Vec in terms of processing and results [4].

In the realm of natural language processing, several studies have explored the effectiveness of n-grams [5]. Marafino, Davies, Bardach, Dean, and Dudley utilized the SVM algorithm to process n-grams and extract relations. They also combined the n-gram method with feature extraction and achieved accuracy ranging from 0.86 to 0.98, precision of 0.90 to 0.95, and F1 of 0.88 to 0.95. However, this method was tested on a very small dataset of diverse files, including sound and video files [6].

Regarding the LSTM algorithm, numerous studies have investigated its effectiveness. Graves and Schmidhuber found that bidirectional networks outperformed unidirectional ones, and that LSTM was faster and more accurate than standard RNNs and MLPs. In a study by Graves, Fernández, and Schmidhuber, the authors examined the use of long memory in monophonic and biophonic sound classification, and found that bidirectional LSTM outperformed both unidirectional LSTM and traditional RNNs [7], [8].

One of the most significant studies in this field is by Xiao, Wang, and Zuo, who proposed an efficient method for patent document classification. Their approach involves training a patent text classification model using Word2vec and LSTM on a patent dataset, and addressing the dimensional disaster issue caused by traditional methods. They vectorized a dataset of 50,000 using Word2vec, and compared their model's performance with KNN and CNN. Their results showed that LSTM+Word2vec achieved the highest accuracy rate of 93.48%, while normal LSTM achieved 87.7%, CNN+Word2vec achieved 81.18%, CNN achieved 80.59%, and KNN achieved 33.51%. Notably, their approach outperformed other studies that dealt with small amounts of words and datasets in training and testing [9].

## III. METHODS BACKGROUND

This study encompasses the following methods and models:

### A. Natural Language Processing (NLP)

One of the sub-fields of artificial intelligence, stemming from linguistics and computer science, is concerned with methods of interactions and language processing between computers or machines, human language, and methods of communication. The central issue is how computers can analyze and process vast amounts of language data, including the nuances of human language, so that machines can accurately predict and understand the information and ideas contained in documents. Challenges in natural language processing (NLP) include speech recognition, natural language understanding, and natural language generation. NLP combines computational linguistics, which is modeling based on human language rules, with statistical models, machine learning, and deep learning models. NLP content has three categories that are common in text processing: count-based, prediction-based, and sequential [10]. The first category relies on word frequencies with the assumption that common words in a document have fixed meanings, the second models probabilistic relations between words, and the third is based on the assumption that the sequence or stream of words has a fixed meaning for the document. Nowadays, there are many smart applications that are the basis for some computer or smartphone applications, such as digital assistants [11], speech-to-text dictation software [12], customer service chatbots [13], and other consumer amenities that can process large amounts of text quickly, even in real-time. Among its applications:

- Language recognition [14],
- Distinguish words [15],
- Clarify the meaning of the word [16],
- Identify specific entities [17],
- Sentiment analysis [18],
- Natural language generation [19]

### B. Text preprocessing

Preparing models for text classification is a challenging task, as it requires training models to handle textual data in its raw form [20]. Therefore, preprocessing is necessary to reduce the complexity of the data and transform it into a suitable format. This involves extracting texts, words, and data from the text corpus, and transforming them into a variable length vector representation using a dictionary. One of the three common NLP models, namely sequential, prediction-based, or count-based, is then used to process and predict the words. Sequential methods assume that words in a text corpus are linearly related and extract the sequence of words or their stream from the texts. Prediction-based methods, on the other hand, are based on probabilistic relations between words. Finally, count-based methods rely on the frequency of words in a text corpus, assuming that common words have fixed meanings. To prepare the data for our method, we need to carry out the following operations in the training phase:

*1) Extract text:* The initial stage of training models for text classification is word processing, which is of paramount importance. This is followed by the stage of text processing and filtering to exclude all general words, commonly known as "stop words," which add little or no value to the text or context and are widely used with an unspecified meaning that does not contribute to classification. Additionally, the context of the text is examined, which depends on the sequence of words in sentences or sentence structure. As a result, the characteristics derived from this analysis strengthen the accuracy and effectiveness of the classification subject.

*2) Lemmatization:* The second stage of text preprocessing involves converting each word to its base form, while taking into consideration its context, in order to preserve its meaning. This process is followed by tokenization, which further enhances the form of the text.

*3) Tokenization:* In natural language processing, the process of substituting words with unrelated values called "tokens" is used to facilitate processing and internal system of models, without the need to carry the original words in scope. This process involves splitting the words of a text into units, which may involve splitting individual words, letters, or even numbers.

*4) N-gram:* This study focuses on utilizing NLP modeling for the analysis of a sequence of N-words (number of words) in a given text. The text serves as a base for creating word grams through NLP modeling. One-gram, or unigram, refers to a single-word sequence. In the case of the aforementioned statement, each word can be considered a single-gram word. Two-gram, or bi-gram, refers to a sequence of two words. Similarly, three-gram, or tri-gram, pertains to sequences containing three words, as depicted in Fig. 2.

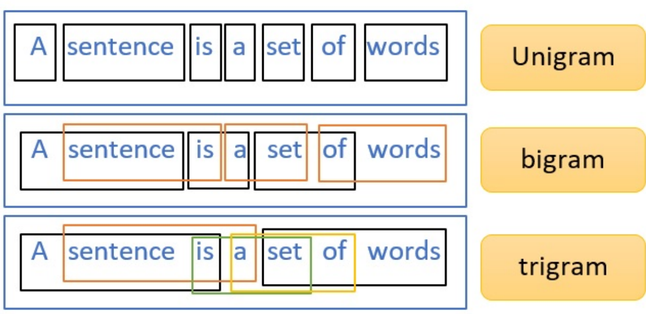


Fig. 2: N-grams

*5) Classification:* In order to prepare a model for text classification, it is crucial to limit the texts that are used to train the model. This involves selecting characteristics that increase the model's experience and reducing errors that may occur during training, such as overlapping topics or repeated words in different fields. To accomplish this, the examination and selection of characteristics using neural convolutional networks have proven effective in filtering and selecting the most relevant features for classification training.

This study focuses on integrating two models, Word2vec and LSTM, to perform the task of classification. This represents an important application of deep learning and natural language processing. The Word2vec model is responsible for processing words and identifying useful words and phrases embedded in the text. The selected phrases and words are considered important for classifying texts, regardless of their convergence to the integrated topic or the specific words used in the document.

On the other hand, the LSTM algorithm predicts the classification based on the proximity of words or the presence of close words (three words or more) and the balance of the words included in the text, such as whether it is an electrical or medical document. Together, these two models can accurately predict the classification of texts based on their unique characteristics and embedded features.

*6) Recurrent neural networks:* RNN, a type of network utilized in the field of deep learning, adopts a conventional approach to handling data and constructing ideas. Upon each data processing event, the network generates new ideas from scratch in a traditional manner. However, the issue with this approach is the failure to maintain previous ideas, which can decay over time. To address this issue, a network cell is introduced to facilitate the exchange of information between cells, and a recurrent neural network can be conceptualized as multiple copies of a network cell, with each copy transmitting a message to its successor. The basic structure of a recurrent neural network resembles that of a chain, and it is used for processing data. This concept has been discussed in a study by Adhikari et al. [21].

*7) Word2vec:* In order to train a classification model, it is necessary to represent words in a numerical format, and these numerical representations must be vectorized so that the model can utilize them based on the relationship between the words. The word2vec algorithm, which is included in the Genism library and utilizes two neural network layers, is used to produce high-quality vectorized representations of words, known as word embeddings, with high efficiency and similarity. Word embeddings are used to represent words in vector form for numerical computation. The word2vec algorithm is designed to target sentences in text in order to generate word embeddings based on the context of the words and their meanings.

Word2vec algorithm consists of two methods: the Skipgram method, which takes a word as input and predicts the surrounding context as output (Fig.3), and the CBOW (Continuous Bag of Words) method, which takes a sentence context as input and predicts the word as output (Fig.4). These methods are then used to generate context for the classification model through word embeddings [4].

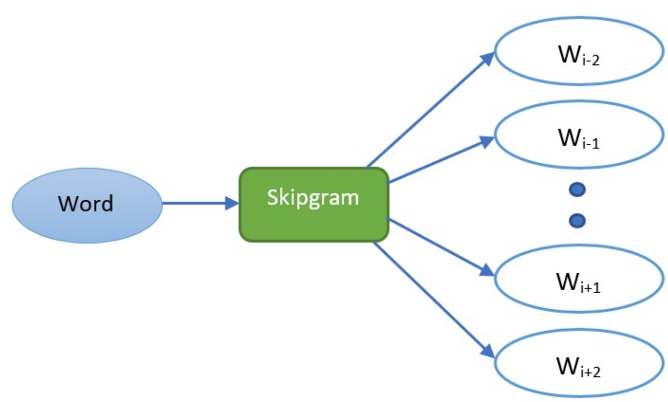


Fig. 3: Skipgram method

*8) LSTM:* LSTM is a widely-known recurrent neural network that is capable of learning long-term dependencies in data. It comprises four layers that interact with each other. The data flows through its units and allows for a few linear

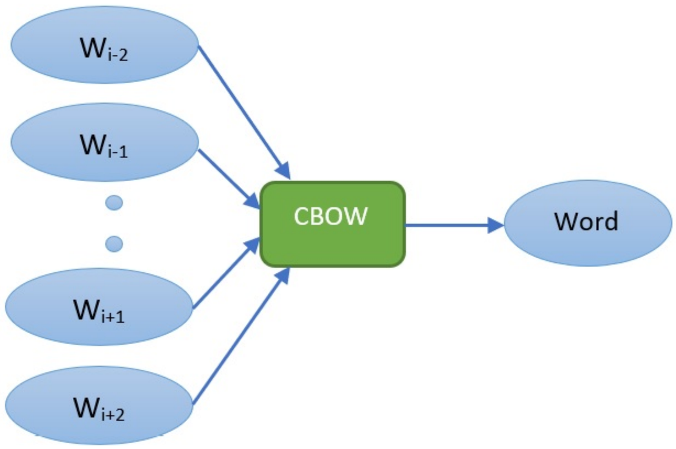


Fig. 4: CBOW method

interactions. By evaluating the combination of corpus, LSTM hyperparameters, and tokenized text, sequential models can accurately classify topics and achieve high performance according to evaluation metrics.

LSTM network has the ability to store previous data and use it during current processing and calculations. This means that the connection between nodes in the hidden layers remains throughout the processing time, which makes the model more effective.

LSTM addresses the issue of vanishing gradients by implementing improvements such as the forget gate, modifying activation functions, and utilizing memory units to enhance connections. Additionally, LSTM maintains the sequence information of text and produces good results on features by considering context, as illustrated in Fig. 5.

This study utilizes word2vec and LSTM with word embeddings to classify documents. The performance of this approach is compared to that of a standalone LSTM model, as well as a combination of LSTM and word2vec, and a combination of LSTM and word embeddings generated after embedding neighboring words.

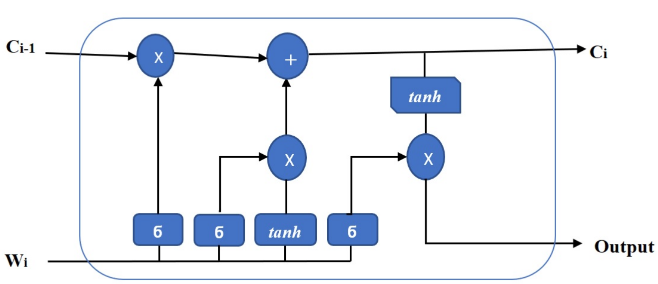


Fig. 5: LSTM model

*9) Word embeddings:* Our study proposes an approach to generating a dense vector representation of words that captures their meaning by employing the word2vec algorithm, which employs two main training algorithms, CBOW and Skipgram, to learn word embeddings. The proposed approach involves integrating LSTM with CBOW and evaluating the performance, followed by integration with Skipgram and performance evaluation. Subsequently, the approach employs the word embeddings generated by both algorithms and evaluates their performance using trigram words, while also generating a neighboring words matrix instead of relying on a dictionary [4].

*10) Dataset:* The Web of Science (WOS) is a dataset for document classification that comprises 46,985 documents belonging to 134 categories, which are further classified under 7 parent categories, as illustrated in Fig. 6.

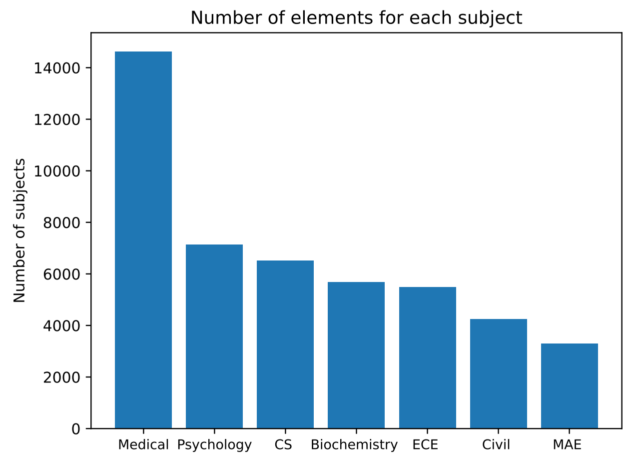


Fig. 6: WOS dataset

## IV. Methodology

The principal motivation of this study is to investigate the effect of contextual information obtained by neighbor words via trigram sequences in textual input for the classification task. First, we generate the trigram word series through the input documents. Then, based on the most discriminative/essential trigram elements, we extract the neighboring words accessed by the shared words/terms in each trigram element. To test our proposed approach, we created two distinct experimental scenarios: **plain models**, where we built regular classification models using general word embeddings, and **neighborhood models**, where we constructed models utilizing neighboring word embeddings. As considered in the proposed novel neighborhood model, we train our word embedding models to gain general word embeddings using *SkipGram* and *Continues-Bag-of-Words* algorithms. Following this step, we extract all the trigram word series and compute the average word embeddings based on the neighboring terms identified through the shared words in the trigrams. Next, we added the calculated neighboring word embeddings to each target word to contextually enhance the embedding of the target word. In the final step, we exploited the updated word embeddings by integrating them into the embedding layer of our LSTM model to perform the classification task.

### A. Preparing data

The dataset contains raw data along with its errors. Prior to training the model, it is necessary to clean the data by examining and validating empty and irregular records. This is followed by applying lemmatization and tokenization to the document texts, and then splitting the data into test and train sets. Subsequently, the first model is trained using the cleaned text data and the results are evaluated for comparison with other models.

### B. Vectorization

Once the data is cleaned and contextual information is retained, it needs to be converted into a vectorized form to enable the classification model to process it effectively. Subsequently, the models are integrated with word embeddings. The second model is integrated with the CBOW algorithm while the third model is integrated with Skipgram, and both are tested and evaluated. The results are presented in Table I.

| | **SKIPGRAM** | **CBOW** | **neighbor WORD** |
|---|---|---|---|
| First Model | ☐ | ☐ | ☐ |
| Second model | ☐ | ☒ | ☐ |
| Third model | ☒ | ☐ | ☐ |
| Fourth model | ☐ | ☒ | ☒ |
| Fifth model | ☒ | ☐ | ☒ |

TABLE I: Models methods (single model and combined models)

### C. Model-vec-word embedding

In this approach, a list of embedded words is prepared to train two separate LSTM models using two methods of Word2vec, Skipgram, and CBOW. This process produces two lists of word embeddings that are used to train two different LSTM models. The performance of the two models is then evaluated to determine which one is better in terms of context and word embeddings. Firstly, the LSTM model is trained with Skipgram vectors, then another LSTM model is trained with CBOW vectors, and finally, the test results are evaluated. This method involves using a model to create vectors that represent the relationships between words and a large number of other words. The Skipgram and CBOW methods both train the model on a large number of words.

### D. Neighbor word embeddings

After incorporating word embeddings through both Skipgram and CBOW methods, the models become more precise in their selection of words and have a broader range of words in their embeddings. However, to improve the accuracy of classification predictions, the models need to have greater accuracy and depth in understanding the relationships between words, thereby narrowing the field between the words. This can be achieved by focusing on the relationships between neighboring words, which can be deduced through the context of similar texts that the models can be trained on. This will narrow the range and number of words, reducing them to the nearest three or more adjacent words. A list of these adjacent words can be compiled and used to train models that minimize the possibilities that may affect the correct classification.

TABLE II: Definition of TP, FP, TN, and FN

| | | **Prediction** | |
|---|---|---|---|
| | | Positive | Negative |
| **Truth** | Positive | True Positive | False Positive |
| | Negative | False Negative | True Negative |

## V. Evaluation

The main concept in classification is to evaluate the models to determine their accuracy and reliability in classification analysis. By reviewing the models' performance, we can identify the most accurate and reliable model for our classification tasks. The results obtained from training and testing the models can be used to assess their performance.

To evaluate the performance of classification models, the confusion matrix is commonly used. This matrix includes four terms, namely true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Based on the confusion matrix, various metrics can be calculated, such as accuracy, precision, recall, F1-score, and others, as shown in Table. II.

The model prediction results are tested and trained, then divided the results in four areas as follows:

- True Positive: The correct model classification of Positive results (Positive).
- False Negative: The incorrect model classification of Positive results (Negative).
- False Positive: The incorrect model classification of Negative results (Positive).
- True Negative: The correct model classification of Negative results (Negative).

These results will measure the metrics as follows:

1) Precision is a ratio calculated between the number of positive samples correctly classified and the total number of samples classified as positive (either correctly or incorrectly). The precision of the model measures how accurately it classifies a sample as positive. Precision is calculated usingEqn 1:

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

2) Recall is a measured ratio that assesses the model's ability to detect positive samples. A higher recall ratio indicates a greater number of positive samples detected. The model's recall is calculated as the ratio between the number of positive samples correctly classified as positive and the total number of positive samples. It is measured using 2:

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

Accuracy is one of the metrics used to measure the overall performance of a model across all classes. It is calculated as the ratio of correct predictions to the total number of predictions, as shown in 3.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

F1-score: It is a measure for evaluating the predictive abilities of a model and assessing its performance by combining two important, competing measures: precision and recall. Currently, this evaluation method is widely used in model assessment, as demonstrated in Eqn. 4.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

The objective of this study is to attain high values in the aforementioned performance metrics for evaluating the models. A high F1-score is obtained when precision and recall are both high, leading to accurate class prediction. To further improve the performance metrics, the models are enhanced using a combined method.

## VI. Results

This experiment involved analyzing texts of 50,000 words per document, as presented in Table I. The study compared the performance of five models trained with Skipgram or CBOW word embeddings. The results showed significant differences in Accuracy, Recall, and F1-score when comparing the performance of the LSTM model as a single model to the combined models. To improve the models, the study used neighboring words for model enhancement and analyzed the predictions.

When evaluating the performance of the LSTM model as a single model, the results showed an accuracy of 0.8126 and an F1-score of 0.8130. When combined with CBOW, the accuracy improved to 0.8724, and the F1-score was 0.8722. However, the best results were obtained by combining LSTM with Skipgram, with an accuracy of 0.8790 and an F1-score of 0.8790, making it the most accurate model in the study. To save time and increase accuracy, the combined models were also evaluated, which showed an accuracy of 0.8721 and 0.8769 for CBOW and Skipgram, respectively, and an F1-score of 0.87.

The models were assessed based on Accuracy, Precision, Recall, and F1-score metrics, and the results are presented in Table III:

| Method | Accuracy | Precision | Recall | F1_score |
|---|---|---|---|---|
| LSTM single model | 0.8126 | 0.8194 | 0.8067 | 0.8130 |
| LSTM with CBOW | 0.8724 | 0.8781 | 0.8664 | 0.8722 |
| LSTM with Skipgram | 0.8790 | 0.8841 | 0.8739 | 0.8790 |
| LSTM with CBOW neighbor word | 0.8721 | 0.8764 | 0.8681 | 0.8722 |
| LSTM with Skipgram neighbor word | 0.87691 | 0.8833 | 0.8708 | 0.8770 |

TABLE III: Comparison of single and combined methods

The single LSTM method curve shows unmatching accuracy between test and train, but combined methods show higher matching, LSTM + Skipgram method shows higher matching in the accuracy curve and higher performance, but the LSTM + Skipgram neighbor word shows the lower loss result between methods, as shown in Fig. 7. The Skipgram algorithm produces pretrained word vectors that capture a wealth of linguistic information from vast text corpora. LSTM's ability to incorporate these pretrained vectors as initial embeddings provides the model with a knowledge base of semantic relationships and word similarities. This, in turn, facilitates faster convergence and better generalization. Since Skipgram's ability to consider surrounding words leads to the generation of contextually rich word embeddings, the LSTM model using Skipgram yields relatively more accurate performance scores. Therefore, the superior performance of LSTM with Skipgram in our study can be attributed to the synergy between LSTM's sequential learning capabilities and Skipgram's context-aware word embeddings. This combination enables the model to grasp text structure, adapt to varying lengths, handle polysemy, and leverage pretrained word vectors, resulting in improved overall performance.

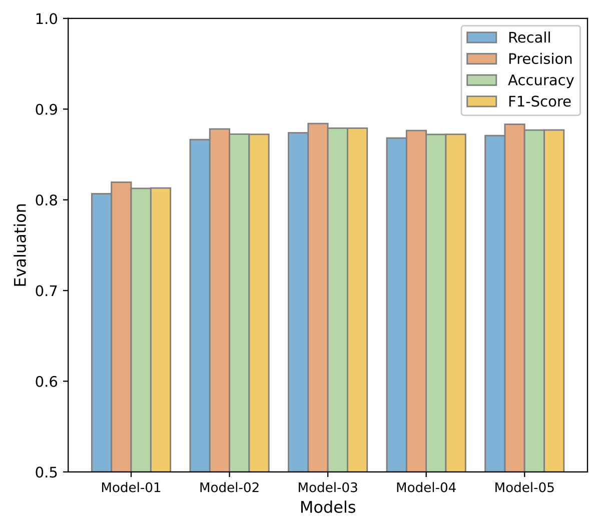


Fig. 7: Models evaluation

## VII. Conclusion

The Documents classification plays an important fundamental role in research and in text classification field. Documents diverse have features that show no single algorithm can perform as accurate in classification fields, however the combined methods serve more accurate with higher performance in classification fields without limiting the studies state-of-the-art depend on single methods or algorithms.

LSTM as one of the NLP classifiers performs better than other NLP classifiers, and in combination with other methods, it has a very high performance. For example, the word2vec method creates appropriate words from context or creates context from words, then gathers these words in a new dataset to train the LSTM model and make the prediction more accurate, as shown in Table III.

The LSTM model approved the best results in Deep learning classification models determining memory, and combined with the word2vec methods, the neighbor word relation may need more development to rise in accuracy and precision.

The single pre-model, as a result of its evaluation, proved that it was trained on text word vectors only and without any relationship between them, which left the words in the memory of this model without any relationship between them as the accuracy decreased in words that shared similar classifications.

The second and third models were trained under a relationship consisting of text vectors formed by methods such as Skipgram and CBOW, where a relationship is formed between words within the memory of the model, and here the model is trained more clearly in the relationship and context between words, where the relationship between words depends on the production of vectors. While Skipgram relies on the word to establish context and yields more accurate evaluation results, as we have observed previously, it's clear that focusing on context is the key to training a model with superior accuracy in classification. In the case of CBOW, as mentioned earlier, the context is utilized to predict the specific word, and this context remains in memory. This can make it easier for the model to make accurate classifications.

It is required to delve deeper into the relationship between words and context. There can exist relationships in texts between words, specifically with adjacent words, which narrows down the range of words within the context or text. This reduced context is used for training the new model, resulting in reduced processing time and improved classification accuracy.

## VIII. Practical Implications & Future Directions

The proposed study's findings have significant practical implications for document classification and natural language processing. Firstly, employing contextually enriched word embeddings, specifically trigram word series, presents a promising approach for enhancing document classification models' effectiveness. This approach leverages contextual information, providing more nuanced and precise text representations.

Beyond this, the demonstrated potential of the proposed approach can lead to opportunities to improve document classification across various domains. Researchers and practitioners can address text classification challenges in subfields like information retrieval, sentiment analysis, and content recommendation with greater precision and effectiveness by utilizing the power of these enriched embeddings. In practical terms, this study presents a valuable contribution to the development of more context-aware document classification models.

From the outcomes, there are two stimulating directions for prospective research as listed below:

- **Dynamic Neighborhood-Based Approaches:** *Investigating dynamic approaches, where the context window dynamically adjusts to the specific requirements of different document types, could further improve the adaptability and accuracy of classification models.*
- **Interplay with Pretrained Models:** *Exploring how contextually enriched word embeddings complement or enhance the performance of pre-trained language models could be valuable. Combining these approaches might result in state-of-the-art document classification systems.*

## References

[1] A. J. Trappey, F.-C. Hsu, C. V. Trappey, and C.-I. Lin, "Development of a patent document classification and search platform using a back-propagation network," *Expert Systems with Applications*, vol. 31, no. 4, pp. 755–765, 2006.

[2] G. Aghila *et al.*, "A survey of na\" ive bayes machine learning approach in text document classification," *arXiv preprint arXiv:1003.1795*, 2010.

[3] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[4] Q. Chen and M. Sokolova, "Specialists, scientists, and sentiments: Word2vec and doc2vec in analysis of scientific and medical texts," *SN Computer Science*, vol. 2, pp. 1–11, 2021.

[5] G. Bakal and O. Abar, "On comparative classification of relevant covid-19 tweets," in *2021 6th International Conference on Computer Science and Engineering (UBMK)*. IEEE, 2021, pp. 287–291.

[6] B. J. Marafino, J. M. Davies, N. S. Bardach, M. L. Dean, and R. A. Dudley, "N-gram support vector machines for scalable procedure and diagnosis classification, with applications to clinical free text data from the intensive care unit," *Journal of the American Medical Informatics Association*, vol. 21, no. 5, pp. 871–875, 2014.

[7] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005: 15th International Conference, Warsaw, Poland, September 11-15, 2005. Proceedings, Part II 15*. Springer, 2005, pp. 799–804.

[8] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[9] L. Xiao, G. Wang, and Y. Zuo, "Research on patent text classification based on word2vec and lstm," in *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1. IEEE, 2018, pp. 71–74.

[10] C. C. Aggarwal and C. C. Aggarwal, "Machine learning for text: An introduction," *Machine learning for text*, pp. 1–16, 2018.

[11] P. Milhorat, S. Schlögl, G. Chollet, J. Boudy, A. Esposito, and G. Pelosi, "Building the next generation of personal digital assistants," in *2014 1st international conference on advanced technologies for signal and image processing (atsip)*. IEEE, 2014, pp. 458–463.

[12] H.-H. Chiang, "A comparison between teacher-led and online text-to-speech dictation for students' vocabulary performance." *English Language Teaching*, vol. 12, no. 3, pp. 77–93, 2019.

[13] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Machine Learning with Applications*, vol. 2, p. 100006, 2020.

[14] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matějka, "Language recognition in ivectors space," in *Twelfth annual conference of the international speech communication association*, 2011.

[15] T. Pedersen and R. Bruce, "Distinguishing word senses in untagged text," *arXiv preprint cmp-lg/9706008*, 1997.

[16] M. Bevilacqua, T. Pasini, A. Raganato, and R. Navigli, "Recent trends in word sense disambiguation: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conference on Artificial Intelligence, Inc, 2021.

[17] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[18] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams engineering journal*, vol. 5, no. 4, pp. 1093–1113, 2014.

[19] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, pp. 57–87, 1997.

[20] B. Kolukisa, B. K. Dedeturk, B. A. Dedeturk, A. Gulsen, and G. Bakal, "A comparative analysis on medical article classification using text mining & machine learning algorithms," in *2021 6th International Conference on Computer Science and Engineering (UBMK)*. IEEE, 2021, pp. 360–365.

[21] A. Adhikari, A. Ram, R. Tang, and J. Lin, "Rethinking complex neural network architectures for document classification," in *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)*, 2019, pp. 4046–4051.

## BIOGRAPHIES

**Raad Saadi Mahmood** earned his BSc in Computer Science from Baghdad University in 2004 and his MSc in Electronics and Computer Engineering from Cankiri Karatekin University in 2023 . With 18 years of experience as a Senior Programmer, he has provided technical support at the Midland Refineries Company, specifically in the Technical Branch of the Measuring Department in Iraq. Mahmood's primary focus is on enhancing industrial systems using AI.

**Gokhan Bakal** received his PhD and MSc degrees in Computer Science from the University of Kentucky, USA in 2019 and 2014, respectively. He is currently working as an Assistant Professor in the Computer Engineering Dept. at Abdullah Gul University. His research areas are Text Mining, Machine and Deep Learning, Bioinformatics, and Computational Drug Discovery.

**Ayhan Akbas** holds BSc and MSc degrees in Electrical and Electronics Engineering, both obtained from the Middle East Technical University in 1991 and 1995, respectively. He earned his PhD degree from the Department of Computer Engineering at TOBB University of Economics and Technology. With over 20 years of experience in the field of Information Technology, Dr. Akbas has occupied both technical and managerial roles at multinational companies. He worked as Assistant Professor at THK University, Cankiri Karatekin University and Abdullah Gül University. Currently, he is pursuing his career at Universiy of Surrey, Guildford, UK. His areas of interest are Wireless Sensor Networks, IoT, 6G, Wireless Communication, and Machine Learning.