Proceedings of CONF-MLA 2024 Workshop: Securing the Future: Empowering Cyber Defense with Machine Learning and Deep Learning
DOI: 10.54254/2755-2721/94/2024MELB0085

Exploring the Impact of Word2Vec Embeddings Across Neural Network Architectures for Sentiment Analysis

Ruoyu Liu

College of Letters and Science, University of California, Los Angeles, California, 90024, United States

cecilialiu@g.ucla.edu

Abstract. Sentiment analysis is crucial for understanding public opinion, gauging customer satisfaction, and making informed business decisions based on the emotional tone of textual data. This study investigates the performance of different Word2Vec-based embedding strategies — static, non-static, and multichannel — for sentiment analysis across various neural network architectures, including Convolution Neural Networks (CNNs), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs). Despite the rise of advanced contextual embedding methods such as Bidirectional Encoder Representations from Transformers (BERT), Word to Vector (Word2Vec) retains its importance due to its simplicity and lower computational demands, making it ideal for use in settings with limited resources. The goal is to evaluate the impact of fine-tuning Word2Vec embeddings on the accuracy of sentiment classification. Using the Internet Movie Database (IMDb), this work finds that multichannel embeddings, which combine static and non-static representations, provide the best performance across most architectures, while static embeddings continue to deliver strong results in specific sequential models. These findings highlight the balance between efficiency and accuracy in traditional word embeddings, particularly when advanced models are not feasible.

Keywords: Sentiment analysis, neural network, word embeddings.

1. Introduction

Artificial intelligence (AI) has expanded rapidly in recent years, permeating nearly every aspect of daily life. It powers everything from virtual assistants to tools that analyze customer feedback or monitor social media trends. The advancement of this technology hinges significantly on its ability to interpret and process human language, underscoring the importance of natural language processing (NLP). A fundamental aspect of NLP is how machines represent words and their meanings [1]. Word embedding, which converts words into dense vector representations, is the foundation of this ability. By capturing semantic and syntactic relationships between words, word embedding allows models to analyze textual data meaningfully [2,3]. Given the central role of word embeddings in NLP tasks, understanding word embeddings is key to grasping how machines process language and improving their performance across diverse applications like sentiment analysis, text classification, and machine translation.

Despite the rapid advancements in NLP, with sophisticated models pushing the boundaries of language understanding, traditional embedding techniques like Word to Vector (Word2Vec) continue to be valuable [4]. While these newer contextual embeddings dynamically adjust word representations

DOI: 10.54254/2755-2721/94/2024MELB0085

based on the context of a sentence, they come with significant computational costs [5,6]. Word2Vec, on the other hand, offers an efficient and low-resource alternative that remains relevant in specific tasks and environments. Its low computational cost and simplicity make it particularly useful in resource-constrained settings, where the overhead of advanced models like Bidirectional Encoder Representations from Transformers (BERT) and Generative Pre-trained Transformer (GPT) may not be justifiable. Additionally, Word2Vec's straightforward interpretability gives it a practical edge in cases where model transparency is essential [5,7].

Recent research has also shown that fine-tuning pre-trained embeddings, particularly in non-static Word2Vec models, can further improve classification performance. Non-static embeddings in Convolution Neural Networks (CNNs) models consistently outperform static embeddings [8].

This paper investigates the performance of different variants of Word2Vec embeddings—static, non-static, and multichannel—when used in conjunction with various neural network architectures such as CNNs, Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRUs). Through this comparison, this work aims to explore how fine-tuning Word2Vec embeddings impacts the performance of sentiment analysis tasks while considering both the efficiency and accuracy of these traditional embeddings in a resource-constrained setting.

2. Methodology

2.1. Dataset

The dataset used in this study is the Internet Movie Database (IMDb) dataset, which was downloaded from Kaggle [9]. The dataset consists of 50,000 movie reviews, equally distributed between positive and negative sentiments, which is suited for binary classification. Rather than using the pre-divided training and test sets provided with the dataset, the entire dataset was randomly split into 80% training and 20% test data.

2.2. Preprocessing

Before training the models, several preprocessing steps were applied to the dataset. First, all reviews were converted to lowercase, and punctuation was removed to ensure consistency. The text was then tokenized using the Natural Language Toolkit (NLTK) library, and common stopwords were removed to eliminate words that do not contribute significant meaning to sentiment classification. Next, a Word2Vec model was trained on the tokenized training data to create word embeddings with a vector size of 100. The reviews were further transformed into sequences of word indices using the Keras Tokenizer, and all sequences were padded to a maximum length of 100 tokens to maintain a uniform input size. Finally, sentiment labels were converted into binary format, where positive reviews were labeled as 1 and negative as 0 to prepare the data for binary classification.

2.3. Embeddings

This study applies the skip-gram model from Word2Vec to generate word embeddings [2]. Mikolov proposes two distinct models for crafting these embeddings: Skip-gram and Continuous Bag of Words (CBOW). The Skip-gram approach focuses on a single word to predict the words around it. For instance, with the word "tennis," Skip-gram may anticipate related words such as "balls" or "racket," effectively capturing complex meanings in language. CBOW works the opposite way, using surrounding words to guess the main word. So, with words like "balls," "racket," and "court," it would predict "tennis."

This work chooses the Skip-gram model due to its proficiency in capturing semantic relationships, especially in smaller datasets, and its effectiveness in handling rare words [10]. Additionally, Skip-gram has consistently shown superior performance to CBOW in sentiment classification tasks because of its capacity to produce more meaningful embeddings for less common words [11].

Building on the chosen methodology, this work configured the Skip-gram model for specific requirements. This work trained the model on tokenized data, selecting a vector size of 100 to represent the dimensionality of the word vectors. Additionally, the author set the window size to 5, allowing the

DOI: 10.54254/2755-2721/94/2024MELB0085

model to consider five words before and after the target word. A minimum word frequency threshold of 5 was imposed to ensure that the vocabulary was limited to words that occur frequently enough to be deemed relevant, allowing the model to concentrate on the most significant words in the corpus.

Once trained, the learned word vectors were stored in an embedding matrix, which was then used to initialize the embedding layers for the neural network models. The embedding matrix contained pre-trained vectors for each word in the vocabulary, with a dimensionality of 100. Inspired by a previous art, this work explored three variants in utilizing these embeddings [8].

Static Embeddings: Word vectors are initialized using pre-trained Word2Vec embeddings, which remain unchanged throughout training.

Non-static Embeddings: Non-static embeddings are also initialized with pre-trained Word2Vec embeddings, but unlike static embeddings, the model is allowed to update these embeddings during training.

Multichannel Embeddings: The multichannel approach combines both static and non-static embeddings by processing them through two separate channels in parallel. The static embeddings remain frozen during training, while the non-static embeddings are fine-tuned based on the task-specific training data. The outputs from the two channels are then concatenated together before being passed to the subsequent layers of the neural network.

2.4. Architectures of neural networks

This study implemented five different neural network architectures—CNN, LSTM, Bidirectional LSTM, GRU, and Bidirectional GRU—to evaluate the performance of the three embedding strategies: static, non-static, and multichannel embeddings. The following sections describe each architecture in detail [12].

2.4.1. CNN. CNNs are widely used in text classification due to their ability to capture local word patterns, such as n-grams, through convolutional filters. CNNs are particularly effective for shorter text sequences like sentiment analysis. This study used filters with kernel sizes of 3, 4, and 5, with each filter applying 250 convolutional filters to the text data, followed by a global max pooling operation to reduce dimensionality.

2.4.2. LSTM. LSTM networks are a type of recurrent neural network designed to capture long-term dependencies in sequential data by using memory cells. This work used an LSTM layer with 100 units to process the input sequences, making it suitable for tasks where the order of words plays a crucial role.

2.4.3. Bidirectional LSTM. Bidirectional LSTMs process input sequences in both forward and backward directions, capturing dependencies from both past and future words in the text. The author used a Bidirectional LSTM with 100 units in each direction.

2.4.4. GRU. GRUs are a more computationally efficient variant of LSTMs, designed to capture long-term dependencies in text sequences with fewer parameters. This study used a GRU layer with 100 units.

2.4.5. Bidirectional GRU. Similar to Bidirectional LSTMs, Bidirectional GRUs capture dependencies from both past and future contexts, with the added advantage of computational efficiency. This work used 100 units in each direction for the Bidirectional GRU.

3. Results and discussion

3.1. Training details

Following the neural network-specific layers, the outputs were passed to a dense layer with 250 units, followed by a sigmoid activation for binary classification. All models were trained using the Adam optimizer with binary cross-entropy loss. Their performance was evaluated using accuracy. Additionally,

DOI: 10.54254/2755-2721/94/2024MELB0085

early stopping was used to prevent overfitting. The early stopping mechanism monitored validation loss and restored the best model weights when no improvement was observed after 20 epochs. This strategy allowed the model to terminate training early if further epochs did not improve performance, ensuring that all models were efficiently trained without unnecessary tuning. No additional steps, such as extensive hyperparameter tuning, were applied beyond early stopping.

3.2. Performance comparison

The results of experiments on the IMDb dataset are presented in the table below. This work evaluated the performance of static, non-static, and multichannel Word2Vec embeddings across five different neural network architectures, as demonstrated in Table 1, including CNN, LSTM, Bidirectional LSTM, GRU, and Bidirectional GRU.

Table 1. Accuracy comparison with the best score marked in bold.

Model	Static	Non-Static	Multichannel
CNN	87.20%	88.75%	**89.88%**
LSTM	**88.88%**	88.45%	88.58%
Bi-LSTM	88.37%	**89.07%**	88.35%
GRU	88.36%	89.01%	**89.07%**
Bi-GRU	88.86%	89.18%	**89.60%**

CNN models with different embedding strategies show a clear trend where multichannel embeddings (0.8988) outperform both non-static (0.8875) and static embeddings (0.8720). This is consistent with Kim's previous findings (2014), where combining static and fine-tuned embeddings allows the model to capture both general and task-specific features, improving accuracy. The significant improvement of multichannel embeddings can be attributed to their ability to capture more complex and varied semantic relationships. In contrast, though still effective, static embeddings may be limited by their fixed nature.

For LSTM models, the static embedding variant (0.8888) slightly outperforms both non-static (0.8845) and multichannel (0.8858). This suggests that the LSTM's ability to capture long-term dependencies in sequences can mitigate some of the limitations of static embeddings. In contrast, Bidirectional LSTM (Bi-LSTM) performed best with non-static embeddings (0.8907), suggesting that fine-tuning embeddings offers an advantage when processing information from both past and future contexts, especially in sentiment analysis, where understanding word nuances in both directions can be crucial.

In the GRU models, multichannel embeddings (0.8907) once again yielded the highest accuracy, followed by non-static (0.8901) and static embeddings (0.8836). The trend is similar for Bi-GRU, where multichannel embeddings (0.8960) outperformed both non-static (0.8918) and static (0.8886) variants. This could be because of GRU's simplified architecture, making it easier for the model to fine-tune task-specific embeddings alongside general-purpose ones without overfitting.

The overall results from the experiments reveal interesting trends in the performance of static, non-static, and multichannel embeddings across the different neural network architectures. While multichannel embeddings performed best for the majority of the results, the relative performance of each embedding type varied depending on the architecture used.

Multichannel embeddings, which combine both static and fine-tuned embeddings, achieved the highest accuracy across most models. For example, the CNN-multichannel (0.8988) and bi-GRU-multichannel (0.8960) models demonstrated the strongest performance. This suggests that leveraging both the general knowledge stored in pre-trained embeddings and the task-specific information from fine-tuning can provide a performance boost in sentiment analysis tasks. By maintaining one set of frozen vectors while allowing another to update, the model benefits from both general word relationships and the ability to adapt to specific patterns in the dataset.

DOI: 10.54254/2755-2721/94/2024MELB0085

Non-static embeddings also performed competitively, especially in architectures that are designed to capture sequential dependencies, such as LSTM and GRU. For instance, the GRU-non-static (0.8901) and Bi-LSTM-non-static (0.8907) models showed high accuracy. In these models, the ability to fine-tune word embeddings allowed the network to adapt the word vectors based on the task, which proved to be advantageous in handling sequential data. However, the performance difference between static and non-static embeddings in some cases was minimal, highlighting that fine-tuning embeddings does not always guarantee better performance, particularly when the architecture already excels in sequential pattern recognition.

Despite not being updated during training, static embeddings delivered strong performance across several models. The LSTM-static model, for example, achieved an accuracy of 0.8888, demonstrating that pre-trained embeddings like Word2Vec can still provide a robust foundation for classification tasks, especially when used with architectures like LSTM that inherently capture long-term dependencies in the data. This indicates that even without task-specific fine-tuning, static embeddings can remain effective in certain architectures, particularly those that are well-suited to processing sequential data.

Overall, the results indicate that multichannel embeddings yielded the best performance in most cases, but the gains over non-static embeddings were sometimes marginal. On the other hand, static embeddings—despite their simplicity—performed surprisingly well, particularly in sequential models like LSTM and GRU, which can compensate for the limitations of fixed word vectors by leveraging their ability to capture long-term dependencies in text.

3.3. Limitations and future work

One limitation of this study is its focus on a single task—sentiment analysis using the IMDb dataset. While this dataset provides valuable insights into model performance, further experiments on other datasets or more complex NLP tasks could offer a broader understanding of how different embeddings and architectures perform across various domains. Additionally, although this work fine-tuned non-static embeddings, no further hyperparameter optimization was performed, leaving room for potentially better results with more extensive tuning.

Future research could explore more advanced fine-tuning techniques for static embeddings to enhance performance further. Additionally, comparing the results of these traditional embeddings with contextual embeddings like BERT or GPT under similar computational constraints could provide deeper insights into the trade-offs between model complexity and efficiency. Finally, experimenting with other NLP tasks, such as text summarization or named entity recognition, would help validate the generalizability of these findings.

4. Conclusion

This study compared the performance of three different Word2Vec-based embedding strategies—static, non-static, and multichannel—across five neural network architectures: CNN, LSTM, Bidirectional LSTM, GRU, and Bidirectional GRU. The primary goal was to investigate how fine-tuning static embeddings (non-static and multichannel) impacts model performance on sentiment analysis tasks.

The results indicated that multichannel embeddings, which combine static and non-static embeddings, provided the best performance across most neural architectures. The CNN-multichannel and bi-GRU-multichannel models, in particular, achieved the highest accuracy scores in the experiments, highlighting the effectiveness of leveraging both frozen pre-trained embeddings and fine-tuned embeddings. However, the non-static embeddings also performed competitively, especially in models designed to capture sequential patterns, such as LSTMs and GRUs. The fine-tuning process enabled these embeddings to adjust to the specific context of the sentiment analysis task, though the performance gains over static embeddings were sometimes modest. Static embeddings, despite not being updated during training, showed strong results, particularly in sequential models like LSTM, where the network's ability to capture long-term dependencies compensated for the static nature of the embeddings.

These findings underscore the use of traditional word embeddings like Word2Vec in modern NLP tasks, particularly in resource-constrained environments. While contextual embeddings (e.g., BERT)

DOI: 10.54254/2755-2721/94/2024MELB0085

have become a mainstream method for many tasks, their computational cost may not be feasible in all settings. The results suggest that models leveraging static or non-static embeddings can still yield high-performance levels when computational efficiency and interpretability are priorities.

References

[1] Reshamwala, A., Mishra, D., & Pawar, P. (2013). Review on natural language processing. IRACST Engineering Science and Technology: An International Journal, 3(1), 113-116
[2] Mikolov, T. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781
[3] Johnson, S. J., Murty, M. R., & Navakanth, I. (2024). A detailed review on word embedding techniques with emphasis on word2vec. Multimedia Tools and Applications, 83(13), 37979-38007
[4] Devlin, J. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[5] Gupta, P., & Jaggi, M. (2021). Obtaining better static word embeddings using contextual embedding models. arXiv preprint arXiv:2106.04302
[6] Devlin, J. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[7] Brown, T. B. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165.
[8] Rakhlin, A. (2016). Convolutional neural networks for sentence classification. GitHub, 6, 25.
[9] Lakshmipathi, N. (2018). IMDB Dataset of 50K Movie Reviews. URL: https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews. Last Accessed: 2024/09/12
[10] Enríquez, F., Troyano, J. A., & López-Solaz, T. (2016). An approach to the use of word embeddings in an opinion classification task. Expert Systems with Applications, 66, 1-6.
[11] Acosta, J., Lamaute, N., Luo, M., Finkelstein, E., & Andreea, C. (2017). Sentiment analysis of twitter messages using word2vec. Proceedings of student-faculty research day, CSIS, Pace University, 7, 1-7.
[12] Sadeeq, M. A., & Abdulazeez, A. M. (2020). Neural networks architectures design, and applications: A review. In 2020 International Conference on Advanced Science and Engineering, 199-204.