Exploration of Word Embeddings with Graph-Based Context Adaptation for Enhanced Word Vectors

Tanvi Sandhu
School of Computer Science
University of Windsor
tanvisandhu1898@gmail.com

Ziad Kobti
School of Computer Science
University of Windsor
kobti@uwindsor.ca

Abstract

In the aspect of information storage, text assumes a central role, necessitating streamlined and effective methods for swift retrieval. Among various text representations, the vector form stands out for its remarkable efficiency, especially when dealing with expansive datasets. Arranging words that are similar in meaning close to each other in the vectorized representation helps improve how well the system performs in different Natural Language Processing related tasks. Previous methods, primarily centered on capturing word context through neural language models, have fallen short in delivering high scores for word similarity problems. This paper investigates the connection between representing words in vector form and the improved performance and accuracy observed in Natural Language Processing tasks. It introduces a method to represent words as a graph, aiming to preserve their inherent relationships and to enhance overall capabilities in semantic representation. Experimental deployment of this technique across diverse text corpora underscores its superiority over conventional word embedding approaches. The findings contribute to the evolving landscape of semantic representation learning but also illuminates their implications for text classification tasks, especially within the context of dynamic embedding models.

Introduction

Natural Language Processing (NLP) is a rapidly evolving field which gives computers the ability to understand and process human language in the form of text. NLP plays a crucial role in various applications, including chatbots, virtual assistants, and language translation services. One of the challenges in NLP is handling the diversity and complexity of human language, including idioms, slang, and cultural variations.

In NLP, the representation of words is a fundamental aspect that bridges the gap between linguistic elements and computational models. For an NLP task, the words must be represented numerically for the computers to understand and process them (Prakash et al. 2011). Initially in Computer Linguistics, indexing was used for word representation which could not achieve good results due to their inability to capture context and similarity in the text. For problems like text classification(Kamran et al. 2019) and text generation(Kun J et al. 2019), this method did not yield good results, the reason being that the correct syntactic meaning of the word cannot be justified as a single index number.

To alleviate this, words are typically transformed into numerical vectors, known as word embeddings. These embeddings encode semantic and syntactic information, capturing the relationships between words in a high-dimensional space. The significance of word representation lies in its ability to preserve contextual meaning and nuances, enabling machines to comprehend language based on the surrounding context. This numerical representation facilitates various NLP tasks, including word similarity measurement, sentiment analysis, and language translation.

Obtaining vector representations for words is an essential initial step. Word vectors, or embeddings, convert text data into a numerical format comprehensible to machine learning algorithms. Various techniques are employed for this purpose, including frequency-based methods like one-hot encoding, bag of words, N-gram, and TF-IDF, as well as prediction-based methods such as static and dynamic embedding. However, while frequency-based approaches focus on statistical co-occurrence and may result in sparse vectors, prediction-based methods offer more nuanced representations by considering semantic meaning and contextual relationships among words.

In the domain of word embeddings, there are two main categories of prediction-based methods: static embedding and dynamic embedding. Static embeddings, also known as fixed or pre-trained embeddings, assign a consistent vector representation to each word or token, regardless of its context within a specific sentence or document. While they are computationally less expensive and require less memory, they struggle to capture nuances of meaning that arise from different contexts. Models like Word2Vec (Mikolov et al. 2013) and Glove (Pennington J et al. 2014) were focused on capturing distributed representations of words based on co-occurrence statistics. These static embedding models do not consider the context in which words appear, leading to challenges in capturing nuances and variations in meaning depending on the surrounding words.

On the other hand, dynamic embeddings, also known as contextual embeddings, are designed to understand variations in meaning based on the words that surround each word or token. These embeddings capture the meaning of a word in relation to its surrounding context, offering a more nuanced representation. However, they are often larger models and require more computational resources. ELMo (Peters et al. 2018), for instance, employs deep contextualized word representations by considering the entire input sentence. It utilizes a bidirectional LSTM (Long Short-Term Memory) to capture the context on both the left and right sides of a word. This results in embeddings that vary depending on the specific instance of a word within a sentence, addressing challenges posed by polysemy and context-dependent meanings. Similarly, GPT(Yenduri et al. 2023), a state-of-the-art transformer-based model, adopts a pre-training strategy to generate contextual embeddings. During the training process, the GPT model takes into consideration the entire context of a word which seizes the intricate patterns and dependencies.

In NLP, the similarity between two words is often judged based on their frequent co-occurrence (Dan Jurafsky and James H 2014). However, this measure of similarity is flawed because certain words may frequently appear together without necessarily sharing similar meanings. Consider the words "cup" and "coffee" in the following examples:

"Can I have a cup of coffee?"

"I am going to have a cup of coffee."

These phrases might often occur in a text corpus, especially in a dialogue corpus. These words appear near each other frequently because they co-participate in an event that happens every day, but that does not directly translate to their similarity. Even though these words belong to the same semantic scope, they cannot be considered similar. Words can be associated with each other in more ways than one way, and this kind of association between words is known as 'relatedness' (Alexander B and Graeme H 2006). On the other hand, word similarity aims to associate words with similar semantic meanings together(Aminul I and Diana I 2008). On the other hand, there might be some words that occur in the same corpus but they are used in vastly different contexts. Thus, their embeddings would get generated far from each other, which hurts their similarity and relatedness.

Now lets consider the word "bat" in two sentences:

"He swung the bat and hit a home run"

"At dusk, bats emerge from their caves to hunt for insects."

In the first sentence, "bat" is associated with sports equipment, while in the second, it refers to a flying mammal. Traditional word vectors find it difficult to get the context hence it will not form different vectors for word bat. It will form one static embedding based on the average meaning in both contexts like below:

$$\text{Vector}_{static}\ \text{bat}$$

Here the one static vector of "bat" is formed because "bat" is treated as one point in space irrespective of its meaning in the two sentences. This limitation is addressed by contextual word embeddings, which consider the surrounding context of a word in a sentence. Below are the vectors generated for the word bat:

$$\text{Vector}_{contextual}\ \text{bat}^{sentence1}$$
$$\text{Vector}_{contextual}\ \text{bat}^{sentence2}$$

Each of these vectors has distinct representations capturing the contextual meaning of the word "bat" in the respective sentences. Unlike traditional static embeddings, which provide a single, averaged representation for a word across different contexts, contextual word embeddings, such as those obtained from models like ELMo, BERT, or GPT, consider the specific context in which the word appears.

This allows the model to generate distinct embeddings for "bat" based on its specific usage, enabling more accurate representation and comprehension of the word's varied meanings in different contexts. The ability of contextual embeddings to adapt to the surrounding context makes them valuable in addressing the limitations of static word vectors in capturing the richness of language semantics.

The integration of graph-based representation learning with NLP models extends the capabilities of traditional approaches. The graph helps us to visualize the word embedding in a more structured way. It can also help us handle polysemy or homonymy challenges. The use of weights on edges in a graph-based word embedding model plays a crucial role in capturing the strength or importance of relationships between words.

In conclusion, this paper explores the dynamic realm of natural language and its significance across various applications, from chatbots to language translation services. Challenges arising from the complexity of human language, such as idioms and cultural nuances, have driven advancements in word representation. Traditional static embeddings like word2vec and GloVe have limitations in capturing nuanced meanings due to their lack of contextual awareness. This paper aims to address these limitations by delving into contextual word embeddings and semantic textual similarity, contributing to ongoing NLP research and enhancing language understanding.

Problem Statement

Hypothesis: Since Graphs are known to perform better in representing entities with contextual relations, using Graph Embedding methods on a relational graph with every word represented as a node and each node connected to the words in its context would yield better results for the tasks in NLP involving relations between words.

The mathematical problem formulation of the problem statement is as follows:

Task 1: Given a text Corpus C containing n words, find a graphical representation G where each word pair acts as nodes and each node is connected with a weight w. This graph G is derived from matrix M[n][n] after formulating a semantic relationship, where n is the number of words after preprocessing the text corpus C

Task 2: Update the initial weight of edges to get the best of the context.

Literature Review

Over the decades, the field of NLP has witnessed significant advancements, driven by a continuous quest to unravel the complexities of human language and enhance computational understanding. Pioneering studies such as "Selected Studies of the Principle of Relative Frequency in Language" (Kingsley Zipf. 1932) delved into statistical regularities in word frequency and meaning distribution, laying foundational groundwork for subsequent research in word representation. Building upon this groundwork, "An Experimental Study of Ambiguity and Context" (Kaplan and Abraham. 1955) demonstrated the potential of context in disambiguating vague or ambiguous terms within sentences, highlighting the importance of contextual awareness in language comprehension. Philosophical works like "Philosophical Investigations" (Wittgenstein, Ludwig. 1953) emphasized the evolving and flexible nature of language, inspiring contemplation on the dynamic interplay between linguistic structures and meaning. Transitioning into the realm of computational semantics, "A Tractable Machine Dictionary as a Basis for Computational Semantics" (Wilks, Y et al. 1990) marked a significant milestone in the development of knowledge-based approaches, paving the way for sophisticated techniques in semantic representation.

The emergence of modern word embedding techniques reshaped the landscape of NLP research. Word2Vec, introduced a method for generating low-dimensional vector representations of words, revolutionizing semantic modeling by capturing intricate semantic relationships, while GloVe employed global statistics to create embeddings.

Predicting the next word in NLP is a tough job because language is complex. Words can have different meanings based on the context in which they are used. Sentence structures vary a lot, and there are many words that mean almost the same thing. Figuring out the next word or linking two words together is not easy. Also, words often have multiple meanings, adding more complexity. To deal with these challenges, smart models and techniques, like neural language models with attention mechanisms, have been created to understand the patterns and connections in language. Bengio tried to predict the next word in their paper(Bengio, Y. 2000). They used a neural network with Long Short-Term Memory (LSTM) units to figure out how words work together in sentences. The model assigned an index to each unique word and used a feed-forward network to predict the next word based on previous ones. It was good at understanding long connections between words, but it had some problems, like being computationally complex and needing a lot of examples to learn well.

In the world of NLP, connecting important words is like an evolution story. From basic sparse encoding to advanced contextual embeddings, the journey has changed how we understand and use words. Word vectors have been crucial in solving this problem.

The development of word vectors in NLP has followed a changing path, showing progress in neural network designs. Early methods using sparse vectors like one-hot encoding couldn't capture the rich meaning in language. Then came distributional semantic models like CBOW and skip-gram, which allowed creating dense vectors that understand complex relationships in language. Pre-trained word embeddings like Word2Vec, GloVe, and fastText became important, giving efficient representations learned from lots of data. Now, contextualized embeddings like those in BERT(Devlin, J. 2018) have become popular, improving word representations by understanding context and word frequencies in documents. This ongoing evolution shows a continuous effort to have better and context-aware word vector representations in NLP.

As the field advances, using graph-based representations has become a new idea. Faruqui introduced innovative graph-based techniques to enhance word embeddings by incorporating semantic knowledge from resources like WordNet (Faruqui et al. 2014). By leveraging existing lexical knowledge, the paper aimed to refine word representations, thereby improving performance in various NLP tasks. However, despite its contributions, the approach had several drawbacks. Firstly, the reliance on pre-existing semantic lexicons like WordNet limited the scope of the technique to the coverage and accuracy of these resources. Secondly, the retrofitting process itself was computationally expensive, involving complex graph-based operations to adjust word embeddings. By focusing primarily on semantic relationships derived from lexicons, the method might have neglected other important linguistic features relevant for tasks such as sentiment analysis or named entity recognition.

Expanding on this concept, "Graph Convolutional Networks for Text Classification" (Yao et al. 2018) sought to address some of these limitations by leveraging graph structures to capture semantic relationships directly from data. Despite its promise, this approach also had its drawbacks. For instance, the complexity of graph convolutional networks made them computationally intensive, requiring substantial resources for training and inference. Additionally, the effectiveness of the approach heavily relied on the quality and structure of the constructed graph, which could be challenging to optimize or generalize across different datasets or domains.

Methodology

The task of generating vector embedding for words according to our method is a semi-supervised task because it involves a step where the graph needs to be created from a text corpus, which is essentially assigning relations to non-relational data. This solution involves 3 key tasks:
(1) Graph creation,
(2) Graph Enhancement

The initial phase of graph formation is a crucial step in elucidating the semantic structure inherent within the text. Here, the graph serves as an essential structural representation of the input dataset, comprising three fundamental com-

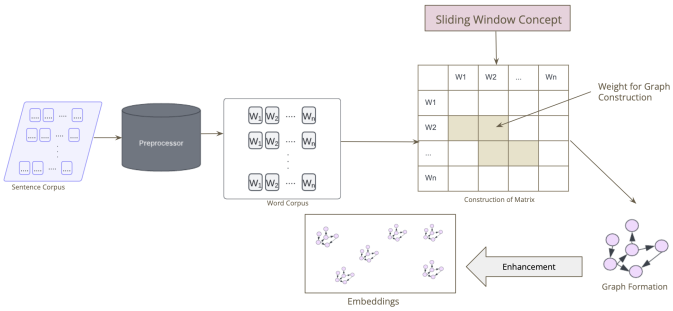

Figure 1: A Flowchart of the methodology

ponents: nodes, edges, and weights. These nodes, integral to the construction of the graph, encompass the vocabulary generated from the input text corpus. This vocabulary, curated through processes such as sentence tokenization and lemmatization, is refined, with the removal of stop-words to enhance the quality of the vocabulary. Following this preparatory phase, the edges of the graph are established, denoting the intricate relationships that exist among the words acting as nodes. These relationships, paramount in reflecting semantic connections, are meticulously defined, thus imbuing the graph with meaningful associations. At the heart of delineating these relationships lies the utilization of the sliding window concept, an ingenious mechanism that systematically traverses the text corpus. By virtue of this approach, a fixed window dynamically moves through the corpus, capturing local associations between words within its purview. These associations, garnered through meticulous examination of co-occurrences within the sliding window, serve as the bedrock for the establishment of edges in the graph. Thus, through the adept utilization of the sliding window concept, the graph's structure emerges, meticulously capturing the intricate web of semantic relationships that underpin the text corpus.

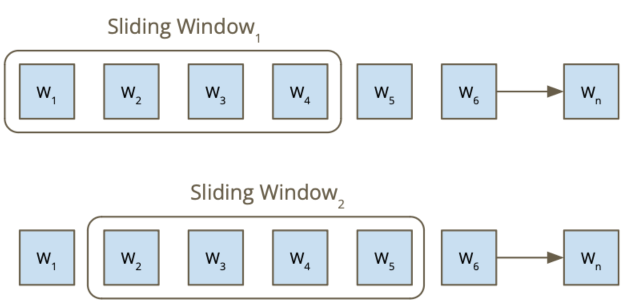

Figure 2: Sliding Window Concept

The sliding window method is a technique wherein a fixed-size window traverses across the text corpus, systematically uncovering local features or patterns between words. As this window glides over the corpus data, the text contained within it serves as input for evaluating the frequency of word pairs. This frequency, symmetrically recorded in a matrix, acts as a significant weight in the resultant graph. Consequently, upon the completion of graph formation, redundant edges are effectively eliminated if no relationships exist between the words under consideration.

In Figure 2, denoted by W_1, W_2, W_3, W_4, and so on, we have a representation of a set of words derived from the corpus. Here, a sliding window of size 4 is visibly depicted, initially spanning the first four words. Within this window, the individual words function as input for computing the relationship, which is then promptly updated in the matrix. Following this computation, the window incrementally shifts from its original position at W_1 to the subsequent word, W_2, in the subsequent iteration. This iterative process continues as the window traverses through the corpus, systematically capturing and updating the relationships between words along the way.

The co-occurrence Matrix for the graph we get from Task 1 is a sparse matrix with not many associations between words. This is true for most word embedding models as well. So, in Task 2, our goal is to enhance this co-occurrence matrix so that the output is a denser, much more comprehensive co-occurrence matrix. Lets consider co-occurrence matrix M, where V is the vocabulary size of corpus C. Hence the matrix generated is M(V×V).

Graph enhancement is the next step in the process. There are many preexisting steps to enhance a graph like Node2Vec, Word2Vec and many more. The technique used here is DeepWalk (Perozzi B et al. 2014) and LINE (Tang J et al. 2015). DeepWalk's emphasis on preserving local structure through random walks aligns with the inherent intricacies of semantic relationships among words in the input dataset. By treating the vocabulary as nodes and simulating random walks, DeepWalk effectively learns embeddings that encapsulate the contextual meaning of words. On the other hand, LINE complements this approach by optimizing both local and global structures of the graph, offering a comprehensive representation of semantic associations. The efficiency of LINE in handling large-scale graphs is particularly advantageous, ensuring scalability and performance even with extensive vocabularies. The combination of DeepWalk and LINE thus contributes to a robust methodology for enhancing the initially constructed graph, providing a refined semantic representation that captures the subtle relationships among words in the input text corpus.

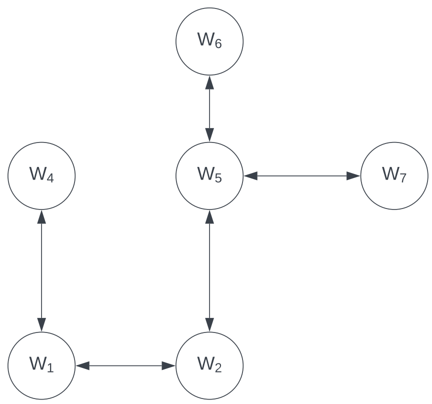

Figure 3: Graph Generated

Figure 3 depicts the graph generated from the dataset where words W $_1$, W $_2$, W $_3$, W $_4$, and so on represent nodes of the graph. The edges are defined only for the nodes having a weight in the matrix, hence no redundant edges are present in this graph.

Experimental Setup

To assess the efficacy of our method, we subjected all the algorithms to word similarity and word analogy tasks for evaluation. The training was done on the 20 News Groups Dataset (K.Lang. 1995).

Datasets

SimLex-999 (Felix H et al. 2015) has emerged as a widely utilized resource for monitoring word similarity. This benchmark dataset focuses on assessing the similarity between words, distinct from relatedness or association. It comprises 666 noun-noun pairs, 222 verb-verb pairs, and 111 adjective-adjective pairs. The dataset serves the purpose of evaluating word similarity, involving an exploration of the relationship between two words based on the algorithm used and subsequent comparison with human-annotated references. The performance of the task is measured using the Spearman Correlation as the evaluation metric.

The SemEval-2012-Task2 (David A et al. 2012) dataset stands out as a significant asset in the realm of natural language processing, specially crafted for the assessment of Word Analogy. This dataset is centered around appraising the level of similarity between pairs of words, presenting a varied collection of lexical items for a thorough evaluation. Covering a range of tasks related to word similarity across diverse languages and domains, the dataset comprises annotated word pairs, with human raters assigning similarity scores. The evaluation of this task relies on the use of Spearman Correlation to gauge the alignment between model predictions and the similarity scores provided by human annotators.

Baselines

DeepWalk is engineered to acquire continuous vector representations, known as embeddings, for nodes within a network. These embeddings are adept at encapsulating both the structural intricacies and interconnections among nodes in the network, effectively placing them within a continuous vector space. In the context at hand, DeepWalk is applied to pre-saved embeddings, playing a pivotal role in social network analysis. The algorithm undertakes random walks on the graph, employing a Word2Vec like methodology to learn distributed representations for each node. Consequently, these node embeddings encapsulate both the structural layout and contextual nuances of the nodes in the graph.

LINE, as a network embedding algorithm, surpasses the limitations of exclusively capturing local proximity. Its objective is to safeguard both first-order, signifying direct connections, and second-order, indicating common neighbor relationships, proximity among nodes. This is accomplished through the formulation of an objective function that seeks to maximize the likelihood of observing existing edges while minimizing the likelihood of non-existing edges.

Results

Models like Deepwalk and LINE were used to generate node embeddings, capturing semantic information about the words in the corpus. These embeddings are then saved for further use to compare these with the other language models like Word2Vec, TF-iDF, GloVe, BERT, XLNET (Yang et al. 2019) and GPT. The datasets SimLex999 and Semeval are used in the experiments to compare the word embeddings. The Table 1 shows the value comparison of the results with Spearmann Correlation as the comparison metric.

	Word Similarity	Word Analogy
	Simlex-999	SemEval
Word2Vec	0.0014	0.038
TF-iDF	0.0001	0.014
GloVe	0.0028	0.056
BERT	0.0143	0.012
GPT	0.034	0.019
XLNET	0.080	0.022
Graph+DEEPWALK	0.114	0.097
Graph+LINE	0.053	0.062

Table 1: Comparison of Graph generated embedding with other model using Spearmann Correlation results for evaluation

The Table 1 presents a comprehensive comparison of word embedding models, assessing their performance in word similarity tasks across diverse datasets—SimLex-999 and SemEval. The models under scrutiny include prominent ones like BERT, GPT, XLNET, and two innovative approaches integrating graph-based context adaptation (Graph+DEEPWALK, Graph+LINE). The evaluation metric is Spearman Correlation, measuring the alignment between model predictions and human-annotated similarity scores. Higher correlation values indicate better proficiency in capturing semantic relationships among words. Notably, Graph+DEEPWALK outshines other models in SimLex-999 datasets, showcasing its efficacy in enhancing word vector's semantic understanding. The better results for this task, which is word similarity, can be attributed to the fact that representing words based on their occurrences in a graph format would result in exploring some relationships that conventional algorithms would not be able to. GPT and XL-NET also demonstrate competitive performances. However, the graph-based models, which leverage structural information from semantic graphs, exhibit a nuanced understanding of word relationships, particularly evident in SemEval. This proves that representing text as graphs also embed similar words together while also preserving other relationships between the words. Also the relatively small differences in correlation scores across the evaluated word embedding models as depicted in the table may be attributed to several factors one of which can be due to the choice of evaluation dataset and task. In this case, the evaluation is based on the Spearmann Correlation coefficient using datasets like Simlex-999 for word similarity and SemEval for word analogy. These datasets may not fully capture the diverse range of semantic relationships present in natural language, leading to relatively consistent performance across models. Furthermore, the complexity of natural language and the inherent ambiguity of word meanings can also contribute to the convergence of performance metrics across different models. The table underscores the importance of graph-based context adaptation in refining word embeddings, offering insights into model's abilities to comprehend the intricacies of semantic connections.

Conclusion

In conclusion, this paper significantly adds to the dynamic landscape of Natural Language Processing by presenting an innovative method for word representation, achieved through the creation and refinement of semantic graphs. This approach, which synergizes the capabilities of graph-based representation learning and advanced embedding techniques, outperforms existing models in word similarity tasks. Moreover, the comparison with established models vividly underscores the effectiveness of the proposed approach. It underscores the crucial aspect of capturing contextual nuances for precise language representation. This emphasis on context-driven understanding positions the proposed methodology as a noteworthy advancement in NLP. Several avenues for research and development can be explored to build upon the findings and methodologies presented such as integrating information from diverse modalities within a unified graph framework could enhance the model's ability to capture complex semantic relationships across different types of data. Ongoing advancements in NLP, as exemplified by this study, continuous progress strives to navigate the intricate challenges of human language, fostering the development of more precise and adaptable language models with applications across a multitude of fields and domains.

References

Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman 2011. *Natural language processing: an introduction* Journal of the American Medical Informatics Association, 18(5), 544–551.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown 2019. *Text classification algorithms: A survey* Information, 10(4), 150.

Kun Jing and Jungang Xu 2019. *A survey on neural network language models* arXiv preprint arXiv:1906.03591.

Dan Jurafsky and James H Martin 2014. *Speech and language processing. vol. 3*, US: Prentice Hall.

Alexander Budanitsky and Graeme Hirst 2006. *Evaluating wordnet-based measures of lexical semantic relatedness*, Computational linguistics, 32(1), 13–47.

Aminul Islam and Diana Inkpen 2008. *Semantic text similarity using corpus-based word similarity and string similarity*, ACM Transactions on Knowledge Discovery from Data (TKDD), 2(2), 1–25.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. *Efficient estimation of word representations in vector space*, arXiv preprint arXiv:1301.3781.

Marco Baroni and Alessandro Lenci 2011. *How we blessed distributional semantic evaluation* in Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics, pp. 1–10.

Romanova, A. 2021. *Semantics graph mining for topic discovery and word associations.* International Journal of Data Mining and Knowledge Management Process, 11(04), 01–14. https://doi.org/10.5121/ijdkp.2021.11401

Karen Sparck Jones 1972. *A statistical interpretation of term specificity and its application in retrieval* Journal of documentation.

Pennington, J., Socher, R., and Manning, C. D. 2014. *Glove: Global vectors for word representation* In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. *Simlex-999: Evaluating semantic models with (genuine) similarity estimation* Computational Linguistics 41(4):665–695.

David A Jurgens, Peter D Turney, Saif M Mohammad, and Keith J Holyoak. 2012. *Semeval-2012 task 2: Measuring degrees of relational similarity* In Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation. Association for Computational Linguistics, pages 356–364.

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* arXiv preprint arXiv:1810.04805.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. 2019. *XLNet: Generalized Autoregressive Pretraining for Language Understanding* arXiv preprint arXiv:1906.08237.

Perozzi B., Al-Rfou, R., and Skiena, S. 2014. *DeepWalk: Online Learning of Social Representations* arXiv preprint arXiv:1403.6652.

Reimers, N., and Gurevych, I. 2019. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks* arXiv preprint arXiv:1908.10084

Tang J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. 2015. *LINE: Large-scale Information Network Embedding* arXiv preprint arXiv:1503.03578.

Ken Lang. 1995. *Newsweeder: Learning to filter netnews, in Machine Learning Proceedings* 331–339.

Rudolph, M., Blei, D. M., and Griffiths, T. L. 2016. *Dynamic Word Embeddings for Evolving Semantic Discovery* arXiv preprint arXiv:1703.00607.

Last, M., Klein, T., and Kandel, A. 2019. *Using Graphs for Word Embedding with Enhanced Semantic Relations* arXiv preprint arXiv:1911.01678.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D. and Stoyanov, V. 2019. *Roberta: A robustly optimized bert pretraining approach* arXiv preprint arXiv:1907.11692.

Zuckerman, M., and Last, M. 2019. *Using graphs for word embedding with enhanced semantic relations* In Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13) (pp. 32-41).

Kingsley Zipf, G., 1932. *Selected studies of the principle of relative frequency in language* Harvard university press.

Kaplan, Abraham 1955. *An experimental study of ambiguity and context* Mech. Transl. Comput. Linguistics 2: 39-46.

Wittgenstein, Ludwig 1953. *Philosophical Investigations* New York, NY, USA: Wiley-Blackwell. Edited by G. E. M. Anscombe.

Wilks, Y., D. Fass, C.M. Guo, J. McDonald, T. Plate and B. Slator 1990. *A Tractable Machine Dictionary as a Basis for Computational Semantics* Journal of Machine Translation, 5.

Bengio, Y., Ducharme, R. and Vincent, P., 2000. *A neural probabilistic language model* Advances in neural information processing systems, 13.

Faruqui, M., Dodge, J., Jauhar, S.K., Dyer, C., Hovy, E. and Smith, N.A., 2014. *Retrofitting word vectors to semantic lexicons* arXiv preprint arXiv:1411.4166.

Yao, L., Mao, C. and Luo, Y., 2019. *Graph convolutional networks for text classification* In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 7370-7377).

Yuxuan Wang, Yutai Hou, Wanxiang Che, and Ting Liu 2020. *From static to dynamic word representations: a survey* International Journal of Machine Learning and Cybernetics, 11, 1611–1630.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel 2015. *Gated graph sequence neural networks* arXiv preprint arXiv:1511.05493.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa 2009. *A study on similarity and relatedness using distributional and wordnet-based approaches*.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent 2000. *A neural probabilistic language model* Advances in neural information processing systems, 13.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa 2011. *Natural language processing (almost) from scratch* Journal of machine learning research, 12(ARTICLE), 2493–2537.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith 2014. *Retrofitting word vectors to semantic lexicons* arXiv preprint arXiv:1411.4166.

Aditya Grover and Jure Leskovec 2016. *node2vec: Scalable feature learn- ing for networks* in Proceedings of the 22nd ACM SIGKDD interna- tional conference on Knowledge discovery and data mining, pp. 855– 864.

Zellig S Harris 1954. *Distributional structure* Word, 10(2-3), 146–162.

George A Miller 1995. *Wordnet: a lexical database for english* Communications of the ACM, 38(11), 39–41.

Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang 2019. *A review of recurrent neural networks: Lstm cells and network architectures* Neural computation, 31(7), 1235–1270.

Peters, M.E., Neumann, M., Zettlemoyer, L. and Yih, W.T., 2018. *Dissecting contextual word embeddings: Architecture and representation* arXiv preprint arXiv:1808.08949.

Staudemeyer, Ralf C., and Eric Rothstein Morris 2019. *Understanding LSTM-a tutorial into long short-term memory recurrent neural networks* arXiv preprint arXiv:1909.09586.

Vaswani, Ashish, et al 2017. *Attention is all you need* Advances in neural information processing systems 30.

Yenduri, Gokul 2023. *Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions* arXiv preprint arXiv:2305.10435.

Gastaldi, J.L 2021. *Why Can Computers Understand Natural Language?* Philos. Technol. 34, 149–214.

Zhang, Yin, Rong Jin, and Zhi-Hua Zhou 2010. *Understanding bag-of-words model: a statistical framework* International journal of machine learning and cybernetics 1: 43-52.

Cavnar, William B., and John M. Trenkle 1994. *N-gram-based text categorization* Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval. Vol. 161175.

Ramos, Juan. "Using tf-idf to determine word relevance in document queries." Proceedings of the first instructional conference on machine learning. Vol. 242. No. 1. 2003.