Uncovering Wonderland: Word Embeddings in Lewis Carroll's Fantasy Novel

Ariana-Andra Șerpar
Faculty of Mathematics and Computer Science
West University of Timișoara
Timișoara, Romania
ariana.serpar00@e-uvt.ro

Erik Zsolt Varga
Faculty of Mathematics and Computer Science
West University of Timișoara
Timișoara, Romania
erik.varga98@e-uvt.ro

***Abstract*—With recent events capturing global attention, the emergence of ChatGPT has generated shockwaves and thus fueled the popularity of Natural Language Processing (NLP). While these colossal transformer models, with their multitude of parameters, have garnered significant attention, older, established models in NLP have lingered in the shadows, despite their proven efficiency and track record of success. In this paper we aim to showcase and promote these models by utilizing them to generate word embeddings, and then comparing them against embeddings created by transformer models. By using the renowned fantasy novel, "Alice's Adventures in Wonderland" as our arguably limited-sized corpus, we will obtain commendable results in terms of word similarities and visualizations of the word embedding vectors.**

***Index Terms*—Natural Language Processing, Word Embeddings, CBOW, Skip-gram, Transformers**

I. Introduction

Distributional semantics, word embeddings, contextual embeddings as well as the representation of words as continuous vectors have played a crucial role in Natural Language Processing (NLP) research. These techniques enable us to capture the semantic and syntactic properties of words and phrases in a high-dimensional space. In the following paragraphs we will provide brief definitions of these terms, touch upon the historical context, and outline the objectives of our paper.

Distributional semantics refers to the idea that words with similar meanings tend to appear in similar contexts. This concept lies at the foundation of building **word embeddings**, which are dense vector representations that capture the meaning of words based on their distributional properties. Word embeddings have revolutionized NLP by enabling machines to comprehend the meaning of words and perform various language-related tasks.

Contextual embeddings take the idea of word embeddings further by considering the surrounding context in which a word appears. Unlike traditional word embeddings, which assign a single vector representation to each word, contextual embeddings generate dynamic representations that capture the word's meaning based on its context. This approach has shown significant improvements in capturing word meaning nuances and handling polysemy.

The representation of words as continuous vectors dates back to the early works of Zellig Harris in the 1950s [6], who introduced the concept of distributional semantics. However, it was not until the recent advancements in deep learning and large-scale datasets that word embeddings became widely adopted and highly effective.

Attempting to merge old literary classics with new technology, the primary focus of out paper is on creating word embeddings based on Lewis Carroll's 1865 novel, "Alice's Adventures in Wonderland" [3]. While the book is relatively short compared to other text corpora commonly used in NLP, we do not see this as a shortcoming, but rather as a challenge. Despite the reduced amount of words and sentences, we aim to achieve compelling results, particularly in word similarity evaluation and visualizations.

II. Related Work

This section will consist of two parts: 1) related work for models we can obtain word and contextual embeddings from and 2) papers that use NLP to analyze "Alice's Adventures in Wonderland".

Word2Vec [8], [11] is a neural network-based model that learns word embeddings through unsupervised learning. It leverages the co-occurrence patterns of words in a given corpus by combining two main techniques: Continuous Bag-of-Words (CBOW) and Skip-gram (SG).

CBOW predicts the target word based on the surrounding context words. It takes the context words as input and predicts the target word as output. CBOW is efficient and faster to train compared to the SG model.

SG, on the other hand, predicts the context words given a target word. The target word serves as input, and the context words are the outputs. SG is generally more effective on datasets with long sentences and a low number of samples. The training process involves adjusting the neural network weights to minimize the loss function using techniques such as stochastic gradient descent.

Word2Vec has gained popularity for its efficiency, scalability, and ability to capture meaningful word representations. It has become a fundamental technique in the field of NLP, paving the way for subsequent advancements in word embedding models.

FastText [1], an extension of Word2Vec, focuses on capturing subword information. It considers character-level n-grams within words, allowing it to handle out-of-vocabulary (OOV) words and effectively represent rare or morphologically rich words. FastText generates word representations by summing up the embeddings of constituent character n-grams. FastText is particularly effective for text classification tasks and can represent both individual words and the overall structure of sentences or documents.

Bidirectional Encoder Representations from Transformers (BERT) [4] is a transformer-based model designed by Google AI Language, able to comprehend and generate human-like language representations. BERT learns to predict masked words within a sentence and determine if two sentences follow each other in a given document during pre-training. It captures both local and global context by considering the complete context of a word through bidirectional understanding. BERT utilizes a transformer architecture with self-attention and feed-forward neural networks to capture long-range dependencies and contextual relationships between words. It can be fine-tuned for specific downstream tasks like text classification and named entity recognition.

Generative Pre-trained Transformer (GPT) [10] is a transformer-based language model developed by OpenAI. It generates coherent and contextually appropriate text based on input. GPT utilizes a transformer architecture and is trained on a massive corpus of text data using unsupervised learning. It excels at tasks like text generation, text completion, and dialogue systems. Unlike BERT, which focuses on language understanding, GPT is primarily a generative model.

There are several papers that use NLP to analyze the book “Alice’s Adventures in Wonderland”. In a 2018 case study [5] two mesoscopic networks were constructed with different thresholds for pruning connections. The networks exhibit a chain-like structure that reflects the order of paragraphs in the book. Connections between distant nodes indicate regions of high contextual similarity. The connectivity patterns among chapters reveal relationships between the content and the structure of the book.

In a more recent case study [2] analyzing the book using NLP tools, the author explores 58 different visualization techniques applied to “Alice’s Adventures in Wonderland”. The study emphasizes the importance of diverse visualizations in digital humanities, spanning areas like humanities research, NLP, fine arts, and computer graphics. Examples include word trees, tag clouds, graphs, storylines, parse trees, and experimental visualizations. The paper suggests additional techniques like sub-word visualizations, word pair visualizations, and chapter/document visualizations to enhance literary analysis.

III. Our contribution

The main contribution of this paper to the field of NLP is centered around training multiple machine learning models using the famous book “Alice’s Adventures in Wonderland” by Lewis Carroll. Despite the relatively small text corpus, with the book containing approximately 14,262 words and 1,102 sentences, we aimed to tackle this exciting challenge and attain passable results at the same time. By doing so, our aim is to generate word embeddings that capture the semantic and contextual information present in the book. This unique approach allows us to explore the representation of words within the context of this beloved fantasy novel.

With our work, we hope to inspire and motivate other researchers to analyze and delve deeper into the application of machine learning techniques on a wider range of fantasy novels, enabling a better understanding of the linguistic intricacies and creativity found within these literary works.

Furthermore, our evaluation of the trained word embeddings extends beyond traditional methods, such as similarity or analogy evaluations, to include visual representations. By visualizing the embeddings, we can gain deeper insights into the distribution and relationships between words, thereby facilitating a more comprehensive analysis of the learned representations.

Through these efforts, we seek to contribute not only to the advancement of NLP techniques but also to foster a deeper appreciation and exploration of fantasy literature within the research community. By applying machine learning models to such imaginative texts, we can uncover new perspectives and enrich our understanding of language representation and interpretation.

IV. Experiments

A. Packages and versions

As transparency and reproducibility are some of our main goals, we start this section by providing all the Python (3.10.12) packages used for the experiments with their respective versions (Table I).

TABLE I
Package versions

Package	Version
google.colab	0.0.1a2
nltk	3.8.1
matplotlib	3.7.1
numpy	1.22.4
wordcloud	1.8.2.2
sklearn	1.2.2
gensim	4.3.1
torch	2.0.1+cu118
transformers	4.29.2

B. Preprocessing

Our experiments start with preprocessing the text from the “Alice’s Adventures in Wonderland” book. Through several test cases, we have discovered that the best results are obtained when we convert the text to lowercase, remove punctuation, and eliminate stop words. On the other hand, we observed that applying lemmatization or stemming decreases the quality of the word embedding vectors. However, it’s important to note that for the transformer models, such as BERT and GPT2, we did not preprocess the text in the same manner. Instead, we tokenized the text, which is necessary in order to feed it into the transformer models effectively.

Taking into consideration the relatively small size of the text corpus, the training times are deemed medium to low. The CBOW, SG and FastText took only a few seconds, while the transformer models required a few minutes.

C. Word Similarity Evaluation

Our initial word similarity evaluation involved comparing the embedding word vectors of “king” and “queen”, as well as “king” and “pig”, using cosine similarity. In the case of the Word2Vec and FastText models, the results were good, as expected. The words “king” and “queen” showed more similarity than “king” and “pig” (Table II). Regarding the transformer models, BERT and GPT2, BERT again demonstrated the expected good result, with the word “queen” being closer to “king” than the word “pig”. However, GPT2 showed a higher correlation between the word “king” and “pig” compared to “king” and “queen” (Table III). This mistake could be attributed to the fact that GPT models in general (including GPT2) were primarily designed for text generation.

For easier interpretation, the better result for each model is highlighted in bold.

TABLE II
COSINE SIMILARITY SCORES FOR WORD PAIRS USING WORD2VEC AND FASTTEXT

Model	First word	Second word	Cosine similarity
Word2Vec - CBOW	king	queen	**0.90104**
Word2Vec - CBOW	king	pig	0.40751
Word2Vec - SG	king	queen	**0.99876**
Word2Vec - SG	king	pig	0.99777
FastText	king	queen	**0.99996**
FastText	king	pig	0.99912

TABLE III
COSINE SIMILARITY SCORES FOR WORD PAIRS USING BERT AND GPT2

Model	First word	Second word	Cosine similarity
BERT	King	Queen	**0.24233**
BERT	King	Pig	0.16726
GPT2	King	Queen	0.80859
GPT2	King	Pig	**0.85301**

The second word similarity evaluation we conducted was to examine the topmost similar words to “king” and “queen”. However, we found that the word ”queen” appeared as the 7th most similar word to “king” only in the case of the SG model. In several other instances, different characters, like Alice, appeared as the most similar words, most likely due to their interactions within the book.

Lastly, we measured the quality of our word embeddings with the help of Wordsim353, which is a widely used benchmark dataset for evaluating the semantic similarity of word embeddings. Wordsim353 consists of 353 word pairs, along with human-rated similarity scores, which are used as a ground truth for evaluating the model’s ability to capture semantic relationships between words. By comparing the similarity scores produced by our word embeddings with the human ratings in Wordsim353, we can assess the quality of our embeddings in capturing the semantic similarity between words.

TABLE IV
WORD SIMILARITY EVALUATION ON WORDSIM353 (OOV RATIO IS 91.21813)

Model	Correlation coefficient	Statistic	P-value
Word2Vec - CBOW	Pearson R	-0.00116	0.99501
Word2Vec - CBOW	Significance	-0.23855	0.19622
Word2Vec - SG	Pearson R	-0.13092	0.48266
Word2Vec - SG	Significance	-0.10425	0.57675
FastText	Pearson R	0.25660	0.16347
FastText	Significance	0.05444	0.77112

The evaluation results (Table IV) suggest that the word embeddings generated by the CBOW and SG models have limited similarity with the human-rated similarity judgments in WordSim353. The FastText model shows a slightly stronger correlation, but it may still not capture the semantic similarity as effectively. Further analysis and improvements to the word embedding models may be necessary to achieve better performance on this word similarity task.

D. Word Analogies Evaluation

As the words “king”, “queen”, “man”, and “woman” appear in the book, we decided to test the famous “king-queen” word analogy [9]. This test assumes that if the word embeddings are good, the arithmetic operation of subtracting the word embedding of “man” from “king” and adding “woman” should result in an approximate value for the word embedding of “queen”. We examined the topmost similar vectors for the equation “king” - “man” + “woman”, but unfortunately, we could not find the word “queen” in any of the models. We attribute this failure to the very low frequency (Table V) of these words in the book, especially in the case of “man” and “woman”.

TABLE V
BOOK FREQUENCY OF THE WORDS IN THE “KING-QUEEN” ANALOGY

Word	Nr. of occurrences
king	63
queen	76
woman	4
man	1

E. Visualization

For visualization, we initially attempted to plot the embeddings of the top 30 most frequently used words in the book (Figure 1). While this provided some visual information regarding the word frequencies, such as “alice” and “said” understandably being the most common words, it did not provide any insights on how we could evaluate the quality of the embeddings.

For our second attempt, we modified our approach and decided to focus on selecting characters that had an adjective closely associated with their nouns (e.g., “The White Rabbit”) or characters that were closely connected within the book (e.g., “Two”, “Five”, and “Seven”). Additionally, we assigned the same color to all the adjectives and nouns or multiple characters that were related (Figure 2). Although the results were not perfect, as exemplified by “The Cheshire Cat” in the CBOW model, we can proudly say that most pairs or groups of three were relatively close to each other.

Due to the high dimensionality of the word embeddings, which exceeded 2 or 3 dimensions, we had to utilize the t-SNE technique [7] to visualize them.

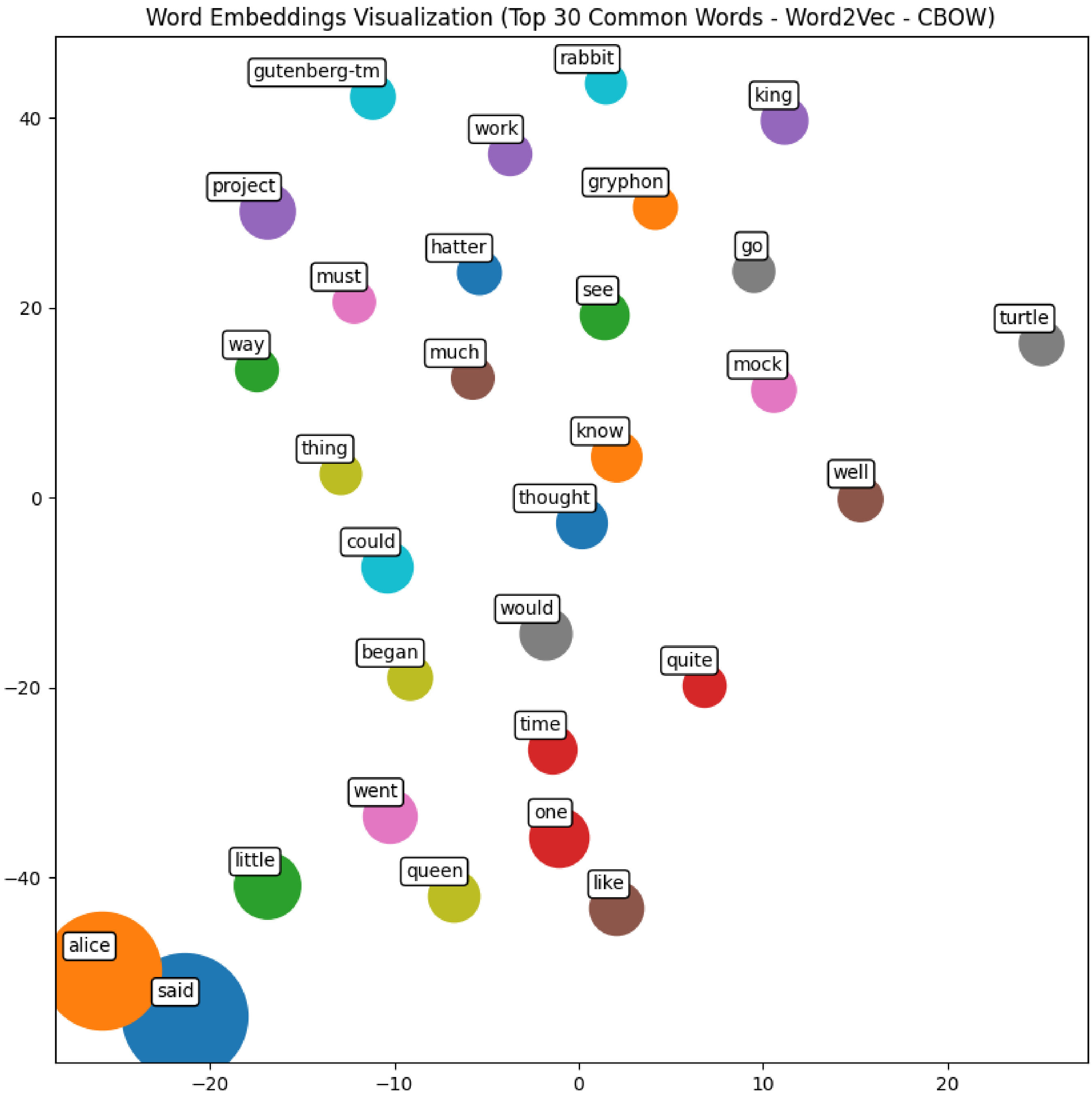

Fig. 1. Word embeddings visualization for the top 30 common words (CBOW).

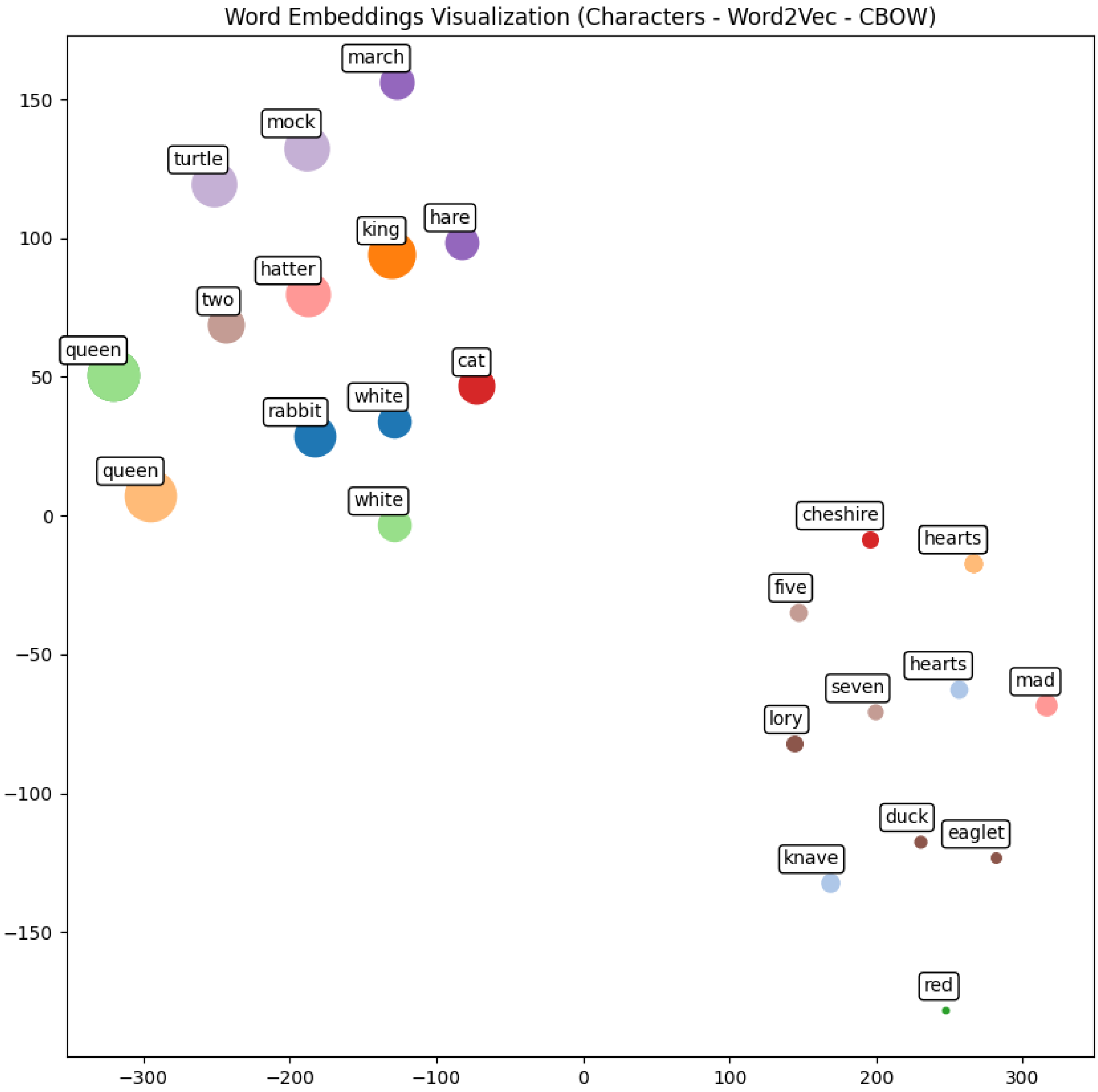

Fig. 2. Word embeddings visualization for the characters (CBOW).

V. Conclusion and Future Work

In conclusion, despite the challenges posed by the relatively small text corpus of "Alice's Adventures in Wonderland" our research and experiments in training multiple machine learning models have yielded several successes. We have achieved positive results in terms of searching for similarities, comparing word embeddings, and visualizing them.

Moving forward, there are several areas of exploration and improvement to consider. Firstly, exploring other models such as GloVe or fine-tuning the hyperparameters of our current models can potentially enhance the quality of the embeddings. Secondly, conducting similar experiments on other fantasy novels, preferably with larger text corpora, can provide valuable insights and broaden our understanding of language representation in this genre. Lastly, placing more emphasis on contextual embeddings rather than solely focusing on word embeddings can lead to more nuanced and comprehensive language understanding.

As mentioned in Section III, our aim is to inspire and motivate other researchers to delve deeper into the application of machine learning techniques on a wider range of fantasy novels. By doing so, we can gain a better understanding of the linguistic intricacies and creative aspects present in these literary works.

Through our efforts, we strive to encourage further exploration of NLP techniques as well as provide new perspectives on classical works of literature. The use of machine learning models to creative texts, especially ones from the fantasy genre, can uncover novel nuances in the representation of language.

References

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[2] Richard Brath. Surveying wonderland for many more literature visualization techniques. *arXiv preprint arXiv:2110.08584*, 2021.

[3] Lewis Carroll. *Alice's Adventures in Wonderland*. Macmillan, London, 1865.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Henrique Ferraz de Arruda, Filipi Nascimento Silva, Vanessa Queiroz Marinho, Diego Raphael Amancio, and Luciano da Fontoura Costa. Representation of texts as complex networks: a mesoscopic approach. *Journal of Complex Networks*, 6(1):125–144, 2018.

[6] Zellig Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[7] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[9] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[10] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[11] Xin Rong. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*, 2014.