# Word Disambiguation using Wikipedia and NLTK

Md Mobusshar Islam (2305578), Gökçe Eylül Ünlü (2308275), Mete Turhan (2308648)

*Project on GitHub:*
`https://github.com/Mobusshar/NLP_Final_Project`
`https://github.com/asajadi/wikisim`
`https://github.com/MeteTurhan/Lesk_algorithm_implementation_using_Wikipedia`

***Keywords*—Wikipedia, Lesk Algorithm Python, NLTK, Word Disambiguation, convex convolution**

## I. Summary:

This report aims to show a comprehensive methods of word disambiguation using Python NLTK (Natural Language Toolkit), Wikipedia, Senseval-2 database, general databases for comparison, word2vec, Glove and FastText. This project benefits from various tools and methods such as Lesk algorithm, Wikisim Framework, Senseval-2, MC, RG and WS353 datasets. Furthermore, this report aims to show a graphical user interface (GUI) to showcase the project's results.

## II. Introduction:

Under different circumstances, one word can have various meanings and in the day-and-age of information generation, making it essential to further develop better word disambiguation methods to make sense of queries. This report displays various detailed examinations of word disambiguation techniques through graphs created by mentioned methodologies.

This work does not only focus on word disambiguation but also factors in the human judgment element by inspecting the correlation between the mentioned models' results and human judgement's results.

## III.Methodology:

a. *Lesk Algorithm Implementation:*

The Lesk algorithm is a word sense disambiguation method used in natural language processing to determine the correct sense of a word within a given context. It relies on the idea that the meaning of a word can be clarified by comparing the context in which it appears with the definitions of its various senses from a lexical resource, such as a dictionary. The algorithm collects the words surrounding the target word and then computes the overlap between these words and the words in the definitions of each possible sense of the word. The sense with the highest overlap is chosen as the most likely sense for the context.

While the fundamental concept underpinning the Lesk algorithm remains consistent, there is a slight alteration in the implementation of the algorithm within the code. This change pertains to the utilization of a different lexical source, Wikipedia, as opposed to the traditional Lesk algorithm, which relied on WordNet.

b. *Wikipedia Query and Sense Overlapping:*

Let us consider a scenario where we have a designated target word, denoted as 'T,' within a given sentence, 'S.' In the process of disambiguating word 'T' in the context of sentence 'S,' the conventional approach involves comparing each word in sentence 'S' with the respective definition provided by WordNet.

However, this method often yields limited comparisons, given that WordNet definitions typically consist of concise sentences, around 5 words in length, assumed to be associated with the target word 'T' when applying the Lesk algorithm.

The advantage of the proposed approach lies in the fact that, instead of relying solely on single-sentence definitions from WordNet, we consider each target word, which corresponds to entities in Wikipedia, as pertaining to an entirely separate page. This results in the availability of a paragraph containing a multitude of words related to each entity. Consequently, there is a substantially larger pool of related words available for comparison with sentence 'S.' This expanded context provides a significantly greater likelihood of identifying a meaningful match to the correct sense, thereby reducing the chances of returning a null result. Given the substantially larger text available in Wikipedia compared to WordNet, it becomes prudent to explore strategies that can mitigate the computational cost associated with word sense disambiguation. One such strategy is the adoption of a window-based approach, which seeks to optimize the process. In this method, we determine the precise position of the target word within the text and establish a defined window size. Within this window, we exclusively consider the words that occur both before and after the target word, thereby classifying them as 'related words.' It is these 'related words' that we subject to a comparative analysis with the words present in the sentence 'S.' This approach not only streamlines the computational workload but also refines the context for a more precise disambiguation process.

Wikipedia features disambiguation pages for each target word, and it may be tempting to consider these pages as senses, akin to those in WordNet. However, this approach becomes less tenable due to the fact that some entities simply redirect the user to other links, rendering this approach illogical.

Prior to commencing the proposed methodology, an initial preprocessing stage is executed for both the sentence under consideration and the textual content extracted from Wikipedia. This preprocessing encompasses several essential procedures, including the removal of stopwords, lemmatization, stemming, and other pertinent operations. The underlying rationale for the execution of these processes resides in the objective of obtaining a more refined and sanitized text and sentence, given that the Lesk algorithm inherently relies on precise matches during the disambiguation process.

Two distinct instances are presented here, each featuring a different target word and corresponding sentence. In the first case, the sentence, "This plant requires sunlight and watering every morning," involves the target word "plant." In the second case, the sentence, "Bank is the financial prison of society," centers around the target word "bank." The results obtained from the disambiguation process in these particular instances exhibit logical and meaningful outcomes. The sense derived from executing the code on the target word 'plant' yields the following interpretation: 'Plants are the eukaryotes that form the kingdom Plantae; they are predominantly photosynthetic.' Conversely, when applying the code to the target word 'bank,' the sense obtained is as follows: 'A bank is a financial institution that accepts deposits from the public and creates a demand deposit while simultaneously making loans.

It is noteworthy that, in these cases, the traditional Lesk algorithm utilizing WordNet was also applied. Remarkably, in contrast to numerous other scenarios within the project where WordNet-based results proved to be inaccurate, the employment of Wikipedia as the lexical resource consistently yielded improved outcomes. This underscores the enhanced efficacy of the Wikipedia-based approach for word sense disambiguation in the specific contexts of these examples.

## c. *WikiSim Framework and Sense Disambiguation:*

Wikisim is a framework which provides Vector-Space Representation of Wikipedia Concepts, Semantic Relatedness between Wikipedia Concepts and Wikification: Entity Linking to Wikipedia[1]. Wikisim is a framework that uses Python 2.7 for a language, Anaconda (or Miniconda) to simplify package management and deployment to run. To make calculations, it uses functions which utilize Normalized Google Distance (ngd) similarity, wor2vec similarity, wlm (ngd) similarity, co-citation similarity, coupler similarity or custom similarity methods. User may choose one of these methods to complete their calculations.

➢ *Understanding of Senseval-2 Dataset:*

Senseval-2 dataset is a fairly small dataset with a structure of xml file. All the items, including but not limited to sentences, words, stopwords, are tagged with appropriate tags. It is crucial to transform this dataset to a vocabulary understandable by the Wikisim package since all the similarity modules from the package expect tuples.

| Lemma | Text | Tag | Attributes |
|---|---|---|---|
| the | The | DET | wf |
| art | art | NOUN | instance |
| english | English | NOUN | instance |
| like | like | ADP | wf |

*Table 1: Example of Senseval-2 dataset's tagging system*

The aim was to make Senseval-2 usable by Wikisim functions, therefore the sentences were processed using their <sentence> tag to find all lemmas, including the stop words which were to be removed later. After the removal of stopwords, lemmatization of remaining words and tokenization, they were ready to be used by the said functions.

While Senseval-2 is a fairly small database, for simplicity's sake, for each test only top similar 10 and least similar 10 words were selected to plot graphs.

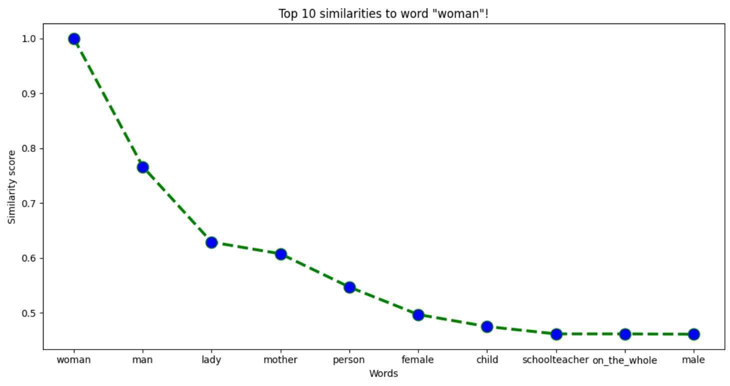


*Table 2: Top 10 most similar words to entry "woman" in Senseval-2 dataset*

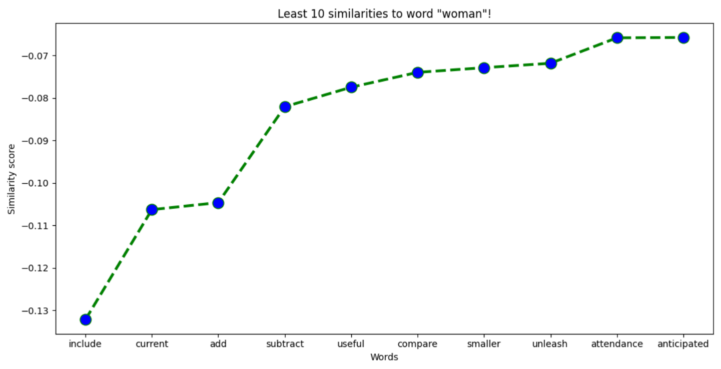
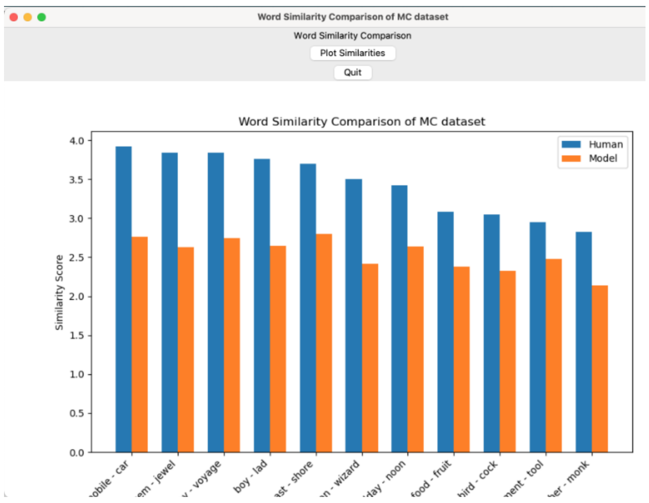


*Table 3: Top 10 least similar words to entry "woman" in Senseval-2 dataset*

## d. *Word Vector Evaluation:*

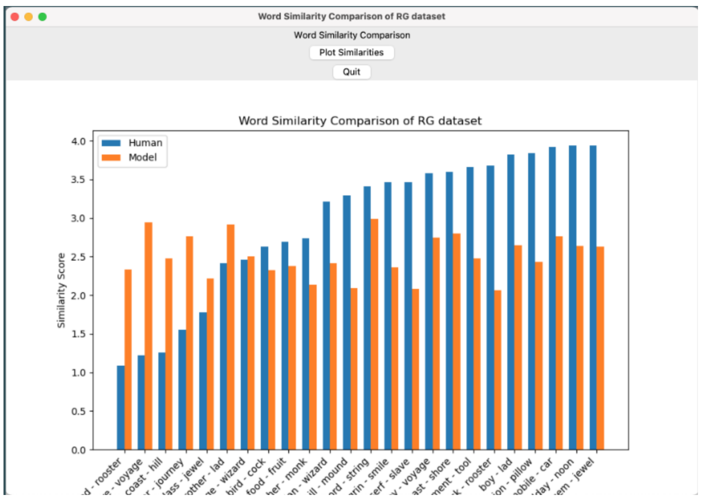
The study of linguistic word similarities has long intrigued researchers and linguists. Advanced computational tools, such as WikiSim, have facilitated the deep dive into these intricacies, allowing for a nuanced comparison of word relationships. Utilizing sophisticated algorithms like Word2Vec and FastText, WikiSim delves into vast textual corpora to determine how closely words are related based on their contextual usage. In the presented visualizations, three datasets—MC, RG, and WS353—serve as the basis for comparison. These datasets represent human perceptions of word similarities and provide a valuable reference point against which to measure the outputs of computational models.

1. MC Dataset Comparison (word2vec):



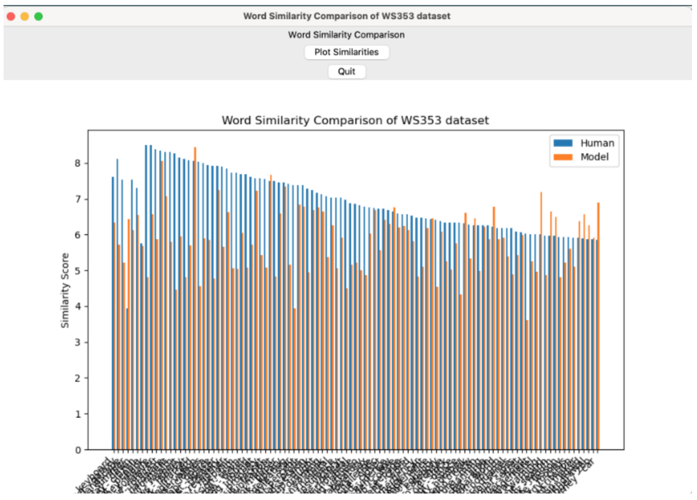
The image showcases the word similarity scores of the MC dataset. At a glance, the blue bars (representing human scores) and the orange bars (representing model scores) vary significantly for some word pairs, while for others, they align closely. For instance, words like 'mobile-car' and 'boy-lad' have almost identical scores between human and model assessments. However, there are words like 'pen-jewel' and 'boy-sex', where the model seems to deviate from human intuition. Such discrepancies can arise due to biases in the training data, insufficient representation, or the inherent limitations of the model in understanding nuanced human perceptions.

2. RG Dataset Comparison (word2vec):



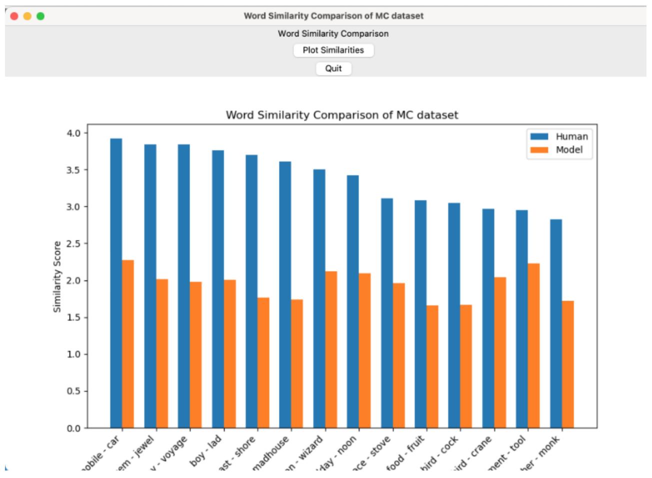
In the RG dataset visualization, the distribution of similarity scores appears tighter than in the MC dataset. Both human and model scores seem to be in closer agreement for many word pairs, with occasional variations. For words like 'car-journey' and 'lad-brother', both humans and the model perceived high similarity. However, there are evident discrepancies in pairs like 'cock-nail' and 'monk-slave', showcasing that even within this dataset, the model doesn't always align perfectly with human judgment.

3. WS353 Dataset Comparison (word2vec):



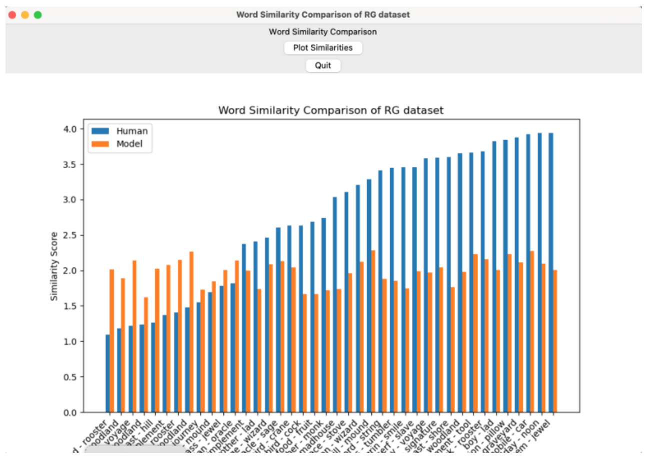
The WS353 dataset visualization presents a more complex picture due to the sheer number of word pairs. While the majority of the bars are closely packed, indicating a general agreement between human and model assessments, there are spikes where the model deviates from human scores. Given the dense presentation, it's essential to dig deeper into individual word pairs to fully grasp where the model excels and where it falters.

4. MC Dataset Comparison (fastext):



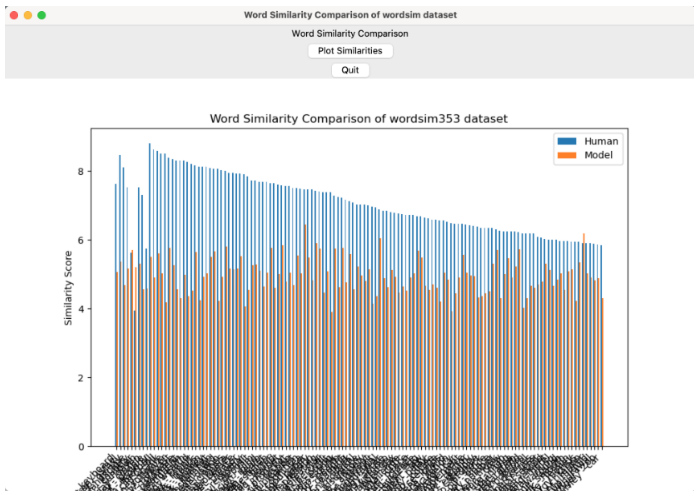
This chart looks at the MC dataset with the Fastext model. It compares words like "mobile-car" and "pencil-voyage" to see how related they are. Sometimes, the model's guess on how related two words are matches what people think. For example, most people think "boy" and "lad" mean pretty much the same thing. But, the model doesn't always agree and scores them differently.

5. RG Dataset Comparison (fastext):



This chart is about the RG dataset. Here, the model often thinks words are less similar than people do. For example, people might see a closer connection between "brother-monk" and "professor-cucumber" than the model does. This shows that humans think about word relationships in many ways, and the model might not catch all of them.

6. WSD353 Dataset Comparison (fastext):



The third chart, based on the WSD353 dataset, provides an in-depth look at a wide array of word pair relationships. It seeks to understand how machine interpretations of these pairs compare to human judgments. At times, the outcomes produced by the computational model harmoniously align with the perspectives of human evaluators, signaling the model's adeptness at grasping complex linguistic relationships. However, not all results are harmonious. In some instances, there's a clear discrepancy between machine and human evaluations. These differences highlight an ongoing challenge: computational models, despite their sophistication, still struggle to mimic the subtle cognitive processes of humans.

e. *Convex Combination of Vectors:*

In this part, we will create composite vectors by performing convex combinations. For these combinations, Wikisim generated vectors and FastText vectors will be used. The task is to evaluate both vectors' effects and try to achieve the optimal performance for various language processing tasks.

We will denote the variables:

| | |
|---|---|
| Combined Vector | Comb_V |
| Wikisim Vector | Wikis_V |
| FastText Vector | Fast_V |
| alpha | 0.3/0.5/0.7 |

*Table 3: variable generation for convex combination task*

Our combination formula follows:

*Comb_v = alpha * Wikis_V + (1-alpha) * Fast_v*

Using this formula, the same top most and least similar words to "woman" from the Senseval-2 dataset are generated.

While convex combination of factor (alpha) is 0.3:

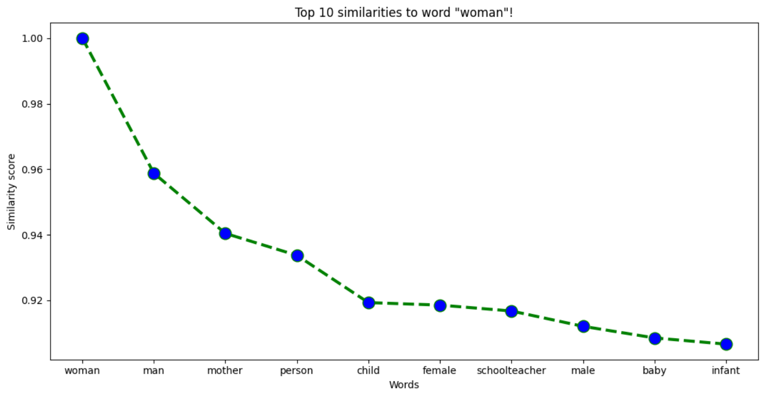


*Table 4: Top 10 most similar words to entry "woman" in Senseval-2 dataset with alpha 0.3*

With the 0.3 alpha value, we have similar results with the top 10 most similar words. However, if we look at the top 10 least similar words, majority of the similar words' similarity value is close to the top 10 most similar words' similarity value.

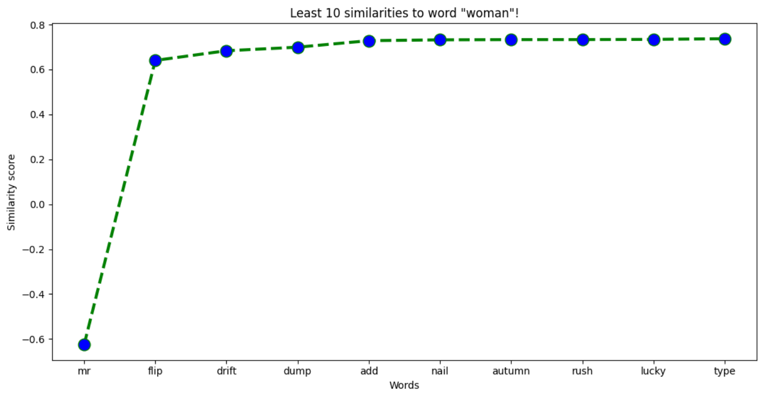


*Table 5: Top 10 least similar words to entry "woman" in Senseval-2 dataset with alpha 0.3*

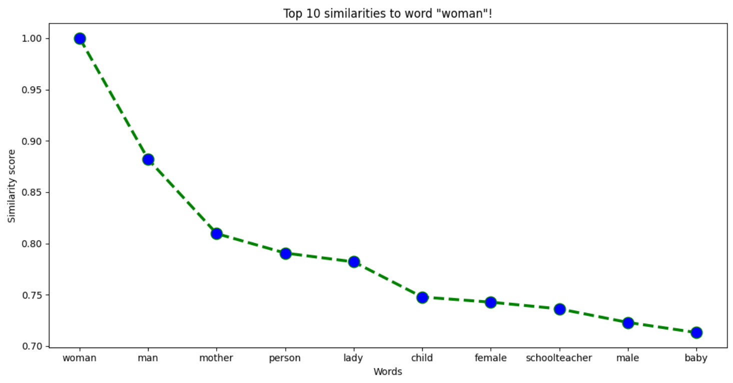


*Table 6: Top 10 most similar words to entry "woman" in Senseval-2 dataset with alpha 0.5*

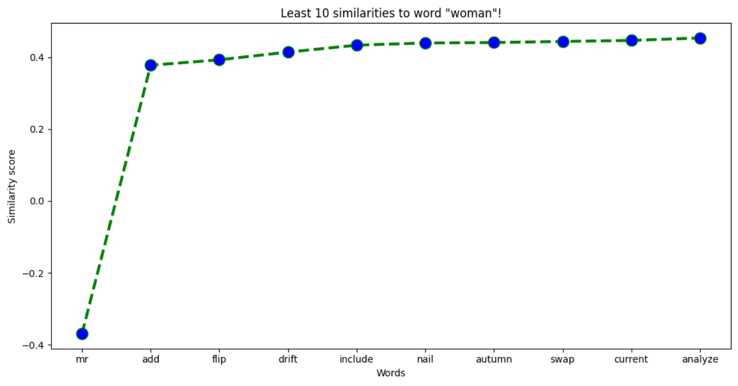


*Table 7: Top 10 least similar words to entry "woman" in Senseval-2 dataset with alpha 0.5*

With the 0.5 alpha value, we now have a better result for the least similar words since they are further apart from the most similar words.

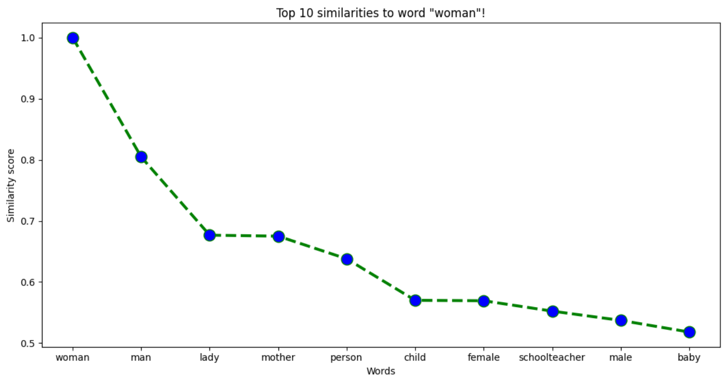


*Table 8: Top 10 most similar words to entry "woman" in Senseval-2 dataset with alpha 0.7*

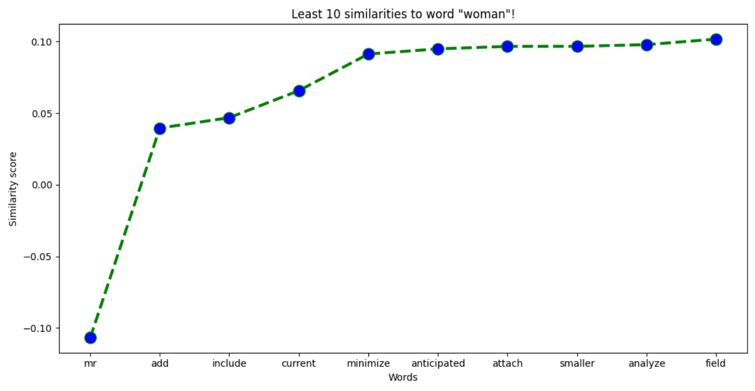
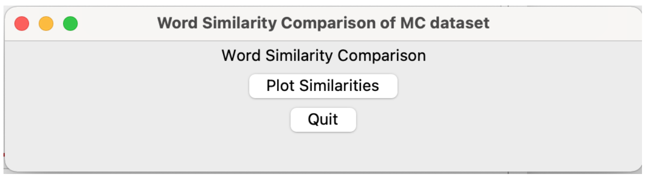


*Table 9: Top 10 least similar words to entry "woman" in Senseval-2 dataset with alpha 0.7*

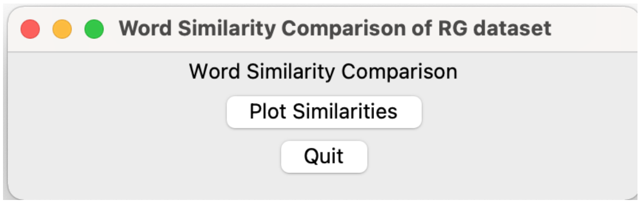
Now, comparing all the results with different alpha values, we can observe alpha = 0.7 has a better result for the least similar words. However, all the alpha values display consistent results for the most similar words. As the alpha value increases, the effect of our Wikisim vectors increases along with consistency. This may be due to the size of our Wikisim database used to calculate similarities. Therefore, we can observe that as the database a model trains on, its correctness increases.

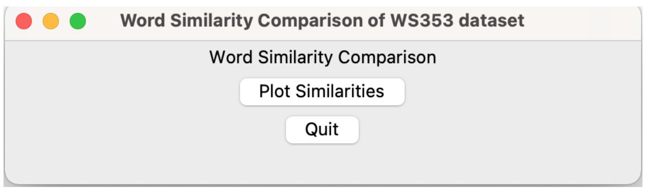
### *f. GUI Design:*

We designed a solution utilizing Tkinter, a well-known Python library, to create an easy-to-use interface. It features straightforward visual components, enhancing user interaction with our app. Specifically, the interface provides areas for users to input texts and a button to initiate the text comparison process, displaying the similarities and differences between the texts for user's ease.


Word Similarity Comparison of MC dataset
Word Similarity Comparison
Plot Similarities
Quit

Users simply interact with the buttons, clicking to begin the process. The application employs effective yet uncomplicated techniques to analyze the texts, examining their language and structure closely. It swiftly determines the similarity between the texts and displays this data on the screen for easy understanding and quick access by the users.


Word Similarity Comparison of RG dataset
Word Similarity Comparison
Plot Similarities
Quit


Word Similarity Comparison of WS353 dataset
Word Similarity Comparison
Plot Similarities
Quit

This blend of a straightforward interface and smart functionality makes it easy for users to interact with the application and get immediate, accurate results. The design of this application puts the user first, giving them a tool that is not just powerful, but also easy to use. Tkinter really shines in making applications that are functional, user-friendly, and responsive.

## IV. Results and Discussion:

The proposed Wikipedia-based Lesk algorithm offers several key benefits over the traditional WordNet-based algorithm for word sense disambiguation. First, it is able to achieve more accurate results in cases where the WordNet-based algorithm falls short. This is due to the fact that Wikipedia provides a much larger and more comprehensive context for each target word, in the form of paragraphs of related text. This allows the algorithm to make more meaningful comparisons and identify the correct sense of the word with greater confidence. Second, the Wikipedia-based Lesk algorithm is able to disambiguate words in a wider range of contexts. This is because Wikipedia contains articles on a wide variety of topics, from science and technology to history and culture. This makes it possible to disambiguate words in even the most specialized contexts, which may be difficult or impossible to do with WordNet. Third, the Wikipedia-based Lesk algorithm is less likely to return null results. This is because Wikipedia contains a much larger vocabulary than WordNet. This means that the algorithm is more likely to find a match for the target word, even if it is a rare or specialized word.

However, there are also some limitations to the Wikipedia-based Lesk algorithm. First, it requires a large and comprehensive Wikipedia corpus. This can be computationally expensive to create and maintain because of this reasoning the window-based approach is used in the project. Second, the algorithm may be less effective for words with few Wikipedia articles or disambiguation pages. This is because there will be less context available for the algorithm to make its comparisons. Finally, the computational complexity of the algorithm may increase as the size of the Wikipedia corpus grows. This is because the algorithm needs to compare the target word to all of the words in the corpus to find the best match. Overall, the Wikipedia-based Lesk algorithm is a promising approach to word sense disambiguation with the potential to improve the performance of natural language processing systems. However, it is important to be aware of the limitations of the algorithm.

## V. Conclusion:

In conclusion, the proposed Wikipedia-based Lesk algorithm has been demonstrated to be a promising approach for word sense disambiguation, particularly in cases where the traditional WordNet-based algorithm falls short. The expanded context provided by Wikipedia, in the form of paragraphs of text related to each target word, leads to more meaningful comparisons and accurate results. This is evident in the examples presented, where the Wikipedia-based algorithm outperforms the WordNet-based algorithm in both cases. While further research is required to evaluate the performance of the proposed approach on a larger dataset and in more diverse contexts, the initial results suggest that it has the potential to be a valuable tool for word sense disambiguation.

Several patterns emerge from these visualizations. Firstly, it's evident that no model, including word2vec or fastext, can capture human semantic understanding perfectly across all word pairs. While there are numerous instances of close alignment, discrepancies exist and are crucial for researchers and practitioners to be aware of. These gaps can arise from various factors, including the inherent limitations of the model, biases in training data, or the multifaceted nature of human language perception.

Secondly, the consistency in scores across datasets suggests that while word2vec and fastext might be robust in their representations, they're not infallible. The model seems to perform better with certain word pairs, possibly those that are more frequently represented or contextually clearer in its training data.

Lastly, the comparison underscores the importance of continuous validation. Embedding models, despite their power, need to be checked against human ground truth regularly to ensure their applicability and accuracy in real-world scenarios.

## VI. Acknowledgments:

Word embeddings, while transformative in natural language processing tasks, are not a silver bullet. As the comparisons across the MC, RG, and WS353 datasets highlight, there's a nuanced interplay between human semantic understanding and machine representation. As science advances, it's critical to keep developing these models, drawing inspiration from such comparative assessments, in order to get closer to the complicated texture of human language and cognition.

## VII. References:

1. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
2. Pennington, J., Socher, R., & 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).
3. Levy, O., & Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In Advances in neural information processing systems (pp. 2177-2185).
4. Baroni, M., Dinu, G., & Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 238-247).
5. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Paşca, M., & Soroa, A. (2009). A study on similarity and relatedness using distributional and WordNet-based approaches. In Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the Association for Computational Linguistics (pp. 19-27).
6. Rehurek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks (pp. 45-50).
7. Rong, X. (2014). word2vec parameter learning explained. arXiv preprint arXiv:1411.2738.
8. Gabrilovich, E., & Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In IJCAI (Vol. 7, pp. 1606-1611).
9. Milne, D., & Witten, I. H. (2008). Learning to link with Wikipedia. In Proceedings of the 17th ACM conference

on Information and knowledge management (pp. 509-518).
10. Zesch, T., Müller, C., & Gurevych, I. (2008). Using Wiktionary for computing semantic relatedness. In AAAI (Vol. 8, pp. 861-866).
11. Yeh, E., Ramage, D., Manning, C. D., Agirre, E., & Soroa, A. (2009). WikiWalk: random walks on Wikipedia for semantic relatedness. In Proceedings of the Workshop on Graph-based Methods for Natural Language Processing (pp. 41-49).
12. Ponzetto, S. P., & Strube, M. (2011). Knowledge derived from Wikipedia for computing semantic relatedness. Journal of Artificial Intelligence Research, 30, 181-212.
13. Mihalcea, R., & Csomai, A. (2007). Wikify!: linking documents to encyclopedic knowledge. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (pp. 233-242).
14. Radinsky, K., Agichtein, E., Gabrilovich, E., & Markovitch, S. (2011). A word at a time: computing word relatedness using temporal semantic analysis. In Proceedings of the 20th international conference on World wide web (pp. 337-346).
15. Speer, R., & Havasi, C. (2012). Representing general relational knowledge in ConceptNet 5. In LREC (pp. 3679-3686).
16. Hu, M., Sun, A., & Lim, E. P. (2012). Comments-oriented document summarization: Understanding documents with readers' feedback. In Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval (pp. 291-300).
17. Witten, I. H., & Milne, D. (2008). An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In Proceedings of the AAAI workshop on Wikipedia and artificial intelligence: An evolving synergy (pp. 25-30).
18. West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., & Lin, D. (2014). Knowledge base completion via search-based question answering. In Proceedings of the 23rd international conference on World wide web (pp. 515-526).
19. Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot: Distributed word representations for multilingual NLP. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning (pp. 183-192).