VCET TECHZETTE विसीईटी ज्ञानपत्र
ISSN: 2584-0886 (Online)

Lesk's Algorithm: A Knowledge Based Approach Word Sense Disambiguation

Mr. Harshkumar Devmurari

Introduction

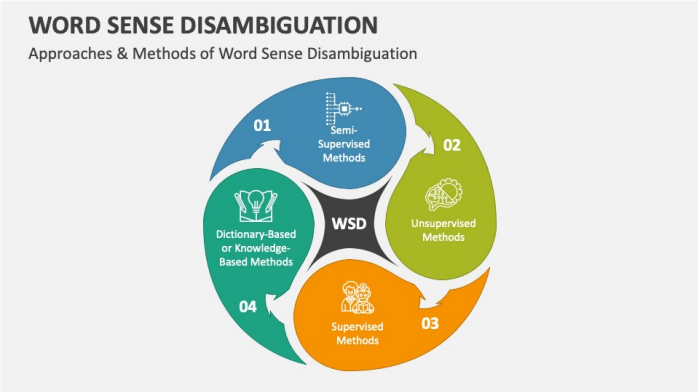
In the realm of the English language, over 38% of words exhibit polysemy, a phenomenon where a single word encompasses multiple distinct definitions or "senses" (Edmonds, 2006). A prime example is the term "set," which can be defined as a noun, verb, and adjective, boasting numerous diverse interpretations. This unique linguistic trait becomes strikingly evident when we encounter sentences like, "Kindly set the set of cutlery on the table." In this context, how does one intuitively discern the intended meaning of "set"? The key lies within the contextual backdrop. The human brain's intricate neural networks orchestrate semantic comprehension, storage, and retrieval, endowing us with an inherent proficiency in word-sense disambiguation (WSD). In essence, we possess an innate aptitude for unravelling the intended significance of words with multiple connotations, guided by the contextual milieu. This intrinsic capability has significantly influenced the evolution of natural language, shaping it to seamlessly incorporate these intricate webs of contextual associations.

Word-sense disambiguation in NLP

In the realm of Natural Language Processing (NLP), a facet of artificial intelligence dedicated to deciphering and comprehending human language, the intricate dance of words wielding multiple meanings presents a formidable challenge. While our cognitive faculties navigate this linguistic issues adeptly, computers struggle with the same task. The significance of a word's connotation hinges on its surroundings, but for machines, capturing the encompassing context remains a formidable feat. Metaphors, modifiers, negations, and the myriad subtleties of

ISSN: 2584-0886 (Online)

language intricately interweave, confounding machine learning.



The pragmatic applications of NLP underscore the criticality of disentangling word senses, propelling the emergence of diverse disambiguation techniques. Among these, contemporary machine learning methodologies encompass supervised tactics. Here, algorithms learn from a trove of manually annotated words, each tagged with its specific sense, expediting classification. Nevertheless, this route is beset with challenges: assembling these tagged datasets proves expensive, time-intensive, and inherently imperfect, given that even human annotators concur on word senses only 70-95% of the time (Edmonds, 2006). Counterbalancing this effort, unsupervised approaches come into play, encompassing methods that endeavour to cluster words grounded in shared contextual traits.

Lesk's Algorithm: A simple method for word-sense disambiguation

Perhaps one of the earliest and still most commonly used methods for word-sense disambiguation today is Lesk's Algorithm, proposed by Michael E. Lesk in 1986. Lesk's algorithm is based on the idea that words that appear together in text are related somehow, and that the relationship and corresponding context of the words can be extracted through the definitions of the words of interest as well as the other words used around it. Developed long before the advent of modern technology, Lesk's algorithm aims to disambiguate the meaning of words of interest — usually appearing within a short phrase or sentence — by finding the pair of matching dictionary "senses" (i.e. synonyms) with the greatest word overlap in dictionary definitions. The Lesk algorithm is an knowledge based approach to word sense disambiguation. It works by comparing the context of the target word to the contexts of its

ISSN: 2584-0886 (Online)

different senses in a dictionary. The sense with the most similar context is the most likely correct sense.

In its most basic essence, Lesk's algorithm operates by tallying the common features between the dictionary definitions of a target word and those of the neighboring words within a designated "context window." This window encompasses the surrounding terms which would be considered for respective words. The algorithm then selects the definition associated with the word that displays the highest count of overlaps, while excluding common stop words like "the," "a," and "and." This chosen definition is posited as the inferred "sense" of the word.

STEPS:

1. **Select Word**: Choose the word for which you want to determine the correct meaning. Let's call this the "target word."

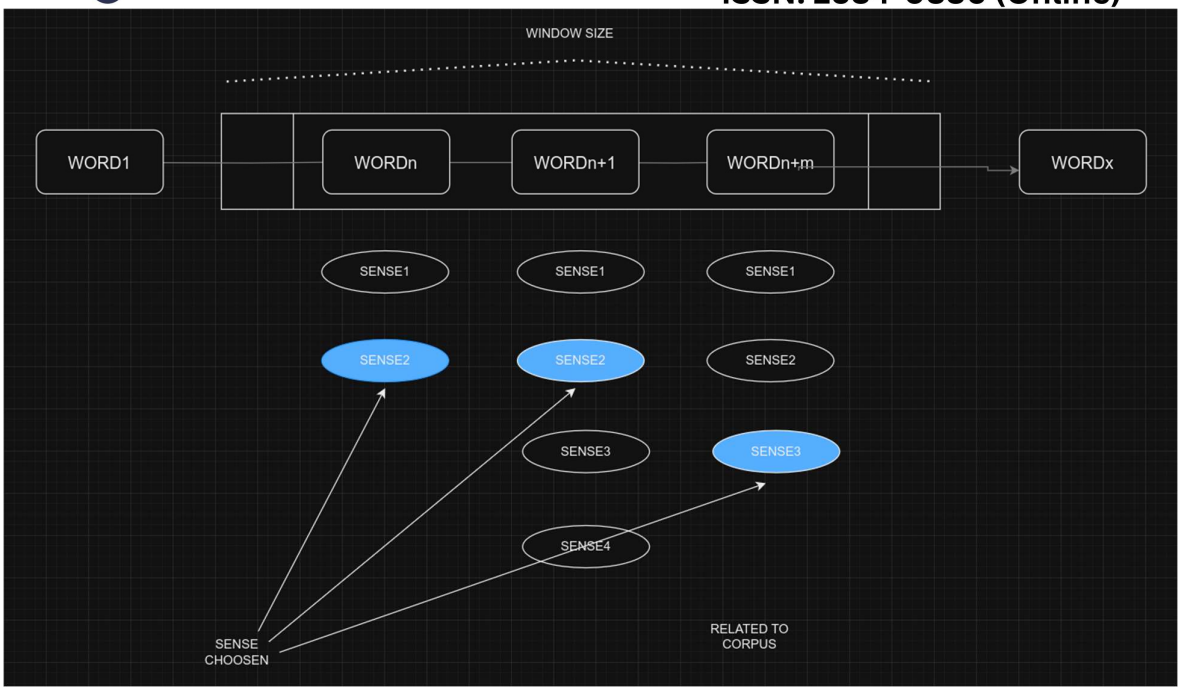
2. **Define Context Window**: Identify a group of nearby words around the target word. This collection of words is known as the "context window."

3. **Compare Definitions**: Compare the dictionary definitions of the target word with the definitions of the words within the context window.

4. **Count Overlaps**: Count the number of words that appear in both the target word's definitions and the definitions of the words in the context window. Exclude common words like "the," "a," and "and."

5. **Choose Most Overlapping Definition**: Select the definition of the target word that has the highest count of overlapping words from step 4. This chosen definition represents the most likely meaning of the target word in its current context.

VCET TECHZETTE विसीईटी ज्ञानपत्र
ISSN: 2584-0886 (Online)



Advantages and Disadvantages

Advantages

1) **Simple to implement:** Lesk's algorithm is a relatively simple algorithm that can be implemented in a few lines of code. This makes it a good choice for beginners and for applications where speed is important.
2) **Applicable in a variety of different contexts**: Lesk's algorithm does not rely on global information or the structure of the sentence. This means that it can be applied to new words and new contexts without having to be extensively modified. For example, the algorithm can be used to disambiguate the word "bank" in the sentences "The bank robber ran down the street" and "The bank of the river is very steep."
3) **Easily generalizable**: Lesk's algorithm is easily generalizable to new words and new contexts. This is because the algorithm does not rely on global information or the structure of the sentence.

ISSN: 2584-0886 (Online)

4) **Non-syntactic**: Lesk's algorithm is non-syntactic, which means that it does not rely on the arrangement of words in a sentence. This makes it more robust to errors in grammar and punctuation.
5) **Accuracy level**: Lesk's algorithm has been shown to be accurate in most cases, especially for words with clearly defined sentence with different sense of similar words. However, the algorithm can be less accurate for words with multiple senses that are semantically similar.

Disadvantages

1) **Low accuracy**: Lesk's algorithm has been shown to have low accuracy, especially for words with multiple senses that are semantically similar. For example, the word "bank" can have multiple senses, such as a financial institution, a riverbank, or a slope.
2) **Low recall**: Lesk's algorithm also has low recall, meaning that it cannot disambiguate many words. This is because the algorithm does not consider the context of the word in the sentence.

Overall, Lesk's algorithm is a simple, effective, and versatile algorithm for word sense disambiguation. It is a good choice for a variety of applications, especially for those where speed and accuracy are important.

Overall, Lesk's algorithm is a simple and effective algorithm for word sense disambiguation. However, it has some limitations, such as its low accuracy and low recall. These limitations can be addressed by using a better dictionary, weighting the matched terms, and adjusting the context window size as mentioned below

Unanswered questions

Lesk's algorithm leaves several questions unanswered, such as:

1) What dictionary should be used? The Lesk algorithm does not specify which dictionary should be used. This can affect the accuracy of the algorithm, as different dictionaries may have different definitions for the same word.

ISSN: 2584-0886 (Online)

2) Should all matched terms be considered equally, or should they be weighed by the length of the dictionary definition? The Lesk algorithm does not specify how to weight the matched terms. This can also affect the accuracy of the algorithm, as some terms may be more important than others.
3) How wide should the context window be? The Lesk algorithm does not specify how wide the context window should be. This can also affect the accuracy of the algorithm, as a wider context window may capture more information about the word, but it may also include irrelevant information.

These types of questions help us discover the best fit for our usecase and improve algorithm as per our need,

One way to improve the accuracy of Lesk's algorithm is to use a better dictionary. A better dictionary would have more accurate and complete definitions for words.

Another way to improve the accuracy of is to weight the matched terms. Some terms may be more important than others, and weighting the terms can help to improve the accuracy of the algorithm.

Another way to improve is to adjust the context window size. A wider context window may capture more information about the word, but it may also include irrelevant information. Adjusting the context window size can help to improve the accuracy of the algorithm by capturing the relevant information while filtering out the irrelevant information.

Other approaches -

Supervised approaches require a labeled corpus, which is a corpus of text where each word has been manually assigned a sense. The algorithm is trained on this corpus and then used to disambiguate words in new text.

Some supervised approaches to WSD include:

Naive Bayes: This is a simple but effective algorithm that uses Bayes' theorem to calculate the probability of a word having a particular sense given the context of the word.

ISSN: 2584-0886 (Online)

Decision list: This is a rule-based algorithm that uses a set of rules to disambiguate words. The rules are typically learned from a labeled corpus.

Semi-supervised approaches use a combination of labeled and unlabeled data. The algorithm is trained on the labeled data and then used to disambiguate words in the unlabeled data.

Semi-supervised approaches to WSD include:

Yarowsky's algorithm: This algorithm uses a bootstrapping technique to create a labeled corpus from an unlabeled corpus. The algorithm starts with a small set of labeled data and then uses this data to automatically label more data.

Unsupervised approaches to WSD include:

Hyperlex: This algorithm uses a hypergraph to represent the relationships between words and senses. The algorithm then uses a graph search algorithm to disambiguate words.

Conclusion

In conclusion, Lesk's Algorithm stands as a testament to the enduring power of innovative ideas in shaping our understanding of language. As we navigate the intricate landscape of Natural Language Processing, Lesk's Algorithm reminds us that even in the digital age, the wisdom of context, the resonance of shared meanings, and the delicate interplay of words continue to be pivotal. While newer techniques have emerged, this algorithm's simplicity and effectiveness showcase the elegance of linguistic investigation. As technology advances and our grasp of language deepens, the journey of refining algorithms like Lesk's will undoubtedly contribute to ever more precise and insightful language understanding, enriching both human communication and the capabilities of artificial intelligence.

Each of these approaches has its own advantages and disadvantages. Supervised approaches are typically more accurate than unsupervised approaches, but they require a labeled corpus, which can be expensive and time-consuming to create. Semi-supervised approaches can be a good compromise between accuracy and efficiency. Unsupervised approaches are the most efficient, but they are also the least accurate.

ISSN: 2584-0886 (Online)

The best approach to WSD depends on the specific application. For example, if accuracy is critical, then a supervised approach is likely the best choice. If speed is critical, then an unsupervised approach may be a better choice.