Integrating Lesk Algorithm with Cosine Semantic Similarity to Resolve Polysemy for Setswana Language

Tebatso Gorgina Moape, Oludayo O. Olugbara, Sunday O. Ojo
Dept. of Information Technology, Durban University of Technology, Durban, South Africa

***Abstract*—Word Sense Disambiguation (WSD) serves as an intermediate task for enhancing text understanding in Natural Language Processing (NLP) applications, including machine translation, information retrieval, and text summarization. Its role is to enhance the effectiveness and efficiency of these applications by ensuring the accurate selection of the appropriate sense for polysemous words in diverse contexts. This task is recognized as an AI-complete problem, indicating its longstanding complexity since the 1950s. One of the earliest proposed solutions to address polysemy in NLP is the Lesk algorithm, which has seen various adaptations by researchers for different languages over the years. This study proposes a simplified, Lesk-based algorithm to resolve polysemy for Setswana. Instead of combinatorial comparisons among candidate senses that Lesk is based on that cause computational complexity, this study models word sense glosses using Bidirectional Encoder Representations from Transformers (BERT) and Cosine similarity measure, which have been proven to achieve optimal performance in WSD. The proposed algorithm was evaluated on Setswana and obtained an accuracy of 86.66 and an error rate of 14.34, surpassing the accuracy of other Lesk-based algorithms for other languages.**

Keywords*—*Word sense disambiguation; Lesk algorithm; cosine similarity; Bidirectional Encoder Representations from Transformers (BERT)

I. Introduction

Setswana sometimes referred to as Tswana, is an African language spoken in South Africa, Botswana, and parts of Namibia. It is the national language of Botswana and one of the eleven official languages in South Africa. Setswana has about a total of 8.2 million speakers across the different countries. Similar to other natural languages, Setswana contains words with ambiguity, known as polysemous words that have multiple senses in various contexts in which they appear. Consider the following context sentences below:

1) C1: Noka ya mme e botlhoko (mother’s waist is painful).

2) C2: Moapeyi o noka nama letswai (The chef is seasoning the meat with salt).

3) C3: Setlhare se mothokoga ga noka (The tree is next to the river).

The Setswana word “noka” has three distinct senses based on the context in which it appears. It means waist in the first context C1, represents season (pour) in the second context C2, and a river in the third context C3. This linguistic characteristic is referred to as polysemy. Polysemy poses challenges in natural language processing (NLP) applications such as machine translation [1], information retrieval [2], and text summarization [3], where accurately determining the intended meaning of a polysemous word based on context is important. Resolving polysemy is crucial for improving the accuracy and precision of these applications. The dedicated task of resolving polysemy in NLP is known as Word Sense Disambiguation (WSD) and is considered one of the most difficult tasks in artificial intelligence [4].

WSD can be resolved using three approaches, namely, knowledge-based, unsupervised, and supervised methods. Knowledge-based methods use various lexical resources such as WordNet, Wikipedia, and BabelNet for disambiguation. Examples of this approach include the Lesk algorithm [5] and its variations, such as the simplified Lesk [6] and adapted Lesk [7] algorithms, which rely on context-gloss overlap.

Unsupervised methods employ clustering techniques for disambiguation from unannotated collection texts. The commonly used techniques are sense clustering algorithms such as K-Means [8] and graph-based algorithms such as PageRank [9]. Supervised methods rely on the availability of annotated datasets where a classifier is trained based on the annotations and features extracted from the data. Techniques used for this method include the use of support vector machine (SVM) [10] and Naïve Bayesian [11].

Unsupervised and supervised methods require the substantial collection of unannotated and annotated data, which is not available for resource-scarce languages such as Setswana. The scarcity of datasets poses a significant challenge for training robust models for NLP tasks. Due to this constraint, a knowledge-based approach was adopted to resolve WSD for Setswana using the simplified Lesk algorithm.

This paper makes several significant contributions to the field of WSD for low-resource languages, specifically focusing on Setswana. Firstly, it proposes a novel simplified Lesk-based algorithm that effectively resolves polysemy in Setswana by leveraging sentence embeddings generated using the PuoBERTa language model and employing the Cosine similarity measure to determine the most appropriate sense. Secondly, the study addresses the computational complexity inherent in traditional Lesk algorithms by encoding the context sentence and candidate glosses in a single operation, resulting in a linear growth rate of comparisons, thus mitigating the exponential growth of computational complexity. Lastly, this research provides a valuable WSD evaluation dataset for Setswana, which is currently unavailable, creating an essential benchmark for testing and comparing the performance of future Setswana WSD models.

This paper is structured as follows: Section II explores related works, materials, and methods presented in Section III. The evaluation of the proposed algorithm is provided in Section IV, including the obtained results. Discussion and conclusion is given in Section V and finally Section VI paves the way for future work.

II. Related Works

One of the first algorithms developed for WSD is Lesk's original algorithm. The original Lesk algorithm operates on the assumption that the sense of a word in a particular context can be inferred by examining the words that co-occur in the surrounding text. This algorithm relies on the availability of lexical semantic resources such as machine-readable dictionaries and wordnets to obtain glosses for each sense of the polysemous word. To disambiguate, the algorithm calculates the overlap between the words in the context and the words in the definitions to determine the appropriate sense. One of the major drawbacks of the Lesk algorithm is its computational complexity, which stems from the exponential growth of comparisons needed for numerous candidate senses associated with polysemous words across various lexical resources [12].

To overcome this limitation, researchers have proposed and developed variations of the algorithm, such as the simplified [6] and adapted [7] Lesk to improve the algorithm’s effectiveness. The simplified Lesk addresses combinatorial explosion by calculating overlaps between the definitions of candidate senses for the target word and the context words. The adapted Lesk expands the scope by considering not only the overlap between the context and target word senses but also introducing additional linguistic features or syntactic structures for a more comprehensive analysis.

Several researchers have adapted the core overlap idea and integrated various techniques into the original Lesk algorithm to enhance its performance. The study in [13] integrated topic modeling to simplify Lesk as the topic-document relationship between senses to determine the correct sense of the target word. Their model achieved an F1 score of 66%. The study in [12] used the adapted Lesk and trained a classifier responsible for retrieving senses of the target word from Wordnet, assigning a score based on the number of words common between the target gloss and context word gloss.

Tripathi, et al. [14] used Lesk to disambiguate Hindi words. The appropriate sense of the polysemous word is determined through a scoring method that assigns a sense score to each token of the Hindi sentence. The sense score is calculated based on the gloss, hypernym, hyponym and synonym of the combinations of different sense of tokens. The sense with the highest score in the combination is allocated as the most suitable sense within that context. In another study [15] used the LESK algorithm for Indonesian and achieved an accuracy of 78.6% for one Indonesian ambiguous word and 62.5% for two ambiguous words.

To disambiguate senses for Marathi languages, The study in study [16] used a modified Lesk algorithm coupled with a dynamic context window approach. After pre-processing, the algorithm stores the context words in an array to increase the context window size during processing while comparing the results with the static size context window output. The model achieved the highest precision of 0.79. Arabic, Kaddoura and Nassar [17] developed a WSD dataset that contains one hundred Arabic polysemous words with a minimum of three and maximum of eight senses. The paper adapts the idea of the Lesk algorithm [5] to compute the overlap between contextual information and dictionary definitions. They utilized a similarity measure to determine the appropriate sense of the target word. Their proposed method integrates Bidirectional Encoder Representations from Transformers (BERT) and introduces new features to enhance the effectiveness of disambiguation. The WSD system's performance achieved an impressive F1 score of 96%.

In this paper, a WSD system for Setswana is proposed based on the simplified Lesk algorithm. To tackle computational complexity, our algorithm first encodes the context sentence containing the polysemous word and the candidate glosses obtained from the Universal Knowledge Core (UKC) in a single operation. Secondly, the algorithm computes the Cosine semantic similarity measure between the context and each candidate gloss. As a result, the number of comparisons is equal to the *n* number of senses of a polysemous word. In contrast to other adaptations of the Lesk algorithm, the proposed algorithm exhibits a linear growth rate rather than an exponential one, mitigating computational complexity.

III. Materials and Methods

A. Task Definition

In NLP, Navigli [4] formally describes the WSD problem as: given a text T as a sequence of words (w_1, w_2, . . ., w_n), WSD is the task of assigning appropriate sense of a polysemous word in T, that is, to identify a mapping A from words to senses, such that $A(i) \subseteq SensesD(w_i)$, where $SensesD(w_i)$ is the set of senses encoded in a knowledge source K for word wi^1 and $A(i)$ is the subset of the senses of w_i which are appropriate in the context T. The mapping A can assign more than one sense to each word $w_i \in T$. However, only the most appropriate sense is selected, that is, $| A(i) |= 1$.” A knowledge source can be in various lexical formats. In this study, the UKC is used as the knowledge source.

B. Materials

This section outlines the material used for the development of the proposed algorithm. These are divided into three major components: the sense inventory, the WSD technique, and the disambiguation algorithm. Together, these elements form the architecture depicted in Fig. 1.

At the center of the architecture, the WSD algorithm takes the context sentence as input and produces the most appropriate sense for the polysemous word within that specific context. In the subsequent sub-section, the study incorporates the UKC as the sense inventory to retrieve glosses for the polysemous word. Following the UKC is the WSD technique that employs PuoBERTa for encoding and generating embeddings in Setswana, utilizing a Cosine semantic similarity measure to determine the correct sense.

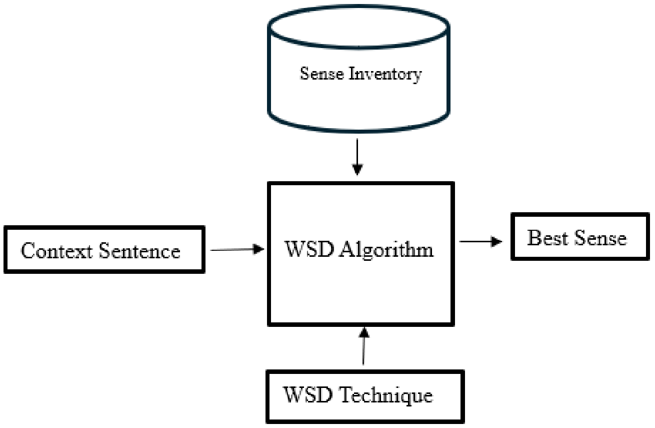
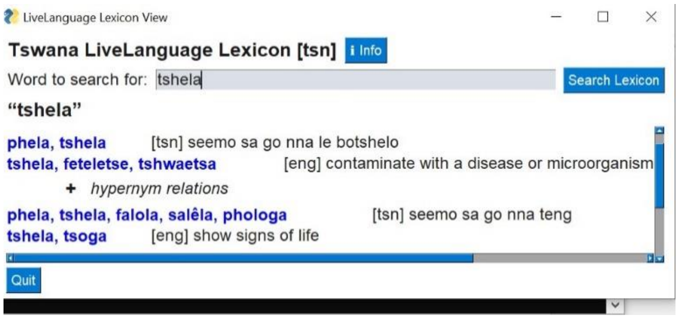


Fig. 1. WSD architecture.

1) The universal knowledge core: The Universal Knowledge Core (UKC) is a multilingual, high quality, large scale, and diversity-aware machine-readable lexical resource that currently consists of the lexicons of over a thousand languages, represented as wordnet structures [18], including Setswana. The UKC has two core layers, the concept and the language core [19]. The concept core is the knowledge layer of the UKC that provides a conceptual representation of concepts as we see them in the world. The concepts are interconnected through semantic relations and are language-independent. The language core is the language layer that is dedicated to lexical relations between lexical units (words). This layer encompasses multiple languages. The UKC provides an XML lexicon viewer, depicted in Fig. 2 that allows individuals to query and search for words, discovering their meanings and relationships.



Tswana LiveLanguage Lexicon [tsn]

Word to search for: tshela

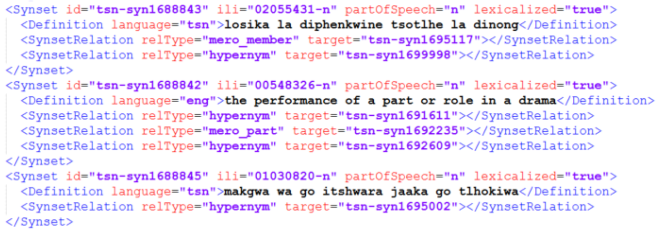
"tshela"

phela, tshela [tsn] seemo sa go nna le botshelo
tshela, feteletse, tshwaetsa [eng] contaminate with a disease or microorganism
+ *hypernym relations*
phela, tshela, falola, salêla, phologa [tsn] seemo sa go nna teng
tshela, tsoga [eng] show signs of life

Fig. 2. Setswana UKC XML lexicon viewer.

Fig. 2 illustrates the Setswana lexicon in the UKC. The word "tshela" is a polysemous word that has two distinct senses. The senses are "to live" and "to pour". The UKC provides glosses for each sense, along with English translations and available relations. For implementation of the proposed algorithm, we used the XML version of the lexicon in Python, Anaconda prompt. The snippet of the XML is depicted in Fig. 3.

In the XML, each synset has a synset ID that links it to the Princeton Wordnet, a part of speech tag, gloss and synset relation. The algorithm proposed in this paper extract Setswana glosses for disambiguation from the XML file through unique synset IDs and leverages part-of-speech tags and synset relations to navigate the hierarchical structure of the XML data, pinpointing relevant glosses associated with each synset.



```
<Synset id="tsn-syn1688843" ili="02055431-n" partOfSpeech="n" lexicalized="true">
  <Definition language="tsn">losika la diphenkwine tsotlhe la dinong</Definition>
  <SynsetRelation relType="mero_member" target="tsn-syn1695117"></SynsetRelation>
  <SynsetRelation relType="hypernym" target="tsn-syn1699998"></SynsetRelation>
</Synset>
<Synset id="tsn-syn1688842" ili="00548326-n" partOfSpeech="n" lexicalized="true">
  <Definition language="eng">the performance of a part or role in a drama</Definition>
  <SynsetRelation relType="hypernym" target="tsn-syn1691611"></SynsetRelation>
  <SynsetRelation relType="mero_part" target="tsn-syn1692235"></SynsetRelation>
  <SynsetRelation relType="hypernym" target="tsn-syn1692609"></SynsetRelation>
</Synset>
<Synset id="tsn-syn1688845" ili="01030820-n" partOfSpeech="n" lexicalized="true">
  <Definition language="tsn">makgwa wa go itshwara jaaka go tlhokiwa</Definition>
  <SynsetRelation relType="hypernym" target="tsn-syn1695002"></SynsetRelation>
</Synset>
```

Fig. 3. Setswana UKC XML lexicon.

2) PuoBERTa (Bidirectional Encoder Representations from Transformers): Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based language representation model introduced to the NLP landscape by [20] developed to capture the context and bidirectional dependencies of words in a sentence. This model is based on the transformer architecture illustrated in Fig. 4.

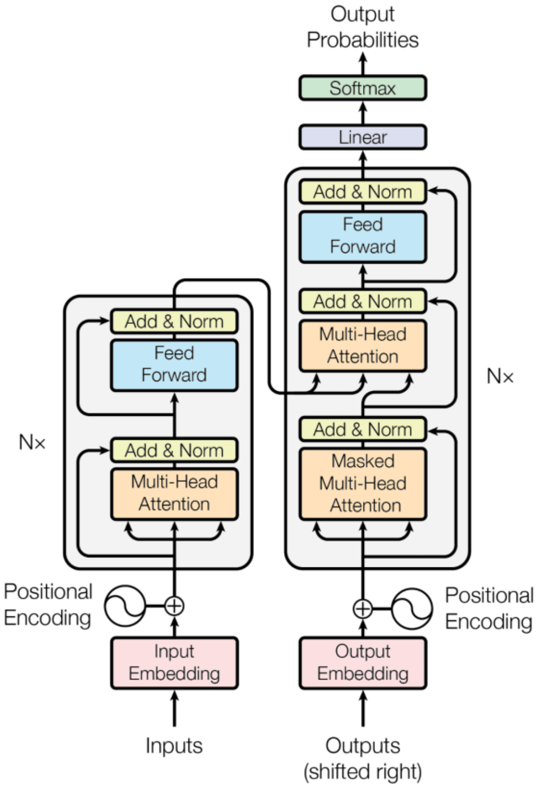


Fig. 4. Transformer architecture.

The transformer architecture employs a self-attention mechanism to capture dependencies between input tokens. The architecture consists of encoder and decoder stacks, each comprising multiple layers with multi-head attention, feedforward neural networks, and layer normalization. Positional encodings are incorporated to provide information about token positions. The model's success lies in its ability to efficiently handle long-range dependencies, parallelize computations, and serve as the foundation for various state-of-the-art models like BERT and GPT. Compared to traditional language models that process text sequentially from left to right or right to left, BERT processes the entire input sequence at once, taking into account both the preceding and following words for each word in a sequence [20]. This bidirectional approach allows BERT to capture more nuanced relationships and context in a language making it superior in tasks such as WSD. In addition, BERT can explicitly model the relationship of a pair of texts, which has proven to be beneficial for various pair-wise natural language understanding tasks [21]. Researchers have adapted the transformer architecture of BERT and trained language-specific models such as Arabic BERT [22] and Indict-BERT [23], to address distinct linguistic differences and used across various NLP tasks. For Setswana, PuoBERTa was developed and trained on Setswana data. It is a Setswana version of BERT trained by Marivate, et al. [24] using a corpus in the Setswana language. The PuoBERTa was trained on 24,295,328 Setswana tokens and evaluated on part-of-speech tagging and named entity recognition tasks. For Setswana WSD, PuoBERTa was used as an encoder to process and encode both contextual information, and the glosses of polysemous words extracted from the UKC.

3) Disambiguation Algorithm: This paper adopted the simplified Lesk algorithm for Setswana disambiguation. The reason for this selection is that compared to other variants, Simplified Lesk's primary objective is to maintain disambiguation effectiveness while simplifying the computation by reducing computational complexity [6]. This algorithm reduces computational complexity by using limited context, instead of considering the entire context surrounding the polysemous word. Simplified Lesk limits the scope to a predefined window of adjacent words and has the ability to work well for languages with limited resources, making it suitable for Setswana as a resource-scarce language. Another method that Simplified Lesk adopts to decrease computational complexity involves minimizing linguistic features. This is done by reducing reliance on extensive linguistic features and syntactic structures within the context, simplifying the feature set. However, this is a crucial capability when disambiguating agglutinative and morphologically rich languages such as Setswana. To address this, the linguistic features are integrated into the encoding process, utilizing PuoBERTa to encode the complete context sentence and the respective glosses to generate sentence embeddings. Sentence embeddings capture contextual information and the compositional nature of a language, this provides a representation that reflects the combination and interaction of words within a sentence [25]. Our algorithm is more closely related to the Simplified Lesk algorithm [6] but leverages on the importance of word sense definitions using embeddings and similarity measure. The metric used to measure semantic similarity is the Cosine similarity as it effectively captures the directional similarity between vectors, making it suitable for assessing the relationship between gloss embeddings and context representations [26]. Research that employs Cosine similarity for disambiguation includes [27], [28], [29], [30].

The algorithm consists of the following steps:

Input: Ambiguous word (W), Context sentence (C), Sense inventory with glosses for each sense of W.

Disambiguation:

1) *Tokenization:* Tokenize the context sentence (C)

2) *Encode:* Encode context sentence (C) using PuoBERTa

3) *Sense selection:* For each sense of the ambiguous word (W), retrieve the corresponding glosses

4) *Encode and measure similarity:* Encode corresponding glosses (G) using PuoBERTa Measure semantic similarity using Cosine similarity on Eq. (1) between C and G

$$\cos(\theta) = \frac{\mathrm{A}\cdot\mathrm{B}}{\|\mathrm{A}\|\|\mathrm{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

5) *Sense ranking:* Rank the senses based on the degree of similarity. The sense with the highest similarity is considered the most likely appropriate sense.

6) *Disambiguation:* Assign the sense with the highest similarity score as the disambiguated sense for the ambiguous word (W).

The steps above are formalized as the Algorithm 1 below:

Algorithm 1: Setswana WSD

```
Input:  target word (w), context sentence (cs)
Output: best disambiguate senses (bestDef) of the w
STX ← Pre-process(RemoveStopwords(Tokenized(Text))
Syn = GetSynsetFromUKC(w)
Est = defEncode(STX)
highestSim = 0
    For s in Syn
             Sdef = GetDefinition(s)
             Eqs = defEncode(Sdef)
                 If Similarity > highestSim then
                     highestSim = Similarity
                 End
    bestDef = s
    End
return bestDef
```

Algorithm 1 illustrates the pseudocode of the proposed method. The process starts with the input of the target word (*w*) and the context sentence (*cs*) into the system. The provided context sentence is pre-processed. Following this, the system retrieves synsets of the target word from the UKC, extracting various glosses associated with the target word. The context sentence is then encoded using PuoBERTa. Next, each synset gloss of the target word is encoded, and a similarity measure is computed for each gloss in comparison to the context sentence. The algorithm identifies and returns the gloss with the highest similarity measure as the correct definition of the target word.

C. Datasets

To construct the evaluation data set, the Senseval-3 lexical sample structure by Mihalcea, et al. [31] was adopted. Senseval-3 is one of the evaluation benchmark datasets and a follow-up to Senseval-1 and Senseval-2. The dataset was bult using Open Mind Word Expert system proposed in [32]. To construct the evaluation dataset, words that were mapped to more than one synset in the African Wordnet were extracted, together with their glosses and example sentences. Additional words, glosses, and example sentences were extracted from Oxford and Pharos bilingual dictionaries, the different senses were indicated with numbers superscript in the dictionaries. From these resources, an evaluation dataset of 1200 Setswana sentences was created. Currently, there is no existing WSD evaluation dataset for Setswana, this dataset serves as a valuable resource for evaluating and benchmarking models designed to addresses the complexities of polysemy in Setswana.

D. Experimental Settings

All experiments were conducted using the Python programming language with preinstalled NLTK [33] and scikit-learn library [34] run in Window 10 Pro, 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 1.69 GHz, 17GB installed ram. For experimentation, each context sentence was passed into the algorithm for pre-processing and encoding. Then the glosses of the polysemous word in that context were extracted from the UKC and subsequently encoded iteratively to generate gloss embeddings. With each iteration, a similarity measure was computed between the context and gloss embeddings, and the gloss with the highest similarity was selected as the appropriate sense gloss, which was then compared with the ground truth. If the chosen gloss aligned with the ground truth, the count of correctly disambiguated variables increased by 1. If not, the count of incorrectly disambiguated variables was incremented. For evaluation metrics, we adopted the accuracy (A) and error rate (E) metrics on (2) and (3) utilized in [35] methodology.

$$A = \text{Correctly Disambiguated} / \text{Number of Test Instances} \quad (2)$$

$$E = \text{Incorrectly Disambiguated} / \text{Number of Test Instances} \quad (3)$$

Accuracy is a metric that measures the proportion of correctly identified instances out of the total number of instances. In this context, it is calculated as the number of correctly disambiguated variables divided by the total number of test instances, as shown on Eq. (2). Error rate, on the other hand, represents the proportion of incorrectly identified instances out of the total number of instances. It is calculated as the number of incorrectly disambiguated variables divided by the total number of test instances, as illustrated in Eq. (3).

IV. Results

This section presents the results of the proposed algorithm.

The results of the experiment results are presented Table I. Out of a set 1200 context sentences, the algorithm successfully disambiguated 1040 Setswana sentences accurately but misclassified 160, resulting in an accuracy rate of 86.66% and an error rate of 14.34%.

TABLE I. Evaluation Statistics and Results

Total number of context sentences for polysemous words	1200
Correctly disambiguated sentences	1040
Incorrectly disambiguated sentences	160
Accuracy	86.66
Error	14.34

Fig. 5 illustrates the accuracy and the error rate. The notable high accuracy and comparatively lower error rate suggest that the algorithm demonstrates effectiveness in distinguishing among multiple senses of polysemous words in Setswana context. Fig. 6 presents the disambiguation stats.

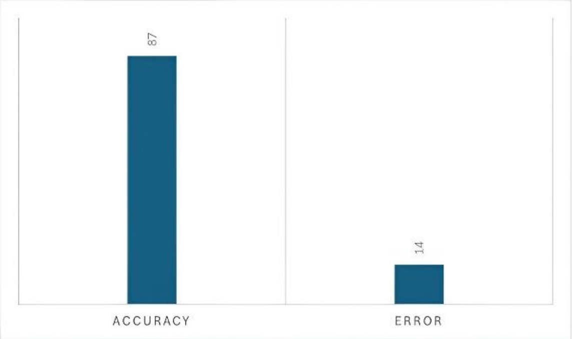


Fig. 5. Disambiguation accuracy and error rate.

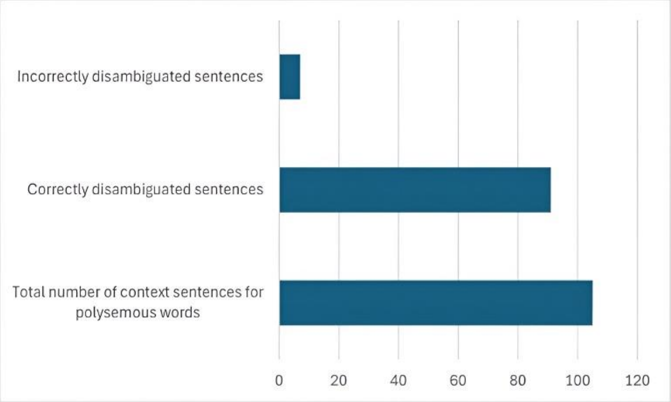


Fig. 6. Disambiguation statistics.

The error rate of 14.34% in the proposed Setswana WSD algorithm can be attributed to several factors. One reason for the error rate is the limited coverage of the Setswana UKC lexical resource. The UKC provides a valuable knowledge base for Setswana, however, it does not encompass all possible senses and glosses for every polysemous word encountered in the evaluation dataset. This limitation led to instances where the algorithm fails to identify the correct sense due to the absence of the appropriate gloss in the lexical resource. Another factor contributing to the error rate is the complexity and ambiguity of certain Setswana words. Setswana, being a morphologically rich language, contains words with intricate morphological structures and highly context-dependent meanings. The proposed algorithm, while effective in handling many cases, it struggles to accurately disambiguate such complex words, especially when the context provided is insufficient. Despite these factors contributing to the error rate, the proposed algorithm achieves a high accuracy of 86.66%, demonstrating its effectiveness in resolving polysemy for Setswana. Further improvements can be made by expanding the coverage of the Setswana UKC, incorporating additional linguistic features, and refining the disambiguation techniques to handle more complex and ambiguous cases.

To conduct a comparative analysis between our results and those of other researchers who utilized and adapted the Lesk algorithms for various languages, Fig. 7 presents a visual representation of the comparative performance based on accuracy.

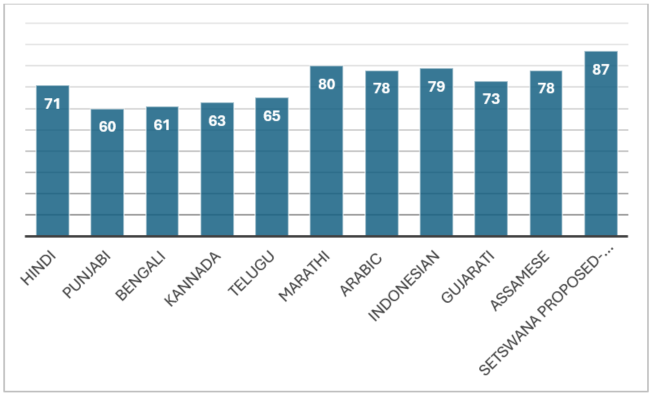


Fig. 7. Comparison of result for WSD for other languages.

For Hindi, Sharma and Joshi [36] used an evaluation corpus of 3000 context sentences, out of which 2143 were correctly disambiguated achieving an accuracy of 71%. Singh and Singh [37] tested the modified Lesk on 15 Punjabi polysemous words and achieved an average accuracy of 60 for all the ambiguous words. Pandit, et al. [38] used two test sets, the first test set with 10 and the second test set with 12 Bengali polysemous words and achieved an accuracy of 61%. Using a single word with five different senses and 2153 context sentences for Kannada, Parameswarappa and Narayana [39] obtained 63% accuracy while Eluri and Siddu [35] obtained 65% testing with 150 context sentences for Telegu. Patil, et al. [16]’s algorithm was evaluated on 6 Marathi polysemous words with a total 14 senses and achieved an overall accuracy of 80%. Arabic [40] and Indonesian [15] achieved almost the same accuracy with 0.1 difference, 78% and 79%. The Arabic WSD was tested on 50 polysemous words with 20 context sentences per word. For Indonesia, the algorithm was evaluated on 140 context sentences. Assamese obtained 78% evaluated on an annotated Assamese corpus with 15606 polysemous nouns. This paper achieved an accuracy of 87% on 1200 Setswana context sentences.

The comparative results on different datasets for various languages is caused by the inherent linguistic characteristics of each language, the size and quality of the datasets used. Each language requires its own evaluation data, specifically for resource-scarce languages like Setswana. Unlike English, which already has existing evaluation benchmark datasets such as Senseval and Semeval series, many low-resource languages lack such standardized datasets. This lack of evaluation data makes it challenging to directly compare the performance of WSD algorithms across different languages.

V. Discussion and Conclusion

This study highlights the critical role of WSD as a significant intermediate task in NLP applications. The Lesk algorithm, one of the first solution to address polysemy in NLP, has witnessed continuous adaptations by researchers for diverse languages. In the context of Setswana, this research introduces a novel approach, a simplified Lesk-based algorithm to effectively resolve polysemy. To address the computationally complex combinatorial comparisons inherent in traditional Lesk, this study leverages sentence embeddings and Cosine semantic similarity measures to model word sense glosses. The word sense glosses are modelled using a transformer-based language model PuoBERTa. The proposed method has been proven to achieve optimal performance in for Setswana WSD. The evaluation of the proposed algorithm on Setswana demonstrates significant success, with an accuracy of 86.66 and an error rate of 14.34. This surpasses the accuracy achieved by other Lesk-based algorithms developed for different languages. Additionally, this study provides a WSD evaluation dataset, currently unavailable, creating an essential benchmark for testing Setswana WSD models.

VI. Future Work

Future works involve expanding coverage of the Setswana UKC, the corpus size of the evaluation data and experimenting with diverse knowledge-based methods to determine their comparative performance. This includes incorporating additional linguistic morphological features, and refining the disambiguation techniques to handle more complex and ambiguous cases. Furthermore, we plan to integrate the proposed algorithm into a Setswana-English translation system, investigating its impact on translation quality as part of our ongoing research.

References

[1] S. Saxena, U. Chaurasia, N. Bansal, and P. Daniel, "Improved unsupervised statistical machine translation via unsupervised word sense disambiguation for a low-resource and Indic languages," IETE Journal of Research, pp. 1-11, 2022.

[2] K. Chowdhary, "Natural language processing for word sense disambiguation and information extraction," Fundamentals of Artificial Intelligence, pp. 603-649, 2020.

[3] N. Rahman and B. Borah, "Improvement of query-based text summarization using word sense disambiguation," Complex & Intelligent Systems, vol. 6, pp. 75-85, 2020.

[4] R. Navigli, "Word sense disambiguation: A survey," ACM computing surveys (CSUR), vol. 41, no. 2, pp. 1-69, 2009.

[5] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone," in Proceedings of the 5th annual international conference on Systems documentation, 1986, pp. 24-26.

[6] A. Kilgarriff and J. Rosenzweig, "Framework and results for English SENSEVAL," Computers and the Humanities, vol. 34, pp. 15-48, 2000.

[7] S. Banerjee and T. Pedersen, "An adapted Lesk algorithm for word sense disambiguation using WordNet," in International conference on intelligent text processing and computational linguistics, 2002: Springer, pp. 136-145.

[8] A. M. Butnaru and R. T. Ionescu, "ShotgunWSD 2.0: An improved algorithm for global word sense disambiguation," IEEE Access, vol. 7, pp. 120961-120975, 2019.

[9] F. Meng, "Graph and word similarity for word sense disambiguation," in 2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), 2020: IEEE, pp. 1114-1118.

[10] L. Zhong and T. Wang, "Towards word sense disambiguation using multiple kernel support vector machine," International Journal of Innovative Computing, Information and Control, vol. 16, no. 2, pp. 555-570, 2020.

[11] A. S. Maurya, P. Bahadur, and S. Garg, "Approach Toward Word Sense Disambiguation for the English-To-Sanskrit Language Using Naïve Bayesian Classification," in Proceedings of Third Doctoral Symposium on Computational Intelligence: DoSCI 2022, 2022: Springer, pp. 477-491.

[12] M. Kumar, P. Mukherjee, M. Hendre, M. Godse, and B. Chakraborty, "Adapted lesk algorithm based word sense disambiguation using the context information," International Journal of Advanced Computer Science and Applications, vol. 11, no. 3, pp. 254-260, 2020.

[13] E. F. Ayetiran, P. Sojka, and V. Novotný, "Enhancing Lesk Algorithm by Integrating Selectional Preferences," Journal of Language Modelling, vol. 9, no. 1, pp. 137-168, 2021.

[14] P. Tripathi, P. Mukherjee, M. Hendre, M. Godse, and B. Chakraborty, "Word sense disambiguation in Hindi language using score based modified lesk algorithm," International Journal of Computing and Digital Systems, vol. 10, pp. 2-20, 2020.

[15] S. Basuki, A. S. Kholimi, A. E. Minarno, F. D. S. Sumadi, and M. R. A. Effendy, "Word sense disambiguation (WSD) for Indonesian homograph word meaning determination by LESK algorithm application," in 2019 12th International Conference on Information & Communication Technology and System (ICTS), 2019: IEEE, pp. 8-15.

[16] A. P. Patil, R. Ramteke, R. Bhavsar, and H. Darbari, "Marathi Language Word Sense Disambiguation using Modifies Lesk Algorithm," International Journal of Advanced Computer Science and Applications, vol. 12, no. 7, pp. 365-372, 2021.

[17] S. Kaddoura and R. Nassar, "EnhancedBERT: A Feature-rich Ensemble Model for Arabic Word Sense Disambiguation with Statistical Analysis and Optimized Data Collection," Journal of King Saud University-Computer and Information Sciences, p. 101911, 2024.

[18] G. Bella et al., "A major wordnet for a minority language: Scottish gaelic," in Twelfth International Conference on Language Resources and Evaluation: Conference Proceedings, 2020: European Language Resources Association (ELRA), pp. 2812-2818.

[19] F. Giunchiglia, K. Batsuren, and A. Alhakim Freihat, "One World-Seven Thousand Languages (Best Paper Award, Third Place)," in International Conference on Computational Linguistics and Intelligent Text Processing, 2018: Springer, pp. 220-235.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1, pp. 4171-4186, 2019.

[21] L. Huang, C. Sun, X. Qiu, and X. Huang, "GlossBERT: BERT for word sense disambiguation with gloss knowledge," Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3509-3514, 2019.

[22] M. El-Razzaz, M. W. Fakhr, and F. A. Maghraby, "Arabic gloss WSD using BERT," Applied Sciences, vol. 11, no. 6, p. 2567, 2021.

[23] R. R. Kannan, R. Rajalakshmi, and L. Kumar, "IndicBERT based approach for Sentiment Analysis on Code-Mixed Tamil Tweets," 2021.

[24] V. Marivate, M. Mots' Oehli, V. Wagnerinst, R. Lastrucci, and I. Dzingirai, "PuoBERTa: Training and evaluation of a curated language model for Setswana," in Southern African Conference for Artificial Intelligence Research, 2023: Springer, pp. 253-266.

[25] B. Scarlini, T. Pasini, and R. Navigli, "Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation," in Proceedings of the AAAI conference on artificial intelligence, 2020, vol. 34, no. 05, pp. 8758-8765.

[26] K. Orkphol and W. Yang, "Word sense disambiguation using cosine similarity collaborates with Word2vec and WordNet," Future Internet, vol. 11, no. 5, p. 114, 2019.

[27] Sarika and D. K. Sharma, "Hindi word sense disambiguation using cosine similarity," in Proceedings of International Conference on ICT for Sustainable Development: ICT4SD 2015 Volume 2, 2016: Springer, pp. 801-808.

[28] S. Sari, R. Manurung, and M. Adriani, "Indonesian WordNet Sense Disambiguation using Cosine Similarity and Singular Value Decomposition," ICSIIT 2010, p. 234, 2010.

[29] R. Yatabe and M. Sasaki, "Semi-supervised word sense disambiguation using example similarity graph," in Proceedings of the graph-based methods for natural language processing (TextGraphs), 2020, pp. 51-59.

[30] A. Hari and P. Kumar, "WSD Based Ontology Learning from Unstructured Text Using Transformer," Procedia Computer Science, vol. 218, pp. 367-374, 2023.

[31] R. Mihalcea, T. Chklovski, and A. Kilgarriff, "The Senseval-3 English lexical sample task," in Proceedings of SENSEVAL-3, the third international workshop on the evaluation of systems for the semantic analysis of text, 2004, pp. 25-28.

[32] T. Chklovski and R. Mihalcea, "Building a sense tagged corpus with open mind word expert," in Proceedings of the ACL-02 workshop on Word sense disambiguation: recent successes and future directions, 2002, pp. 116-122.

[33] S. Bird, E. Klein, and E. Loper, Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.", 2009.

[34] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," the Journal of machine Learning research, vol. 12, pp. 2825-2830, 2011.

[35] S. Eluri and V. Siddu, "A Knowledge Based Word Sense Disambiguation in Telugu Language," International Journal of Engineering and Advanced Technology (IJEAT) ISSN, pp. 2249-8958, 2020.

[36] P. Sharma and N. Joshi, "Knowledge-Based Method for Word Sense Disambiguation by Using Hindi WordNet," Engineering, Technology & Applied Science Research, vol. 9, no. 2, 2019.

[37] J. Singh and I. Singh, "Word sense disambiguation: enhanced lesk approach in Punjabi language," International Journal of Computer Applications, vol. 129, no. 6, pp. 23-27, 2015.

[38] R. Pandit, S. Sengupta, S. K. Naskar, and M. M. Sardar, "Improving Lesk by incorporating priority for word sense disambiguation," in 2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT), 2018: IEEE, pp. 1-4.

[39] S. Parameswarappa and V. Narayana, "Target word sense disambiguation system for Kannada language," in 3rd International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2011), 2011: IET, pp. 269-273.

[40] A. Zouaghi, L. Merhbene, and M. Zrigui, "Word Sense disambiguation for Arabic language using the variants of the Lesk algorithm," WORLDCOMP, vol. 11, pp. 561-567, 2011.