ADVANCING CONTEXTUAL UNDERSTANDING: EVALUATING POLYSEMY DETECTION IN MODERN NATURAL LANGUAGE PROCESSING

Neha Nair[1,*], Mukta Joshi[1,*], Ramakrishna M[1]

[1]Department of Data Science and Computer Applications,
Manipal Institute of Technology, Manipal Academy of Higher Education,
Manipal, INDIA
E-MAIL: neha1.mitmpl2022@learner.manipal.edu, mukta.mitmpl2022@learner.manipal.edu, ramakrishna.m@manipal.edu
*Authors contributed equally

Abstract:

Understanding the nuances of language is fundamental to natural language processing (NLP) applications. The challenge of polysemy –ere a single word can carry multiple meanings depending on the context, continues to hinder progress in areas such as machine translation, sentiment analysis and question-answering. Despite it being a necessity to gain further advancement in these applications, minimal research has been conducted on how different models approach this problem. Finding the best fit for the polysemy involves taking into account the computational and practical qualifications of the models as well. In this paper, we implemented five models –BERT, RoBERTa, AutoSense, Sense2Vec and ELMo to gauge their performance and determine the best-suited model. Our results demonstrate that context-aware models like BERT and RoBERTa outperform static-embedding-based models like AutoSense and Sense2Vec. This research not only fills a critical void in the literature but also sets the stage for future innovations in semantic understanding, making polysemy detection a central focus for advancing NLP.

Keywords:

Polysemy detection, BERT, RoBERTa, ELMo, Sense2Vec, AutoSense, Natural Language Processing.

1. Introduction

Polysemy detection focuses on identifying and interpreting words with multiple meanings based on contextual usage. As a linguistic phenomenon, polysemy adds richness to language but poses significant challenges for computational systems, making it harder to accurately process and understand text[1]. This issue frequently arises in NLP tasks like machine translation, information retrieval, and sentiment analysis, where precise word sense disambiguation is crucial for success.

For instance, semantic variations make it challenging to distinguish nuanced meanings such as cell in biology vs telecommunications[2]. Individual interpretation and contextual ambiguity particularly in Named Entity Recognition (NER) aggravate these issues [3][4].Traditional dictionary definitions fail to capture the fluidity of real-world usage [5], while blending multiple meanings into single vector representations reduces model accuracy and hinders tasks like event detection [6]. High annotation costs in specialized fields like biomedical research further limit the availability of reliable polysemy datasets [4].

The integration of polysemy detection across various fields not only enhances accuracy and cultural sensitivity but also improves the overall effectiveness of communication and analysis in an increasingly complex and interconnected world. Addressing these challenges requires robust solutions to advance polysemy detection. This paper implements and evaluates five models—BERT, RoBERTa, ELMo, Sense2Vec, and AutoSense using a base metric and a common training corpus, Senseval[7] —specifically for handling polysemy, establishing a definitive baseline for future research. While prior studies have compared these models on various aspects, the challenge of polysemy has not received sufficient attention. By focusing on this gap, this study aims to bridge the current understanding of polysemy in NLP with the critical advancements needed to enhance model performance.

The findings in this paper demonstrate the shift from static word embeddings, such as those in Sense2Vec, to contextual embeddings in models like BERT and RoBERTa, which significantly improved the ability to resolve polysemy, as evidenced by our evaluation of the Senseval dataset, where BERT achieved a higher F1 score in distinguishing word senses. BERT and RoBERTa's ability to learn word meanings from context outperforms Sense2Vec's static multi-sense embeddings, achieving a more accurate representation of polysemous terms by leveraging dynamic, context-aware embeddings. The flexibility of BERT and RoBERTa models, combined with their ability to be fine-tuned for a variety of NLP tasks, allows for more precise polysemy detection in complex and evolving language contexts, outperforming traditional static models.

2 Literature Review

Numerous papers compare multiple models on various other dimensions such as text classification, machine translation etc. along with studies that explore the polysemy handling of individual models. This section discusses the existing research on polysemy handling for each model independently.

2.1 Traditional Models

WordNet's ambiguity and computational overhead in word sense disambiguation (WSD) of polysemous words have been critiqued, with a new model being proposed to address these limitations. This model introduces clue words to organize polysemous words more effectively.

2.2 Static-Embedding Based Models

Sense2Vec addresses the challenge of polysemy in neural word embeddings by disambiguating word senses into separate embeddings [12]. This method uses supervised labels for clustering word senses based on their context, reducing computational overhead and improving word-sense modelling. Key findings include disambiguation of word senses by parts of speech, the distinction between positive and negative sense vectors, and improved accuracy in syntactic dependency parsing. Related works include Word2Vec [13], Wang2Vec [14], and the Multi-Prototype Vector-Space Model [15].

2.3 Context-Based Models

A. Gar´ı Soler et. al [8] in their study presented how BERT and ELMo differentiate between monosemous and polysemous words in English and examined whether these models encode information about lexical polysemy and sense distributions. Using the Usim task, which predicts word similarity in context without sense annotations, the study classifies polysemous words into three bands (poly-same, poly-rand, and poly-bal) and evaluates them with the SelfSim score. The results show that as the number of senses increases, the SelfSim score decreases. BERT outperforms ELMo, particularly in predicting polysemy levels and word clusterability. The findings suggest that semantic information is distributed across BERT and ELMo layers, with higher disambiguation layers yielding better predictions. Relevant related work includes BERT's word sense disambiguation [9] [10] and studies on lexical polysemy in static embeddings [11].

2.4 Domain Specific Applications

J. A. Lossio-Ventura et. al [16] found a method to detect polysemy in biomedical terms using new features extracted from textual datasets and an induced co-occurrence graph. The approach uses two dictionaries to distinguish term usage across domains (biomedical and agronomy) and employs statistical measures to extract features. A multilayer perceptron algorithm shows the best performance for feature mixing. The method is effective for polysemy prediction, evaluated through accuracy, precision, recall, and F-measure. Related work includes ambiguity detection [17] and polysemy in prepositions [18].

The research reveals a significant gap in research comparing the polysemy handling capabilities of multiple models directly. Most studies either focus on individual models or prioritize other aspects of language processing over polysemy. Despite this, valuable insights have emerged from various model-specific investigations. BERT, in particular, has consistently demonstrated superior performance in encoding lexical polysemy and disambiguating word senses, especially when compared to models like ELMo. Sense2Vec also offers an innovative approach by clustering word senses and reducing computational costs, further contributing to advancements in polysemy handling. While traditional methods like WordNet-based WSD have faced critiques for their complexity and inefficiency, newer algorithms introducing clue words offer promising improvements. Transformer models, especially RoBERTa, continue to dominate in related tasks, proving their flexibility and effectiveness across different NLP challenges.

3 Experimental Setup

To establish the experimental setup, existing model features were evaluated for utility, and necessary modifications were identified.

The experiments were conducted in either the Kaggle or Google Colab environment, leveraging an NVIDIA T4 GPU

TABLE 1. Comparison of Models for Word Sense Disambiguation

Model	Advantages	Disadvantages
BERT (Base, Uncased)	• Strong contextual understanding. • Pretrained on large corpora.	• High inference cost. • Quadratic time complexity.
RoBERTa (Base)	• Improved BERT variant. • Robust predictions, better long-range handling.	• High computational demand. • Resource-intensive.
Sense2Vec	• Fast disambiguation. • Sense-specific embeddings.	• No context handling. • Weak on long/complex sequences.
ELMo (Logistic Regression)	• Lightweight and fast. • Easy to train.	• Lower accuracy. • Instability on complex data.
ELMo (LSTM)	• Rich embeddings. • Good with local dependencies.	• Overfitting risk. • Slow training, struggles with long text.
AutoSense (KMeans)	• Strong unsupervised WSD. • Stable, minimal overfitting.	• High tuning complexity. • Sensitive to parameters.

accelerator, with Python v3 as the programming language. The Senseval dataset, consisting of multiple .pos files, was used for this task. These files were parsed using Python's xml.etree.ElementTree module. The preprocessing phase involved extracting sentences and their corresponding labels, followed by exploratory data analysis (EDA) to visualize the distribution of labels and sentence lengths

The Senseval dataset was then used for implementation, with results plotted and tabulated to assess performance. This dataset is a benchmark for Word Sense Disambiguation (WSD) tasks, consisting of .pos files for ambiguous words like hard,interest,line and serve. Senseval is considered due to its lexical resource alignment and task relevance as well. Each file contains sentences labeled with sense identifiers (e.g., line1, line2), representing distinct meanings as defined by lexical resources like WordNet.

Each model was evaluated on metrics like F1 score, accuracy, recall and precision. Loss curves were plotted for better visualization of performance.

The following section provides a comprehensive summary of all libraries, modules, imports, tools, and additional packages utilized throughout the experiments, including optimizers and learning rate schedulers.

3.1 BERT

BERT (Bidirectional Encoder Representations from Transformers), specifically the bert-base-uncased variant was implemented with the following specifications.

- Dataset Split Ratio- 80:10:10
- Optimizer- AdamW with learning rate scheduler
- Pateince- 3 epochs
- Training epochs- 10
- Batch size- 16

The sentences were tokenized using the BERT tokenizer with labels encoded as integers. The pre-trained BERT model for sequence classification was fine-tuned. Early stopping with patience was incorporated to prevent overfitting, and a custom learning rate scheduler was dynamically adjusted the learning rate during training.

Both training and validation loss/accuracy over epochs was tracked. After training, the model was evaluated using the validation set, and classification reports were generated. A confusion matrix was also plotted to analyze incorrect classifications across different classes. Finally, the model and tokenizer were saved for future inference.

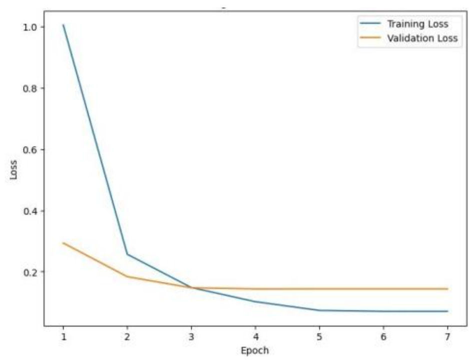

FIGURE 1. BERT

3.2 RoBERTa

RoBERTa (Robustly Optimized BERT Pretraining Approach), specifically the roberta-base variant was implemented with the following specifications.

- Dataset Split Ratio- 80:10:10
- Optimizer- AdamW with learning rate scheduler
- Pateince- 3 epochs
- Training epochs- 10
- Batch size- 16

Tokenization of sentences was performed using the RoBERTa tokenizer, which preserved the subword embeddings while ensuring compatibility with the RoBERTa model's vocabulary.

The RoBERTaForSequenceClassification model was fine-tuned for this classification task. To mitigate overfitting, early stopping was implemented.

While training the model, both training and validation loss/accuracy metrics were tracked. After the training phase, the model was evaluated on the test set and classification reports were generated. Additionally, a confusion matrix was plotted to analyze model performance.

To validate the effectiveness of the model, learning curves (training and validation loss/accuracy) were visualized to confirm convergence and generalization. The trained RoBERTa model and tokenizer were saved for future inference tasks, ensuring reproducibility and utility for downstream applications.

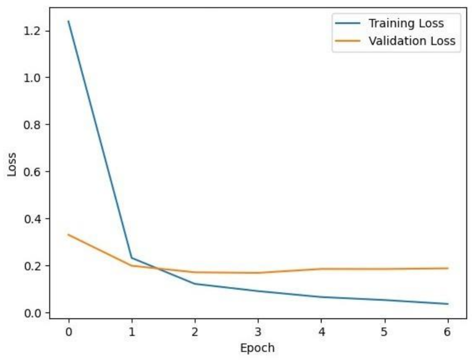

FIGURE 2. RoBERTa

3.3 ELMo

A word sense disambiguation (WSD) model using Embeddings from Language Models (ELMo) was implemented with two distinct approaches: Logistic Regression and an LSTM-based neural network. A custom function was prepared to tokenize each sentence from embedding generation into character IDs.

The AllenNLP framework was used to generate context-aware ELMo embeddings. The pre-trained ELMo model was initialized using the options.json and lm_weights.hdf5 files from the AllenNLP repository.

Datasets were split in an 80-10-10 ratio and PyTorch datasets were created for efficient training and 2 pipelines were built. The first one used ELMo with logistic Regression by averaging ELMo embeddings over sentence lengths. These fixed length embeddings were then passed into a Logistic Regression Classifier with hyper parameter tuning done. Model performance was evaluated using metrics like accuracy, precision, recall and F1 scores.

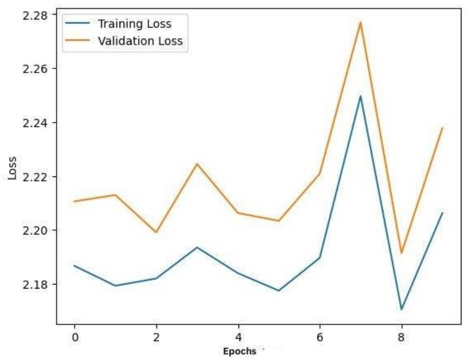

FIGURE 3. ELMo(Logistic)

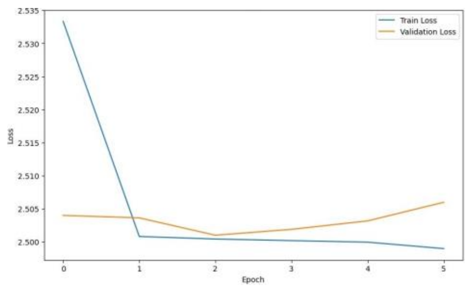

FIGURE 4. ELMo(LSTM)

The second pipeline had ELMo embeddings combined with an LSTM based neural network implemented in PyTorch. The LSTM model had 2 hidden layers and dropout regularization to prevent overfitting. Adam optimizer with a learning rate scheduler was used to optimize the model. Early stopping with patience of three epochs was used to halt training when validation loss showed no improvement.

Metrics such as training and validation losses were tracked over epochs. After training, confusion matrices were plotted to visualize model performance across different sense labels, and the models were saved for future inference using pickle. A custom inference function was also created to allow predictions on new sentences by generating their ELMo embeddings and feeding them into the trained classifiers.

3.4 Sense2Vec

In this work, a machine learning pipeline was implemented for word sense disambiguation using the Senseval dataset and Sense2Vec embeddings.

The Sense2Vec model was downloaded and loaded using the sense2vec package. Sentence tokenization and POS(Parts of Speech tagging) was done. Fixed-sized vectors representing features for classification were formed using word embeddings for each token averaged over each sentence.

A Logistic Regression model was trained on these features for classification. The dataset split had 80-20 train-validation split and the performance was evaluated using metrics like accuracy, precision, recall, F1 scores and a confusion matrix.

Different values of the regularization parameter C and varying kernel types (linear,rbf,poly and gamma) values were also used. The better performing model among both the classifiers was selected.

This pipeline demonstrates the application of Sense2Vec embeddings for word sense disambiguation with logistic regression and SVM classifiers. The incorporation of grid search optimization highlights the importance of hyperparameter tuning in improving classification performance. The overall approach serves as an effective integration of embedding techniques and machine learning methodologies for semantic tasks.

3.5 AutoSense

A contextual embedding-based model for word sense disambiguation (WSD) was implemented, leveraging the principles of AutoSense-like approaches with a focus on contextual embeddings, clustering, and classification. The experiment was conducted on the Senseval dataset, which contains multiple .pos files that were parsed using Python's xml.etree.ElementTree module.

BERT (Bidirectional Encoder Representations from Transformers) model was implemented to generate sentence-level embeddings. The BERT embeddings were generated using the BertModel and BertTokenizer classes. Once the embeddings were created, cosine similarity was calculated to identify pairs of semantically similar sentences.

To assign sentences to different sense clusters, KMeans clustering was employed. A threshold-based method was applied to filter out sentence pairs based on their similarity scores, helping to establish relationships between highly similar sentences. For classification, the sentence embeddings were then passed to a neural network classifier built using TensorFlow's Keras API. The network comprised dense layers trained over 20 epochs, with early stopping based on validation accuracy. The model was compiled with Adam optimizer and sparse_categorical_crossentropy as the loss function. Throughout the process, training and validation accuracy was tacked along with the loss curves. The model was evaluated using standard metrics such as F1-score, accuracy, and confusion matrix, and the results were logged in a separate text file for detailed analysis. The classification report was generated.

Key libraries used in this experiment included TensorFlow for neural network implementation.

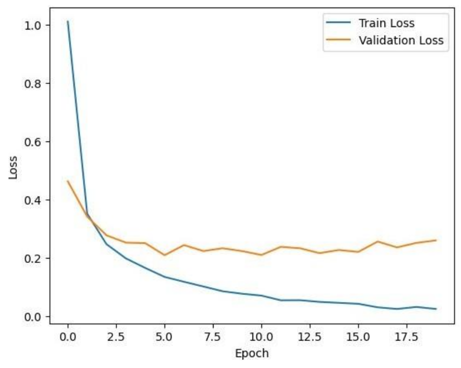

FIGURE 5. AutoSense

4 Results

In analyzing the results across all models, it is evident that BERT (base uncased) is the most suitable model for this task, achieving a high accuracy of 0.97, along with a precision, recall, and F1-score of 0.96 or higher. This model's slightly overfitted learning curves suggest that although the model is very powerful at capturing nuanced word contexts, further regularization techniques (such as dropout, data augmentation, or early stopping) could help it generalize even better. BERT's attention-based mechanism allows it to excel in tasks requiring deep contextual understanding, by modeling the complex relationships between words in different contexts.

TABLE 2. Experimental Setup Requirements

Model Name	**Libraries Used**	**Modules/Imports**	**Additional Tools/Packages**
BERT (base uncased)	Hugging Face Transformers, PyTorch	transformers, xml.etree.ElementTree, torch, sklearn.metrics.classification_report	matplotlib, seaborn, AdamW optimizer, Learning Rate Scheduler
RoBERTa	Hugging Face Transformers, PyTorch	transformers, xml.etree.ElementTree,torch, sklearn.metrics.classification report	matplotlib, seaborn, AdamW optimizer, Learning Rate Scheduler
Sense2Vec	sense2vec,SpaCy	sense2vec,spacy, xml.etree.ElementTree	sklearn (Logistic Regression), matplotlib, seaborn
ELMo (Logistic Regression)	AllenNLP,PyTorch	xml.etree.ElementTree, sklearn.metrics,torch	AllenNLP embeddings,matplotlib,seaborn, RandomizedSearchCV
ELMo (LSTM)	AllenNLP,PyTorch	xml.etree.ElementTree,torch, sklearn.metrics	AllenNLP embeddings, matplotlib,seaborn,Adam optimizer, ReduceLROnPlateau
AutoSense (KMeans)	Hugging Face Transformers,TensorFlow,scikit-learn	transformers,xml.etree.ElementTree, sklearn.cluster.KMeans	matplotlib, seaborn, Adam optimizer, Learning Rate Scheduler

TABLE 3. Results Comparison of Models

c

Comparison of Models for Word Sense Disambiguation							
Model Name	Accuracy	F1-Score	Precision	Recall	Predicted Label (Reference)	Training Time/Complexity	Inference from Learning Curves
BERT (base uncased)	0.97	0.96	0.96	0.97	HARD1	$O(n^2)$	Slightly overfitting
RoBERTa	0.9688	0.9685	0.9699	0.9688	HARD2	$O(H \times (n \times d(n + d + m + 1)))$ per layer	Slightly overfitting
Sense2Vec	0.79	0.77	0.78	0.79	HARD1	$O(T \cdot d)$	-
ELMo (Logistic Regression)	0.33	0.23	0.25	0.33	cord	$O(T \cdot d^2_{ELMo}) + O(d_{ELMo} \cdot N)$	Instability in training
ELMo (LSTM)	0.2325	0.0877	0.051	0.2325	HARD1	$O(T \cdot d^2_{ELMo}) + O(T \cdot d^2_{LSTM})$	Overfitting
AutoSense (KMeans)	0.91	0.91	0.91	0.91	cord	$O(n^2)$	Overfitting

RoBERTa, while also achieving strong results with an accuracy of 0.9685, similarly demonstrates slightly overfitting behavior. Its complexity per layer, represented by the cumulative time complexity $O(H\times(n\times d(n+d+m+1)))$, results from its attention mechanism, which enhances performance but requires significant computational resources. Despite this, RoBERTa falls short of BERT in overall performance, likely due to its fine-tuning configurations. However, it still proves highly effective for tasks that require large-scale language understanding and could be competitive with further optimization.

Sense2Vec, though simpler, shows reasonable performance with an accuracy of 0.79 and balanced precision and recall. However, its lower complexity and computational needs reflect its limitations in handling deep contextual relationships compared to transformer models like BERT or RoBERTa. The training and learning curves suggest a model that doesn't generalize as well as BERT or RoBERTa, but its efficiency makes it an appealing option for applications where resources are constrained.

ELMo (Logistic Regression) and ELMo (LSTM) performed significantly worse, with accuracy dropping to as low as 0.33 and unstable training behavior. ELMo, despite its original promise in polysemy detection due to its bidirectional LSTM-based embeddings, struggles to achieve competitive results. This is primarily because ELMo embeddings are pre-trained and task-specific, making them less adaptable to fine-tuning for new tasks like BERT or RoBERTa. ELMo cannot capture the deeper contextual nuances that transformers excel at, especially since fine-tuning ELMo for this particular task was not feasible in this setup. The instability observed in the learning curves indicates that it could not converge properly, likely due to the inherent limitations of its architecture in handling modern tasks like word sense disambiguation when compared to more recent models.

AutoSense performed well, with an accuracy of 0.91 and an equally balanced F1-score, precision, and recall of 0.91. The KMeans clustering approach used in AutoSense for sentence embeddings enables it to effectively group semantically similar words, even without relying on labeled data. However, despite its simplicity, the model shows clear signs of overfitting in the learning curves. AutoSense's unsupervised nature may fail to capture deeper word-context relationships as effectively as transformer-based models, which use extensive attention mechanisms to learn the intricate dependencies between words. This overfitting could be attributed to the inherent limitations of clustering techniques in modeling the full complexity of language, especially when compared to attention-based models like BERT.

In summary, BERT emerged as the top-performing model, with its attention mechanisms and contextual embeddings providing superior accuracy in polysemy detection. This aligns with existing research, underscoring BERT's ability to capture nuanced meanings in context. AutoSense is a close second, providing a lightweight and computationally efficient alternative that, while effective, may struggle with generalization due to its simpler architecture. RoBERTa, which shares architectural similarities with BERT, also performed well but required extensive fine-tuning to optimize for this specific task. Sense2Vec and ELMo, while interesting in their approaches, do not perform well in comparison, with ELMo's inability to be fine-tuned for specific tasks and Sense2Vec's limitations in handling deep contextual relationships preventing them from competing with modern transformer-based architectures. The performance of these transformer models reinforces their suitability for polysemy detection, as they excel at integrating deep contextual information through bidirectional attention.

ELMo, despite its bidirectional LSTM architecture, showed limitations in accurately handling polysemous words, reflecting the constraints of pre-transformer models in capturing fine-grained contextual nuances. Similarly, Sense2Vec provided a lightweight alternative for polysemy detection but lacked the depth needed for complex disambiguation, as it relies on word-vector representations rather than contextual embeddings. AutoSense, an unsupervised clustering model, performed efficiently, but its simplicity made it less effective at distinguishing meanings in diverse contexts.

Models such as XLNet, T5, and large language models (LLMs) like GPT variants were intentionally excluded. These models, while capable of deep language generation, bring additional computational costs and are often optimized for text generation or bidirectional prediction tasks rather than targeted word-sense disambiguation. Additionally, simpler embedding models such as GloVe or Word2Vec were excluded because their static embeddings lack adaptability to sentence context, rendering them ineffective for tasks that rely on dynamic word meanings. PolyLM [19], a model specific to polysemy, was also intentionally excluded. While it may excel in this task, the limited availability of research, a single paper, and a GitHub repository make it difficult to justify its inclusion. This paper encourages further research to enhance models like PolyLM in addressing polysemy-related challenges.

The decision to compare these particular models was rooted in the goal of balancing accuracy, interpretability, and accessibility for practical use. BERT and RoBERTa represent state-of-the-art transformer capabilities, while Sense2Vec and AutoSense offer feasible alternatives for less resource-intensive applications. ELMo serves as an intermediate baseline, illustrating the performance gap between early contextual models and modern transformers. The inclusion of these models addresses a significant shortcoming in existing research, providing a practical assessment of various methods for polysemy detection and underscoring the relevance of efficient, accessible models for real-world NLP applications.

5 Conclusion

Among the evaluated models, BERT demonstrated the highest accuracy for polysemy detection due to its bi-directional attention mechanism, while Sense2Vec provided computational efficiency suited for resource-constrained applications. RoBERTa while slightly less accurate than BERT also performed well due to its robust training strategies. ELMo underperformed due to its reliance on static embeddings and limited adaptability, making it less-effective at capturing fine-grained word meanings.

The chosen models represent a spectrum of complexity and methodologies, from transformer-based architectures to simpler, computationally efficient approaches. It also highlights the unique strengths and limitations of each model, offering insights into their ability to disambiguate words with multiple meanings and demonstrating the balance of performance, computational feasibility, and model interpretability. The findings serve as a foundation for further applications of polysemy detection in particularly high-stakes domains.

Acknowledgements

During the preparation of this work, the authors used ChatGPT to improve readability and ScopusAI for citing references. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

[1] E. Gomede, ”Deciphering Shades of Meaning: Enhancing NLP Systems to Navigate the Challenges of Polysemy,” in *Artificial Intelligence in Plain English*, Mar. 30, 2024. [Online]. Available: https://ai.plainenglish.io/deciphering-shades-of-meaning-enhancing-nlp-systems-to-navigate-the-challenges-of-polysemy-a935b262f63b.

[2] W. Geng and M. Liang, ”From words to senses: A sense-based approach to quantitative polysemy detection across disciplines,” *Journal of English for Academic Purposes*, 2024. Available: https://doi.org/10.

[3] H. Liang, The Perception and Acquisition of Chinese Polysemy, 2024. Available: https://doi.org/10.

[4] L. Li, X. Huang, J. Ma, and Y. Li, ”An Enhanced MRC Framework with Triggers as Explanations for Named Entity Recognition,” in *IMCEC 2022 - IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference*, 2022. Available: https://doi.org/10.1109/IMCEC55196.2022.9786639.

[5] J. DeCesaris, ”Polysemy and sense extension in bilingual lexicography,” in *EURALEX Proceedings*, 2018. Available: https://doi.org/10.29092/978-83-64770-15-3_19.

[6] C. Du, H. Sun, J. Wang, and B. Ma, ”Investigating capsule network and semantic feature on hyperplanes for text classification,” *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*, 2019. Available: https://doi.org/10.18653/v1/D19-5005.

[7] P. Edmonds, ”Senseval-2: Overview,” *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (Senseval-2)*, 2001, Toulouse, France.

[8] A. Gar´ı Soler and M. Apidianaki, ”Let's Play Mono-Poly: BERT Can Reveal Words' Polysemy Level and Partitionability into Senses,” *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 825-844, 2021. doi: https://doi.org/10.1162/tacl_a_00400

[9] E. Reif, A. Yuan, M. Wattenberg, F. Viegas, A. Coenen, A. Pearce, and B. Kim, ”Visualizing and measuring the geometry of BERT,” in *Advances in Neural Information Processing Systems*, vol. 32, pp. 8592-8600, 2019.

[10] G. Wiedemann, S. Remus, A. Chawla, and C. Biemann, ”Does BERT make any sense? Interpretable word sense disambiguation with contextualized embeddings,” in *Pro- ceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)*, pp. 161-170, 2019.

[11] A. Jakubowski, M. Gasic, and M. Zibrowius, ”Topology of word embeddings: Singularities reflect polysemy,” in *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pp. 103-113, 2020. doi: https://www.aclweb.org/anthology/2020/

[12] A. Trask, P. Michalak, and J. Liu, ”sense2vec - A Fast and Accurate Method for Word Sense Disambiguation In Neural Word Embeddings,” 2015. Available: https://doi.org/10.48550/arXiv.1511.06388

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ”Efficient Estimation of Word Representations in Vector Space,” *arXiv preprint*, arXiv:1301.3781, 2013. Available: https://arxiv.org/abs/1301.3781

[14] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, ”Two/Too Simple Adaptations of Word2Vec for Syntax Problems,” in *Proceedings of NAACL-HLT*, 2015, pp. 1299-1304. Available: https://www.aclweb.org/anthology/N15-1142/

[15] J. Reisinger and R. J. Mooney, ”Multi-Prototype Vector-Space Models of Word Meaning,” in *Proceedings of NAACL-HLT*, 2010, pp. 109-117. Available: https://www.aclweb.org/anthology/N10-1014/

[16] J. A. Lossio-Ventura, C. Jonquet, M. Roche, and M. Teisseire, ”Automatic biomedical term polysemy detection,” in *Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC 2016*, 2016.

[17] T. Baldwin, Y. Li, B. Alexe, and I. R. Stanoi, ”Automatic term ambiguity detection,” in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 804-809, 2013.

[18] M. Ko¨per and S. S. im Walde, ”Distinguishing polysemy and homonymy by verb sense clustering,” in *Proceedings of the First Workshop on Metaphor in NLP*, pp. 11-17, 2014.

[19] A. Ansell, F. Bravo-Marquez, and B. Pfahringer, ”PolyLM: Learning about Polysemy through Language Modeling,” in *Proceedings of the 16th Conference of the European Chapter of the Association for Compu- tational Linguistics (EACL)*, 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2101.10448