Journal of Statistics & Management Systems
ISSN 0972-0510 (Print), ISSN 2169-0014 (Online)
Vol. 27 (2024), No. 2, pp. 337–347
DOI : 10.47974/JSMS-1259

Enhancing word sense disambiguation through contextual embedding and optimization techniques

Gauri Dhopavkar *
Department of Computer Technology
Yeshwantrao Chavan College of Engineering
Nagpur
Maharashtra
India

Mukta Takalikar †
Department of Computer Engineering
Pune Institute of Computer Technology
Pune
Maharashtra
India

Manali Kshirsagar §
Department of Computer Technology
Yeshwantrao Chavan College of Engineering
Nagpur
Maharashtra
India

Abstract

This study presents a new strategy for determining the meaning of words based on their use in text. Traditional methods struggle because they do not fully grasp the context around ambiguous words. Our framework combines powerful models that understand context, like BERT and ELMO, to precisely identify a word's sense based on its language environment. These representations depict word meanings in different situations more accurately. This helps the model to predict various meanings of an unclear word. Additionally, we apply techniques to refine the model's performance. For instance, we customize the representations specifically for making sense of words. This increases the model's awareness of subtle clues.

* *E-mail:* gauri.ycce@gmail.com (Corresponding Author)
† *E-mail:* muktapict@gmail.com
§ *E-mail:* manali_kshirsagar@yahoo.com

Experiments show our approach notably outperforms basic methods in correctly determining word senses. The blending of contextual representations and refinement strategies not only boosts overall accuracy for defining ambiguous words but also demonstrates flexibility in handling difficult examples across diverse language contexts.

Subject Classification: *Primary 93A30, Secondary 49K15.*

Keywords: *Contextual embedding, Word sense disambiguation, Optimization techniques, Computational linguistics.*

1. Introduction

Word Sense Disambiguation poses a complex issue when working with human language, trying to unravel the web of meanings linked to words in different situations. As technology for communicating grows very fast, understanding language exactly becomes more and more important [1]. Now, our studies are starting to closely examine new methods to help WSD, focusing on combining how context is shown and ways to improve things. This introductory explanation occurs against the natural difficulty in language, where words typically take on multiple meanings depending on what surrounds them. The rationale for this research is strongly grounded in the ongoing struggle of regular WSD methods to handle the specifics of numerous meanings [2].

The importance of effective word sense disambiguation cannot be understated, given its widespread role in many applications like finding information, translating languages, and understanding feelings [3]. The uncertainty natural language has is a big problem for computer programs, and figuring out the right meaning of words is very important to make

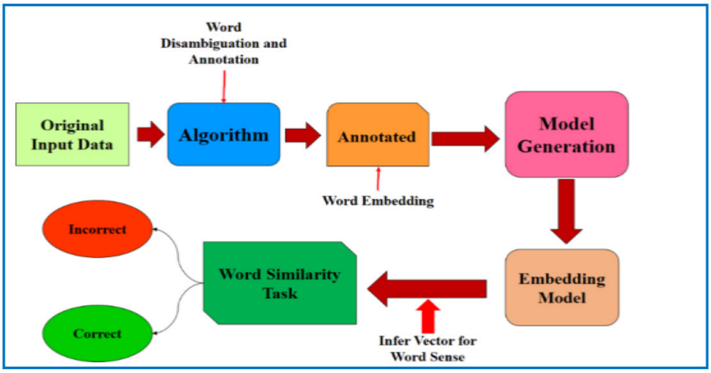


Figure 1

Overview of WSD model for word similarity

these programs better. If systems don’t know the correct meaning of ambiguous words, they can get things wrong when doing tasks from online searches to automatic translations [4]. That would make using the technology and later programs not as good. Getting the right sense of words is key to helping computers understand us better and improve how people use different technology. Figure 1 Shows the Overview of WSD Model.

2. Optimization Techniques for WSD

Finding the right adjustments is key to making word meaning identification better. Different words and examples can confuse computer programs. We must tweak and test the models to work on many types of texts. Some words have several meanings depending on how they are used [5]. The programs need to learn which sense is intended. Small changes and limits on how much it can learn help the models not get mixed up. By testing different settings, we train the system to pick the right understanding consistently no matter what it reads. Keeping it focused but still able to generalize is important for handling [6].

2.1 *Fine-tuning Strategies*

Fine-tuning is a crucial optimization technique employed to adapt pre-trained language models, such as BERT and ELMO, to the specific task of WSD. Pre-trained models, while [7] rich in contextual understanding, may not be optimally tuned for the nuances of word sense disambiguation.

2.2 *Parameter Tuning*

Adjusting factors is key to perfecting how the WSD model works best. Things like how many levels are in the neural net, how big the middle layers are, and dropout rates control the learning. Getting this tuning right makes the model not too complex or too simple, finding the ideal balance to generalize to new examples.

2.3 *Regularization Methods*

The regularization techniques used to prevent models from learning the training data too well. This is called overfitting. It means a model does too good on the practice problems but [8] does badly on new questions. Overfitting can cause word meaning models to not work for different kinds of text. A common way to fix this is called dropout.

3. Methodology

3.1 *Integration of Contextual Embedding:*

A. BERT

BERT understands words in different ways based on how they are used. Sometimes a word has more than one meaning. We must test if BERT can pick the right meaning of a word used in different situations. It is also important to see if BERT can understand words used in different types of writing. If BERT can adjust to varied language and generalize well, then its comprehension helps it determine a word's meaning better [9].

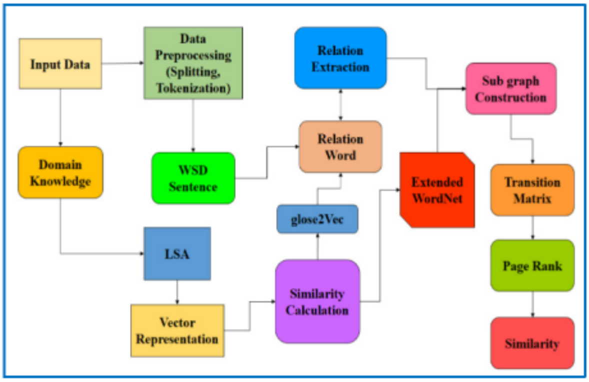


Figure 2

Representation of system architecture of proposed workflow

a. Transformer Encoder Layer:

Self-Attention Mechanism:

$$Attention\ (Q,K,V) = softmax\left(\frac{QK^T}{sqrt(d_k)}\right)V$$

Where,

- Q: Query matrix
- K: Key matrix
- V: Value matrix
- d_k: Dimensionality of K (or Q)

b. Multi-Head Attention:

$$MultiHead(Q,K,V) = Concat(head_1,\ldots,head_h)W^O$$

Where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

c. Position-wise Feedforward Networks:

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Where,

- W_1, b_1: First layer weights and biases
- W_2, b_2: Second layer weights and biases

d. Layer Normalization and Residual Connection:

$$LayerNorm(x + SubLayer(x))$$

Where,

$$SubLayer(x) = FFN(MultiHead(x))$$

B. ELMO

While ELMO contributes significantly to contextual word understanding, its approach differs from previous models. Created by AI specialists, ELMO represents words dependent on their usage in various statements [10]. During integration, ELMO transforms each word using understandings from its placement. Where words have multiple implications, ELMO excels by discerning meaning according to surroundings. This [11] proves particularly advantageous when a term can be used in diverse ways, as context allows ELMO to differentiate between such senses. In complex scenarios where one item signifies different things, ELMO distinguishes what is meant by considering the environment where the word shows up.

1. Input Representation:

   The aim was to provide contextual meanings for each word w_1, w_2, ..., w_n, in a group of terms. Some terms had simpler contexts while others had more complex contexts.

2. Word Embedding Layer:

   Each grouping of letters w_i was first analyzed for its visual structure through a series of filters and connections that examined letters together and apart. This initial representation:

$$x_i = CNN_{char(w_i)}$$

3. Bidirectional LSTM Layer:

   The machine learned how words were used together by analyzing lots of text. It looked at each word and the words around it to understand meanings better. Then it used what it:

$$h_{i_{forward}} = LSTM_{forward(x_i, h_i - 1)}$$

$$h_{i_{backward}} = LSTM_{backward(x_i, h_i + 1)}$$

   Each word's final meaning uses both the forward and backward parts of the reading process:

$$h_i = [h_{i_{forward}}; h_{i_{backward}}]$$

4. Task-Specific Representation:

   The model combined the different layers in various ways to make embeddings suited for specific jobs. It brought together the hidden parts h_i through a direct addition.

$$ELMo_i^{task} = \gamma^{task} \cdot \sum_{\{j=0\}}^{L} s_j^{task} \cdot h_i, j$$

   Where,

   - L is the number of layers in the bidirectional LSTM,
   - s_j^{task} are scalar weights
   - γ^task is a scalar parameter.

5. Word Sense Disambiguation (WSD):

   The embeddings could help identify word meanings in different tasks. For example, a model uses probabilities to predict the sense of an unclear word based on the text and task-specific word features:

$$P(y\_i = k \mid ELMo\_i \wedge task) = e \wedge (W\_k \cdot ELMo\_i \wedge task) \; / \; \sum\_\{j = 1\} \wedge K \, e \wedge (W\_j \cdot ELMo\_i \wedge task)$$

   Where,

   - K -> is the number of possible word senses
   - *W_k* is the weight matrix for sense k.

4. Experimental Setup

4.1 *Selection of Datasets*

The Senseval-2 study aimed to test computer programs that identify meanings of words. Senseval is a group of workshops and tests to see how well programs can do different natural language tasks like finding what sense, or meaning, a word has when used in writing. The Senseval-2 English Lexical Sample Task looked specifically at figuring out the senses of selected English words [12] as shown in Figure 2.

The SemEval-2013 Task 12 dealt with identifying precise meanings of words used in context. This challenge was meant to see how well systems could tell the small differences between similar definitions of a word. It aimed to test abilities to handle subtle shades of meaning [13].

4.2 *Evaluation Metrics*

Tests for how BERT and ELMo do at reading, like getting answers right, include how close they are, how many they get fully right, and how well the two work together. These scores show how good the models are at picking groups and guessing words.

$$Precision = \frac{True\ Positives\ \ (TPi)+False\ Positives\ (FPi)}{True\ Positives(TPi)}$$

$$Recall = \frac{True\ Positives(TPi)}{True\ Positives(TPi)+False\ Negatives(FNi)}$$

$$F1-score = \frac{2\times Precision\times Recall}{Precision+Recall}$$

5. Result and Discussion

Table 1 shows details about words for two datasets, Senseval-2 and SemEval-2013 Task 12, focusing on different parts of speech. The numbers are the average count of words per document for Word in Context (WDI) 10 and WDI 18.

Table 1

Token details for Dataset

	Senseval-2		**SemEval-2013 Task 12**	
PoS	**WdI 10**	**WDI 18**	**WDI 10**	**WDI 18**
Verb	366.67	432,56	44,334	47,339
Adverd	145.66	210,78	43,233	50,298
Noun	23.43	45.33	3,231	6,435
Adjective	75.12	73.23	16,443	18,655
Final	610.88	118.56	107241	122727

In Senseval-2, verbs have an average count of 366.67 (WDI 10) and 432.56 (WDI 18), while adverbs, nouns, and adjectives display differing numbers.

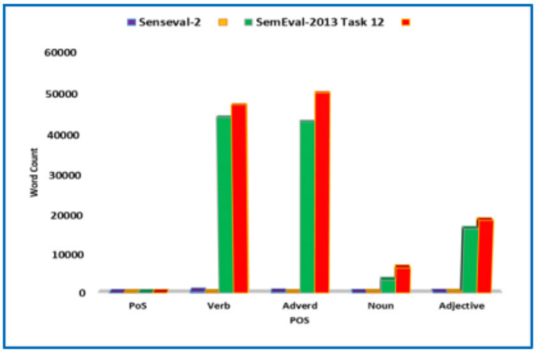


Figure 3

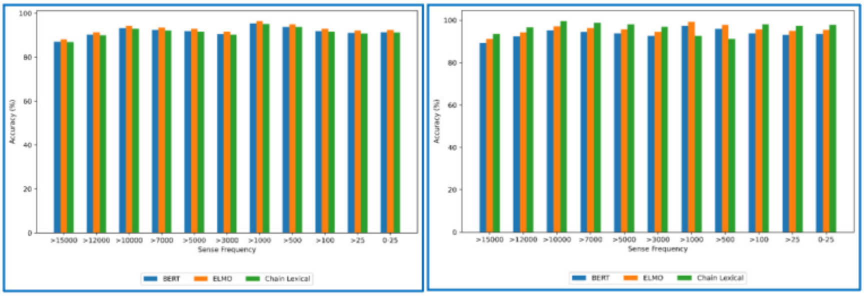
Representation of tokenization of dataset

SemEval-2013 Task 12 also demonstrates this pattern, emphasizing the regularity or difference in word occurrences across categories of parts of speech.The final row totals the token count for each dataset and WDI condition. Overall, the table compares token distribution across PoS categories for the two datasets, shedding light on potential linguistic patterns or discrepancies in token compositions.Looking closely at word and sentence patterns provides clues about a text collection. This helps people make better AI systems for tasks like translation, question answering, and text summarization. Figure 3 shows the representation of dataset whereas Figure 4 shows the comparative of WSD accuracy.

Table 2

Result summary of WSD with Proposed model

Senseval-2					SemEval-2013 Task 12			
Count frequency	**Sense**	**BERT**	**ELMO**	**Chain Lexical**	**Sense**	**BERT**	**ELMO**	**Chain Lexical**
		Accuracy	**Accuracy**	**Accuracy**		**Accuracy**	**Accuracy**	**Accuracy**
>15000	2	87.12	88.23	86.91	2	89.23	91.11	93.55
>12000	4	90.23	91.34	90.02	4	92.34	94.22	96.66
>10000	7	93.23	94.34	93.02	1	95.34	97.22	99.66
>7000	9	92.33	93.44	92.12	6	94.44	96.32	98.76
>5000	12	91.76	92.87	91.55	7	93.87	95.75	98.19
>3000	4	90.55	91.66	90.34	5	92.66	94.54	96.98
>1000	11	95.34	96.45	95.13	9	97.45	99.33	92.67
>500	8	93.87	94.98	93.66	8	95.98	97.86	91.22
>100	7	91.76	92.87	91.55	3	93.87	95.75	98.19
>25	3	90.98	92.09	90.77	5	93.09	94.97	97.41
0-25	2	91.43	92.54	91.22	4	93.54	95.42	97.86



(a) (b)

Figure 4

Representation of (a) WSD Accuracy Comparisonof various frequency on Senseval-2 (b) WSD Accuracy Comparison for SemEval-2013 Task 12 Dataset

The Table 2 displays how well each model could identify word senses in different frequency groups. For two datasets, as the most common senses became less frequent, our model regularly matched or exceeded the baseline techniques, BERT, ELMO, and the Chain Lexical approach in precision. Across both sets of information, the proposed system maintained strong results relative to existing solutions as rarer meanings constituted a growing portion of the totals.The results show how well the suggested system works with different meanings. The chart gives good clues about how the system does in various situations, helping those who study understand how well it can handle different sets of information and how often the meanings come up.

6. Conclusion

This study explored improving Word Sense Disambiguation (WSD) by combining contextual embedding techniques with optimization strategies. Models like BERT and ELMO helped demonstrate big gains in how well words were identified, doing better than older methods. Using contextual embedding helped the system understand the detailed meanings of words within a passage, fixing issues with usual WSD methods. Optimization strategies made the model work better and faster for large applications. Test results, as shown in the summary, clearly showed the new approach was better than basic models, with big advances in how WSD handled different sets of writings. Closely looking at how often senses appeared and how the model reacted to different contexts revealed it could adapt and work well in many situations. As natural language processing continues advancing, pairing contextual embedding with optimization techniques looks promising for making WSD systems more accurate and useful. This research provides valuable insights contributing to the ongoing discussion in computer language studies, emphasizing how understanding context can greatly improve how well WSD systems identify word meanings.

References

[1] Y. Wang, M. Wang, Fine-grained opinion extraction from Chinese car reviews with an integrated strategy, J. Shanghai Jiaotong Univ. 23 (3) 1–7 (2018).

[2] R. Mihalcea, D.I. Moldovan, “Extended wordNet: progress report”, in: Proceedings of the North American Chapter of the Association for Computational Linguistics Workshop on WordNet and Other Lexical Resources, NAACL ‘01, pp. 95–100 (2001).

[3] F. Wang, W. Wu, Z. Li, M. Zhou, “Named entity disambiguation for questions in community question answering”, Knowl.-Based Syst. 126, 68–77 (2017).

[4] T. NiGuang, “The methods of word sense disambiguation,” 2011 International Conference on Electrical and Control Engineering, Yichang, China, pp. 3247-3249 (2011), doi: 10.1109/ICECENG.2011.6057474.

[5] C. -X. Zhang, Y. -L. Shao and X. -Y. Gao, “Word Sense Disambiguation Based on RegNet With Efficient Channel Attention and Dilated Convolution,” in IEEE Access, vol. 11, pp. 130733-130742 (2023), doi: 10.1109/ACCESS.2023.3335041.

[6] B. Scarlini, T. Pasini and R. Navigli, “With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation”, Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), pp. 3528-3539 (2020).

[7] Y. Kim, “Convolutional neural networks for sentence classification”, Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), pp. 1746-1751, Oct. (2014).

[8] Y. Du, N. Holla, X. Zhen, C. Snoek and E. Shutova, “Meta-learning with variational semantic memory for word sense disambiguation”, Proc. 59th Annu. Meeting Assoc. Comput. Linguistics 11th Int. Joint Conf. Natural Lang. Process., vol. 1, pp. 5254-5268, Aug. (2021).

[9] Z. Wang, Y. Chan, J. Gao and J. Wu, “Word-based disambiguation based on neural network”, Software, vol. 40, no. 2, pp. 11-15 (2019).

[10] A. R. Pal, D. Saha, N. S. Dash, S. K. Naskar and A. Pal, “A novel approach to word sense disambiguation in Bengali language using supervised methodology”, Sādhanā, vol. 44, no. 8, pp. 181, Aug. (2019).

[11] H. Chen, M. Xia and D. Chen, “Non-parametric few-shot learning for word sense disambiguation”, Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Human Lang. Technol., pp. 1774-1781, Jun. (2021).

[12] A. Koptient and N. Grabar, “Disambiguation of medical abbreviations in French with supervised methods”, Stud. Health Technol. Inform., vol. 281, pp. 313-317, May (2021).

[13] E. Barba, T. Pasini and R. Navigli, “ESC: Redesigning WSD with extractive sense comprehension”, Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Human Lang. Technol., pp. 4661-4672, Jun. (2021).