Improved Word Sense Disambiguation with Enhanced Sense Representations

Yang Song[1], **Xin Cai Ong**[2], **Hwee Tou Ng**[1,2], and **Qian Lin**[2]
[1]Integrative Sciences and Engineering Programme, NUS Graduate School
[2]Department of Computer Science, National University of Singapore
e0335980@u.nus.edu, xincai@u.nus.edu,
nght@comp.nus.edu.sg, qlin@u.nus.edu

Abstract

Current state-of-the-art supervised word sense disambiguation (WSD) systems (such as GlossBERT and bi-encoder model) yield surprisingly good results by purely leveraging pre-trained language models and short dictionary definitions (or glosses) of the different word senses. While concise and intuitive, the sense gloss is just one of many ways to provide information about word senses. In this paper, we focus on enhancing the sense representations via incorporating synonyms, example phrases or sentences showing usage of word senses, and sense gloss of hypernyms. We show that incorporating such additional information boosts the performance on WSD. With the proposed enhancements, our system achieves an F1 score of 82.0% on the standard benchmark test dataset of the English all-words WSD task, surpassing previous published scores on this benchmark dataset.

1 Introduction

Word sense disambiguation (WSD) refers to the task of automatically identifying the meaning of ambiguous words using computational methods. Given a word in context and a fixed inventory of senses, the system determines the correct word sense. For example, the noun "bank" means different things in "financial bank" and "bank of a river". Ambiguity is one of the central problems faced by natural language processing (NLP) tasks and WSD aims to resolve semantic ambiguity. It is commonly used to help downstream NLP tasks, such as machine translation (Chan et al., 2007; Neale et al., 2016) and information retrieval (Zhong and Ng, 2012).

Supervised WSD approaches typically frame the task as a multi-class classification problem with a fixed sense inventory for each word type. Traditionally, many well-performing methods use manually engineered features to train an independent classifier, or *word expert*, for every word type (Zhong and Ng, 2010; Melamud et al., 2016). Target senses are thus treated as discrete labels. Neural-based supervised methods were also explored, with a unified classifier that shares parameters across all polysemous words (Kågebäck and Salomonsson, 2016). However, they were not able to outperform the *word expert* supervised systems. More recently, the advent of large language models such as BERT (Devlin et al., 2019) has boosted the performance of these neural-based methods. Pre-trained on massive amounts of texts, the language models have a good sense of language context, inherently encoding word sense information. Using these models to generate contextualized word representations, a rapid slew of recent publications has continually redefined the state of the art.

In combination with language models, lexical resources have also been shown to be able to significantly improve WSD scores. Specifically, sense definitions (or glosses) have been used in recent work (Luo et al., 2018; Huang et al., 2019; Blevins and Zettlemoyer, 2020; Barba et al., 2021). In both GlossBERT (Huang et al., 2019) and the bi-encoder model (BEM) (Blevins and Zettlemoyer, 2020), good performance was achieved purely by utilizing the context sentence containing the ambiguous word and sense gloss information. In other words, the queried word sense is solely represented by a sense gloss that is typically less than twenty words. Given the brevity of information in a sense gloss, it is somewhat surprising that these architectures are able to achieve state-of-the-art performance.

In this paper, we show that enhancing the sense representations allows the pre-trained language models to better differentiate between the word senses by improving word sense clustering for each word type. We present a binary sentence pair classification model that is built upon RoBERTa (Liu et al., 2019), with focus on sense representation embellishment. We approach the task as a sentence pair classification problem, performing binary clas-

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 4311–4320
November 7–11, 2021.

sification on context-sense sentence pairs and training it in an end-to-end fashion.

To enhance word sense representation, we introduce a bag of "related" words that is associated with that particular word sense. These "related" words are intuitively chosen to provide more information about the word sense. Concretely, it is derived from synonyms, example phrases or sentences showing usage of word senses, and sense gloss of hypernyms. Incorporating these additional sources to enhance the sense representation improves the performance on the standard all-words English WSD evaluation benchmark. We achieve an F1 score of 82.0% on this benchmark test dataset, surpassing previous published scores on this test dataset.

In summary, the overall contributions of this paper include:

- We present an approach towards sentence-pair classification for WSD with improved performance over current implementations.
- We show that enhancing sense representations (ESR) is indeed able to boost performance on the all-words English WSD task.
- We examine and visualize the impact of additional lexical information on the sense representations with an ablation study, to investigate why our model performs better.

Our source code and trained models are available at `https://github.com/nusnlp/esr`.

2 Related Work

In this paper, we address the English all-words WSD task, where a system disambiguates every ambiguous word in the dataset (Palmer et al., 2001).

In general, supervised methods have been shown to perform better on the task, utilizing expensive human annotated data to achieve superior results. Combined with recent pre-trained language models, supervised neural architectures have gained popularity in recent years. For example, Hadiwinoto et al. (2019) investigates different ways of using pre-trained BERT to perform WSD, with the GLU model outperforming previous work.

While supervised methods traditionally do not leverage lexical resources such as WordNet (Miller, 1995), lexical information has proven to be useful in other methods. For example, the well-known Lesk algorithm (Lesk, 1986) shows that sense gloss is useful, with the algorithm picking the sense whose dictionary gloss shares the most words with the neighborhood of the ambiguous word. With pre-trained language models as feature extractors, sense gloss information can be incorporated into supervised WSD systems, generating significant performance boost. Two such examples are GlossBERT and BEM.

Similar to our work, GlossBERT (Huang et al., 2019) formulates the task as a sentence-pair classification problem – using context-gloss pairs to fine-tune the pre-trained BERT (Devlin et al., 2019) on the labeled SemCor data (Miller et al., 1994). This becomes a binary classification problem where the system predicts whether the ambiguous word matches the queried sense gloss in a single cross-encoder model. However, they use the default BERT architecture for sentence pair classification, applying affine transformation on the [CLS] token. This summarized word sense query makes it more challenging for the model to identify the ambiguous word. In comparison, our system provides additional information about the ambiguous word (on top of [CLS] token), with immediate improved performance.

The BEM model (Blevins and Zettlemoyer, 2020) further improves on this approach by using a bi-encoder approach that independently embeds the ambiguous word with its surrounding context and the sense gloss of each queried sense. Since they are jointly optimized in the same representation space, disambiguation is performed by finding the nearest sense embedding.

Unlike GlossBERT and BEM, ESCHER (Barba et al., 2021) also utilizes sense gloss, but formulates the task as a span extraction problem. The input is a sentence pair where the first sentence contains the context of the ambiguous word and the second sentence contains the concatenation of glosses from all candidate senses. The system is trained to find the text span corresponding to the correct sense.

Another challenge faced by supervised systems is the limited training data size. The work from Yap et al. (2020) utilizes usage examples from WordNet to generate more training data. In contrast, our system uses example sentences to improve sense representations instead.

Other approaches make use of relational information in the lexical knowledge graphs. For example, LMMS (Loureiro and Jorge, 2019) uses annotated data to generate sense embeddings us-

ing BERT. These embeddings are then propagated through the WordNet graph to infer senses that do not appear in SemCor. Similarly, ARES (Scarlini et al., 2020) also achieves full sense coverage but through extraction of relevant contexts. SparseLMMS (Berend, 2020) further makes the embeddings sparse through a dictionary matrix. Connections are made between each dimension of the sparse embeddings and human interpretable semantic content. EWISE (Kumar et al., 2019), on the other hand, learns sense embeddings by pre-training a gloss encoder with sense definitions and knowledge graph information. The learned sense gloss embeddings are then scored via dot product with a contextual vector to perform prediction. EWISER (Bevilacqua and Navigli, 2020) extends EWISE by injecting additional relational knowledge from the lexical knowledge graph via a simple sparse dot product operation with an adjacency matrix formulated with the knowledge graph. Since the pre-trained sense embeddings are used to classify the ambiguous word, the model is able to predict synsets that are not present in the training set, improving zero-shot performance. Our system surpasses previous published systems despite using minimal knowledge graph information (only the sense gloss of hypernyms).

3 Methodology

In this section, we describe the model architecture of our system, and present our method for achieving enhanced sense representations (ESR).

3.1 Model Architecture

The WSD task determines the best synset $\hat{s} \in \mathcal{S}_w$ for an ambiguous word w, where $\mathcal{S}_w$ is the set of candidate synsets for word w.

The inputs of our system are sentence pairs. The first sentence is the context containing the ambiguous word w, and the second sentence is the sense representation of one candidate synset $s \in \mathcal{S}_w$. The two sentences are then concatenated to form a sentence pair containing words $w_1, w_2, ..., w_m$, which will be tokenized into tokens $t_1, t_2, ..., t_n$. In the case of RoBERTa tokenizer, each tokenized sentence in the pair is surrounded by <s> and <\s>, so $t_1 =$ <s>. The tokens are then passed to RoBERTa $\mathbf{T}$, which will produce final layer hidden states:

$$\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n = \mathbf{T}(t_1, t_2, ..., t_n), \mathbf{h}_i \in \mathbb{R}^H$$

where H is the size of one hidden state. If a word w_i is tokenized into multiple tokens $t_j, ..., t_k$, then the average of the corresponding final layer hidden states is used:

$$\mathbf{h}_{w_i} = \frac{1}{k-j+1} \sum_{l=j}^{k} \mathbf{h}_l$$

Note that RoBERTa adds an extra layer with $\tanh$ activation on top of the final layer hidden state of the first token <s> to produce an output for classification tasks:

$$\mathbf{h}_s = \tanh(\mathbf{W}_s \mathbf{h}_1 + \mathbf{b}_s)$$

$$\mathbf{W}_s \in \mathbb{R}^{H \times H}, \mathbf{b}_s \in \mathbb{R}^H$$

This output $\mathbf{h}_s$ and the hidden state of the ambiguous word w are then concatenated and passed to a binary classification layer, whose output is passed to softmax to model the probability of a candidate synset to be positive:

$$\mathbf{o} = \mathbf{W}_{binary} \operatorname{concat}(\mathbf{h}_s, \mathbf{h}_w) + \mathbf{b}_{binary}$$

$$\mathbf{W}_{binary} \in \mathbb{R}^{2 \times 2H}, \mathbf{b}_{binary} \in \mathbb{R}^2$$

$$\mathbf{p} = \operatorname{softmax}(\mathbf{o})$$

Here we use $p_s = \mathbf{p}_2$ to model the probability of a candidate synset s to be positive.

During training, each sentence pair is assigned a label y, with y equals to 1 if the sentence pair contains a positive synset and 0 otherwise. Binary cross-entropy is used as our loss function:

$$\mathcal{L} = -y \log(p_s) - (1-y) \log(1-p_s)$$

During prediction, the synset with the highest probability among all the candidate synsets in $\mathcal{S}_w$ is used as the predicted synset of the ambiguous word w:

$$\hat{s} = \underset{s \in \mathcal{S}_w}{\operatorname{argmax}} \, p_s$$

where $\mathcal{S}_w$ is determined by the lemma and POS tag of the ambiguous word w.

3.2 Baseline System

We use RoBERTa as our transformer model. To better represent the context, we not only use the sentence S containing the ambiguous word, but also include one neighboring sentence before S and one neighboring sentence after S. For sense representation, we join the ambiguous word and the sense definition of the synset with a colon.

Context Sentence:
Has your attitude toward employee benefits encouraged an excess of free "government" work in your **plant**?
Positive Enhanced Sense Representation:
plant: buildings for carrying on industrial labor plant works industrial built large manufacture automobiles whole structure building made interconnected related structures
Gloss:
buildings for carrying on industrial labor
Synonyms:
plant, works, industrial plant
Example Phrases or Sentences:
they built a large plant to manufacture automobiles
Hypernym Gloss:
a whole structure (as a building) made up of interconnected or related structures
Related Words with Stop Words and Repeated Words Removed:
plant works industrial built large manufacture automobiles whole structure building made interconnected related structures
Negative Enhanced Sense Representation:
plant: (botany) a living organism lacking the power of locomotion plant flora life living thing develop ability act function independently
Gloss:
(botany) a living organism lacking the power of locomotion
Synonyms:
plant, flora, plant life
Example Phrases or Sentences: None
Hypernym Gloss:
a living thing that has (or can develop) the ability to act or function independently
Related Words with Stop Words and Repeated Words Removed:
plant flora life living thing develop ability act function independently

Table 1: An example of ESR for the word plant with one positive sense representation and one negative sense representation.

3.3 Enhanced Sense Representations

Built on top of the baseline system, ESR not only uses the sense definition of the synset, but also incorporates words related to the synset to enrich the sense representation.

The related words are constructed by first concatenating the words from the following three sources in order: (i) all the lemmas belonging to the synset (synonyms); (ii) WordNet example phrases or sentences of the synset; (iii) hypernym gloss of the synset. Table 1 shows an example for the word plant, with the words from synonyms, example sentences, and hypernym glosses listed accordingly. We then remove stop words (which are not so informative), and keep one occurrence of a word if it appears multiple times. By appending related words to the sense representation of a synset, we obtain enhanced sense representation. Table 1 gives examples of enhanced sense representations for the positive and negative synset of the word plant in the context sentence[1].

4 Experiments

In this section, we provide the details of our experiments and a comparison with other systems.

4.1 Datasets

We follow the unified evaluation framework for WSD (Raganato et al., 2017). The SemCor dataset for training contains 226,036 annotated instances from 37,176 sentences. By creating positive and negative examples for each instance, we gener-

[1]For brevity, the neighboring sentences of the context sentence are not shown in the table.

	Config			Dev	Test Datasets				Concatenation of all Datasets				
	M	F	G	SE07	SE2	SE3	SE13	SE15	N	V	A	R	ALL
Baseline Systems													
WordNet S1	-	-	-	55.2	66.8	66.2	63.0	67.8	67.6	50.3	74.3	80.9	65.2
Baseline	B	✓	-	74.6	79.8	76.8	78.9	81.8	81.7	67.9	81.8	86.9	78.8
Without WNGC													
EWISE	-	-	-	67.3	73.8	71.1	69.4	74.5	74.0	60.2	78.0	82.1	71.8
LMMS	L	-	-	68.1	76.3	75.6	75.1	77.0	-	-	-	-	75.4
SparseLMMS	L	-	-	68.8	77.9	77.8	76.1	77.5	-	-	-	-	76.8
GlossBERT	B	✓	-	72.5	77.7	75.2	76.1	80.4	79.8	67.1	79.6	87.4	77.0
ARES	L	-	-	71.0	78.0	77.1	77.3	83.2	80.6	68.3	80.5	83.5	77.9
EWISER	L	-	-	71.0	78.9	78.4	78.9	79.3	81.7	66.3	81.2	85.8	78.3
BEM	B	✓	-	74.5	79.4	77.4	79.7	81.7	81.4	68.5	83.0	87.9	79.0
Yap et al. (2020)	L	✓	-	72.7	79.8	77.8	79.7	84.4	82.6	68.5	82.1	86.4	79.5
ESR_{base}	B	✓	-	75.4	80.6	78.2	79.8	82.8	82.5	69.5	82.5	87.3	79.8
ESCHER	L	✓	-	76.3	81.7	77.8	82.2	83.2	83.9	69.3	83.8	86.7	80.7
ESR_{large}	L	✓	-	77.0	81.3	79.9	81.5	84.1	83.9	71.5	83.1	87.2	**81.1**
With WNGC													
SparseLMMS	L	-	✓	73.0	79.6	77.3	79.4	81.3	-	-	-	-	78.8
EWISER	L	-	✓	75.2	80.8	79.0	80.7	81.8	82.9	69.4	83.6	87.3	80.1
ESR_{base}	B	✓	✓	77.4	81.4	78.0	81.5	83.9	83.1	71.1	83.6	87.5	80.7
ESR_{large}	L	✓	✓	78.5	82.5	80.2	82.3	85.3	84.4	73.0	84.4	88.0	**82.0**

Table 2: F1 scores (%) of different WSD systems on the English all-words WSD task. **ALL** is the concatenation of all datasets, including **SE07**. **M** indicates whether the model used is base (**B**) or large (**L**). **F** indicates whether the model is fine-tuned with updated parameters. **G** indicates whether WNGC is used for training. With `roberta-large`, ESR trained on SemCor achieves F1 score of 81.1%, and ESR trained on SemCor and WNGC achieves F1 score of **82.0%**, outperforming prior published systems. All reported scores of ESR are the average scores over 3 runs with different random seeds.

ate 1,544,111 training examples. We also use the Princeton WordNet Gloss Corpus (WNGC) from UFSAC (Vial et al., 2018) for training. 496,776 annotated instances from 117,659 gloss sentences are used to generate 2,104,639 training examples. Similar to past work, we use SemEval-2007 (**SE07**) as development dataset and use Senseval-2 (**SE2**), Senseval-3 (**SE3**), SemEval-2013 (**SE13**), and SemEval-2015 (**SE15**) as test datasets.

In addition, we evaluate few-shot and zero-shot performance of ESR on the FEWS (Blevins et al., 2021) dataset. FEWS is generated from Wiktionary quotations and illustrations. It covers 71,391 senses from Wiktionary and contains a total of 121,459 ambiguous instances, which are divided into 101,459, 10,000, and 10,000 instances for training, development, and testing respectively. Each of the development set and test set contains 5,000 few-shot instances and 5,000 zero-shot instances. By creating positive and negative examples for each instance, we generate 478,604 training examples. Since the sense definitions and usage examples are put together in FEWS, we use `e.g.` as the delimiter to separate them for use with ESR.

4.2 Hyperparameters

We have two settings during training, one with `roberta-base` and the other with `roberta-large`. Both settings fine-tune the pre-trained language model from Hugging Face (Wolf et al., 2020) through 3 epochs with a total batch size of 32. The optimizer used is AdamW (Loshchilov and Hutter, 2019), with learning rate set to 8.5e-6, epsilon set to 1e-6, and weight decay set to 0. The warm up steps are 10% of the total training steps (batches). The number is 14,476 for fine-tuning on SemCor, 34,207 for fine-tuning on both SemCor and WNGC, and 4,487 for fine-tuning on FEWS. The input size (number of tokens n) is limited to 432 for `roberta-base` and 348 for `roberta-large`. During fine-tuning, the model is evaluated every 500 batches. After 1.5 epochs, the checkpoint with the highest SE07 F1 score is saved. If multiple checkpoints have the same SE07 F1 score, the earliest one is chosen to avoid over-fitting.

4.3 Results

In this subsection, we present the scores of ESR on the benchmark WSD evaluation framework and on FEWS.

4.3.1 WSD Evaluation Framework

Table 2 shows the F1 scores of different WSD systems on the English all-words WSD evaluation framework (Raganato et al., 2017). For each of

	Dev			Test		
	Full Set	**Few-shot**	**Zero-shot**	**Full Set**	**Few-shot**	**Zero-shot**
BEM_{BERT}	73.8	79.3	68.3	72.8	79.1	66.5
Human	80.1	80.4	79.9	-	-	-
ESR_{base}	75.9	77.9	73.9	**74.8**	77.8	71.6
$\text{ESR}_{\text{large}}$	80.5	83.8	77.1	**79.6**	83.4	75.8

Table 3: F1 scores (%) of different WSD systems on the FEWS dataset, trained on FEWS only. ESR with `roberta-base` outperforms BEM_{BERT} by 2.0% on the full test set. ESR with `roberta-large` obtains an even higher F1 score of 79.6% on the full test set, and outperforms human on the full dev set. All reported scores of ESR are the average scores over 3 runs with different random seeds.

our systems, we run the experiment 3 times with different random seeds and report the average score over 3 runs in the table.

By incorporating ESR, there is a significant improvement of 1.0% over the baseline system, from 78.8% to 79.8%. The improvement is statistically significant with p-value < 0.01, which shows that ESR is effective.

When training on SemCor only with `roberta-base`, ESR outperforms most prior published systems except ESCHER. However, ESCHER fine-tunes on a large model. The WSD system from Yap et al. (2020) performs close to ESR. However, the `bert-large-uncased` used in their system contains 336M parameters, almost 2.7 times the number of parameters compared to `robeta-base`, which has only 125M parameters. Note that the F1 scores for verbs are all below 70% and more than 10% lower than other POS tags in all previous WSD systems, dragging down the overall performance of the systems. The reason is that the synsets for verbs in WordNet are so fine-grained that it is often difficult for even humans to tell the difference. The performance of ESR on verbs beats all previous WSD systems, including those utilizing WNGC and a large model, which shows that ESR is effective in distinguishing fine-grained senses.

When training on SemCor only with `roberta-large`, ESR surpasses all previous WSD systems with an F1 score of 81.1% on **ALL**. By adding WNGC to the training data, ESR with `roberta-large` further improves to 73.0% on verbs, and achieves an F1 score of **82.0%** on **ALL**. The 0.9% improvement brought by WNGC is statistically significant with p-value < 0.01.

With `roberta-base`, the time taken for training on SemCor is 9 hours on 1 RTX 3090 GPU, and 18 hours for training on both SemCor and WNGC. With `roberta-large`, the time taken for training on SemCor is 8 hours on 2 A100 GPUs, and 17 hours for training on both SemCor and WNGC. Testing time is 0.25 hours for both.

4.3.2 FEWS

Table 3 shows the F1 scores of different WSD systems on FEWS development set and test set. All the systems are trained on the FEWS dataset only. We use BEM_{BERT} from Blevins et al. (2021) as baseline. Compared with the BEM baseline, ESR with `roberta-base` improves on the full test set by 2.0%, and improves on the zero-shot test set by 5.1%. When using `roberta-large`, ESR further improves the F1 score on the full test set to 79.6%. On the full development set, ESR even outperforms human, although its zero-shot performance is still worse than human.

The time taken for training on FEWS is 2 hours on 1 A100 GPU with `roberta-base`, and 3.5 hours on 2 A100 GPUs with `roberta-large`. Testing time is 0.35 hours for both.

5 Analysis

In this section, we will analyze the effectiveness of different components constituting the related words in ESR: synonyms in the synset, example phrases or sentences from WordNet, and sense definition of hypernym for the synset. We will then evaluate the less frequent sense and zero-shot performance of ESR. Finally, we will visualize how ESR separates different synsets of a word with an example, and show that ESR achieves better clustering.

5.1 Ablation Studies

In order to evaluate the effectiveness of different components constituting the related words in ESR, we remove each of them and see how the overall performance is affected.

ESR Ablation	ALL
ESR	79.8
$\text{ESR}_{\setminus\text{Synonyms}}$	79.3
$\text{ESR}_{\setminus\text{Synonyms \& Hypernym}}$	79.1
$\text{ESR}_{\setminus\text{Synonyms \& Examples}}$	78.9
Baseline	78.8

Table 4: F1-scores (%) of ablations on the **ALL** evaluation set by removing each component from related words. The components are: (i) synonyms in the synset; (ii) example phrases or sentences from WordNet; (iii) hypernym gloss of the synset.

			Zero-shot	
	MFS	**LFS**	**Senses**	**Words**
WordNet S1	100.0	0.0	53.9	84.9
Baseline	95.5	47.6	68.4	93.1
$\text{ESR}_{\setminus\text{Synonyms}}$	95.7	48.6	68.8	93.7
ESR	95.8	49.8	69.0	93.7

Table 5: F1-scores (%) on MFS, LFS, and zero-shot F1-scores on the **ALL** evaluation set.

Table 4 shows the F1 scores of different ablations. By removing synonyms from ESR, there is a significant drop of 0.5%, from 79.8% to 79.3%. If we further remove examples, there is a 0.4% drop from 79.3% to 78.9%. If we remove hypernyms instead after removing synonyms, the drop is reduced to 0.2%, from 79.3% to 79.1%. The above ablations show that synonyms play the most significant role in ESR, followed by examples, and hypernyms give the least contribution.

We can also view the results from another angle. By adding examples to the baseline system, there is a 0.3% increase from 78.8% to 79.1%, while adding hypernyms to the baseline system only increases F1 score by 0.1%, from 78.8% to 78.9%. If we add both examples and hypernyms to the baseline system, there is a 0.5% increase in F1 score, from 78.8% to 79.3%, the same increase as further adding synonyms. This again shows that adding synonyms is the most significant in ESR, and adding hypernyms is less significant than adding examples.

One explanation for the above observations is that the synonyms of a synset are semantically close to the synset and make the synset more distinguishable, compared to its examples and hypernym. Besides, the hypernym is shared by all its hyponyms, making it less unique to a specific synset.

5.2 Few-shot and Zero-shot Performance

We have shown the effectiveness of ESR over the baseline system, and synonyms play the most significant role. We further investigate ESR's effectiveness on the most frequent sense (MFS) and less frequent senses (LFS) of a word, where MFS is defined as the first and also the most common sense of a word in WordNet, and LFS is defined as the the other less frequent senses of a word. We also investigate the zero-shot performance of ESR, when it is tested on unseen senses and unseen words in the training data.

As shown in Table 5, both ESR and the baseline system perform better on MFS than on LFS. This is because SemCor is imbalanced and 73.7% of the training instances are MFS. The fewer training instances for LFS and the fine-grained nature of WordNet make it hard to distinguish the different synsets and achieve a high performance on LFS. However, ESR uses related words to make the synset more distinguishable, and improves by 1.0% over the baseline by using only examples and hypernym. If synonyms are used, a further 1.2% improvement is achieved.

Unseen senses are senses that do not appear in the SemCor training data, but appear in the test datasets. By adding examples and hypernyms, a 0.4% improvement can be made. After adding synonyms, a further 0.2% improvement can be made. To see why the performance on an unseen sense can be improved, consider the word `evoke`, where its sense `call to mind` in the **SE2** test set does not appear in SemCor. However, in the SemCor training data, the sense `call forth (emotions, feelings, and responses)` of `evoke` is present. During training, related words of the unseen sense `call to mind` are used as part of a negative sentence pair with a context sentence that contains the ambiguous word `evoke`. As such, even though the sense `call to mind` does not appear in the training data, the ESR system is (indirectly) aware of this unseen sense `call to mind`, via its related words in a negative sentence pair. In this way, ESR is able to leverage the negative sentence pairs so that it can better disambiguate the `call to mind` sense during testing, even though it is an unseen sense that does not appear in the training data at all.

Unseen words are those that appear in the test datasets, but do not appear at all in the SemCor

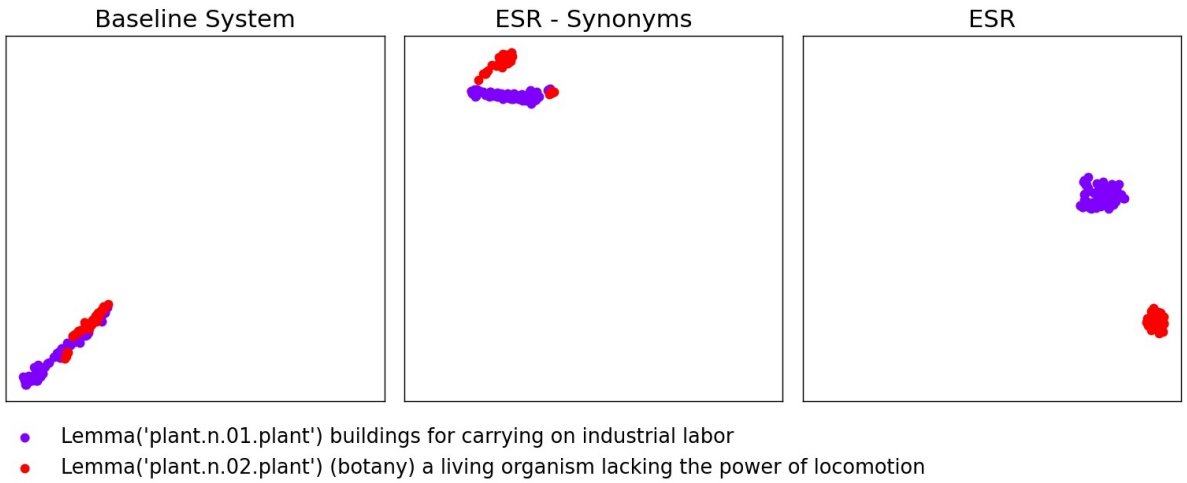

Figure 1: Visualization of `plant` in its two senses in different systems. Each point represents the concatenated hidden states of one positive sentence pair in SemCor. t-SNE is used for dimension reduction.

training data. However, by adding examples and hypernyms, ESR can improve the F1 score on unseen words by 0.6% over the baseline. Although unseen words do not appear as ambiguous words in SemCor, some of them actually show up in the sense representations of seen words. For example, although the word `envoy` in the **SE13** test set never appears as an ambiguous word in SemCor, it shows up in the sense gloss `provide or send (envoys or embassadors) with official credentials ...` of another seen word `accredit`. Hence, some of the unseen words are involved in the training process indirectly through the sense representations of seen words. This explains why ESR can improve the performance on unseen words.

5.3 ESR Improves Clustering

We have shown that ESR improves the performance of WSD by adding related words to make the sense representations more distinguishable through the above analysis. To further illustrate this fact, we evaluate the performance of the baseline system and ESR qualitatively through clustering.

For clustering, we use the concatenated hidden states of the first token and the ambiguous word in the context, which are the inputs of the binary classification layer as described in subsection 3.1. For each ambiguous word in SemCor, only the positive sentence pairs corresponding to its different senses are chosen. For visualization, the high dimensional concatenated hidden states are reduced to 2 dimensions with t-SNE.

Figure 1 shows the ambiguous word `plant` with its two senses in different systems. Each point represents a positive sentence pair in SemCor containing the sense representation of the ambiguous word `plant`. Although the two senses are distinctive, the baseline system cannot separate them well and the points of both senses are mixed together.

By adding examples and hypernyms, the system is able to separate the two different senses. In Tabel 6, the average distance between a point and the cluster centroid for the "building" sense is decreased from 4.04 to 3.12 as the points form better clusters. However, the separation is not perfect due to some outliers from the "botany" sense mixing with the cluster for the "building" sense, causing a decrease in distance from 6.96 to 4.19 between the two centroids compared to the baseline. From visualization, it is clear that ESR separates the points best among all the three systems. The points for each sense form circular clusters with decreased average distance between a point and the cluster centroid, and there are no outliers. The distance between the two clusters is 20.83, much larger than the other two systems and more than enough for separation. This is consistent with the ablation test conclusion that synonyms play a more significant role than examples and hypernyms.

6 Conclusion

In this paper, we present ESR which incorporates related words of a synset from its synonyms, usage examples, and sense definition of hypernym to further boost the performance on WSD over previous state-of-the-art systems. ESR provides more

	$\lvert\mathbf{c}_1 - \mathbf{c}_2\rvert$	$\overline{\lvert\mathbf{p}_1 - \mathbf{c}_1\rvert}$	$\overline{\lvert\mathbf{p}_2 - \mathbf{c}_2\rvert}$
Baseline	6.96	4.04	2.36
$\text{ESR}_{\setminus\text{Synonyms}}$	4.19	3.12	2.77
ESR	20.83	2.07	1.13

Table 6: Distance between the two cluster centroids, and the average distance between a point and the corresponding centroid in each cluster for the two senses of `plant` in different systems.

distinctive representations for senses, making the senses better separated from each other, and improves the performance of a baseline WSD system significantly. ESR not only brings improvements on less frequent senses, unseen senses, and unseen words, but also improves the overall performance and surpasses prior published scores with an F1 score of 82.0%.

While our work shows that ESR improves WSD performance, there is still room for improvement as we only explore limited methods to enhance sense representations. For future work, we believe there are potentially better ways to enrich sense representations and make them more distinguishable, further improving the performance of WSD systems.

Acknowledgements

The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg).

References

Edoardo Barba, Tommaso Pasini, and Roberto Navigli. 2021. ESC: Redesigning WSD with extractive sense comprehension. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4661–4672.

Gábor Berend. 2020. Sparsity makes sense: Word sense disambiguation using sparse contextualized word representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8498–8508.

Michele Bevilacqua and Roberto Navigli. 2020. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2854–2864.

Terra Blevins, Mandar Joshi, and Luke Zettlemoyer. 2021. FEWS: Large-scale, low-shot word sense disambiguation with the dictionary. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 455–465.

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Christian Hadiwinoto, Hwee Tou Ng, and Wee Chung Gan. 2019. Improved word sense disambiguation using pre-trained contextualized word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5297–5306.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3509–3514.

Mikael Kågebäck and Hans Salomonsson. 2016. Word sense disambiguation using a bidirectional LSTM. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon*, pages 51–56.

Sawan Kumar, Sharmistha Jat, Karan Saxena, and Partha Talukdar. 2019. Zero-shot word sense disambiguation using sense definition embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5670–5681.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC*, pages 24–26.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, arXiv:1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Daniel Loureiro and Alípio Jorge. 2019. Language modelling makes sense: Propagating representations through WordNet for full-coverage word sense disambiguation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691.

Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018. Incorporating glosses into neural word sense disambiguation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2473–2482.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, page 39–41.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification. In *Human Language Technology*, pages 240–243.

Steven Neale, Luís Gomes, Eneko Agirre, Oier Lopez de Lacalle, and António Branco. 2016. Word sense-aware machine translation: Including senses as contextual features for improved translation models. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 2777–2783.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pages 21–24.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 99–110.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3528–3539.

Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2018. UFSAC: Unification of sense annotated corpora and tools. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 1027–1034.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Boon Peng Yap, Andrew Koh, and Eng Siong Chng. 2020. Adapting BERT for word sense disambiguation with gloss selection objective and example sentences. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 41–46.

Zhi Zhong and Hwee Tou Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83.

Zhi Zhong and Hwee Tou Ng. 2012. Word sense disambiguation improves information retrieval. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 273–282.