Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation

Loïc Vial   Benjamin Lecouteux   Didier Schwab

Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France

`{loic.vial, benjamin.lecouteux, didier.schwab}@univ-grenoble-alpes.fr`

Abstract

In this article, we tackle the issue of the limited quantity of manually sense annotated corpora for the task of word sense disambiguation, by exploiting the semantic relationships between senses such as synonymy, hypernymy and hyponymy, in order to compress the sense vocabulary of Princeton WordNet, and thus reduce the number of different sense tags that must be observed to disambiguate all words of the lexical database. We propose two different methods that greatly reduce the size of neural WSD models, with the benefit of improving their coverage without additional training data, and without impacting their precision. In addition to our methods, we present a WSD system which relies on pre-trained BERT word vectors in order to achieve results that significantly outperforms the state of the art on all WSD evaluation tasks.

1 Introduction

Word Sense Disambiguation (WSD) is a task which aims to clarify a text by assigning to each of its words the most suitable sense labels, given a predefined sense inventory.

Various approaches have been proposed to achieve WSD: Knowledge-based methods rely on dictionaries, lexical databases, thesauri or knowledge graphs as primary resources, and use algorithms such as lexical similarity measures (Lesk, 1986) or graph-based measures (Moro et al., 2014). Supervised methods, on the other hand, exploit sense annotated corpora as training instances for a classifier such as SVM (Chan et al., 2007; Zhong and Ng, 2010), or more recently by a neural network (Kågebäck and Salomonsson, 2016). Finally, unsupervised methods automatically identify the different senses of words from unannotated or parallel corpora (e.g. Ide et al. (2002)).

Supervised methods are by far the most predominant as they generally offer the best results in evaluation campaigns (for instance (Navigli et al., 2007)). State of the art classifiers used to combine specific features such as the parts of speech and the lemmas of surrounding words (Zhong and Ng, 2010), but they are now replaced by neural networks which learn their own representation of words (Raganato et al., 2017b; Le et al., 2018).

One major bottleneck of supervised systems is the restricted quantity of manually sense annotated corpora: In the annotated corpus SemCor (Miller et al., 1993), the largest manually sense annotated corpus available, words are annotated with 33 760 different sense keys, which corresponds to only approximately 16% of the sense inventory of WordNet (Miller, 1995), the lexical database of reference widely used in WSD. Many works try to leverage this problem by creating new sense annotated corpora, either automatically (Pasini and Navigli, 2017), semi-automatically (Taghipour and Ng, 2015), or through crowdsourcing (Yuan et al., 2016).

In this work, the idea is to solve this issue by taking advantage of the semantic relationships between senses included in WordNet, such as the hypernymy, the hyponymy, the meronymy, the antonymy, etc. Our method is based on the observation that a sense and its closest related senses (its hypernym or its hyponyms for instance) all share a common idea or concept, and so a word can sometimes be disambiguated using only related concepts. Consequently, we do not need to know every sense of WordNet to disambiguate all words of WordNet.

For instance, let us consider the word "mouse" and two of its senses which are the *computer* mouse and the *animal* mouse. We only need to know the notions of "animal" and "electronic de-

vice" to distinguish them, and all notions that are more specialized such as "rodent" or "mammal" are therefore superfluous. By grouping them, we can benefit from all other instances of electronic devices or animals in a training corpus, even if they do not mention the word "mouse".

Contributions: In this paper, we hypothesize that only a subset of WordNet senses could be considered to disambiguate all words of the lexical database. Therefore, we propose two different methods for building this subset and we call them sense vocabulary compression methods. By using these techniques, we are able to greatly improve the coverage of supervised WSD systems, nearly eliminating the need for a backoff strategy that is currently used in most systems when dealing with a word which has never been observed in the training data. We evaluate our method on a state of the art WSD neural network, based on pretrained contextualized word vector representations, and we present results that significantly outperform the state of the art on every standard WSD evaluation task. Finally, we provide a documented tool for training and evaluating neural WSD models, as well as our best pretrained model in a dedicated GitHub repository[1].

2 Related Work

In WSD, several recent advances have been made in the creation of new neural architectures for supervised models and the integration of knowledge into these systems. Multiple works also exploit the idea of grouping together related senses. In this section, we give an overview of these works.

2.1 WSD Based on a Language Model

In this type of approach, that has been initiated by Yuan et al. (2016) and reimplemented by Le et al. (2018), the central component is a neural language model able to predict a word with consideration for the words surrounding it, thanks to a recurrent neural network trained on a massive quantity of unannotated data.

Once the language model is trained, it is used to produce sense vectors that result from averaging the word vectors predicted by the language model at all positions of words annotated with the given sense.

At test time, the language model is used to predict a vector according to the surrounding context,

---

and the sense closest to the predicted vector is assigned to each word.

These systems have the advantage of bypassing the problem of the lack of sense annotated data by concentrating the power of abstraction offered by recurrent neural networks on a good quality language model trained in an unsupervised manner. However, sense annotated corpora are still indispensable to contruct the sense vectors.

2.2 WSD Based on a Softmax Classifier

In these systems, the main neural network directly classifies and attributes a sense to each input word through a probability distribution computed by a softmax function. Sense annotations are simply seen as tags put on every word, like a POS-tagging task for instance.

We can distinguish two separate branches of these types of neural networks:

1. Those in which we have several distinct and token-specific neural networks (or classifiers) for every different word in the dictionary (Iacobacci et al., 2016; Kågebäck and Salomonsson, 2016), each of them being able to manage a particular word and its particular senses. For instance, one of the classifiers is specialized in choosing between the four possible senses of the noun "mouse". This type of approach is particularly fitted for the lexical sample tasks, where a small and finite set of very ambiguous words have to be sense annotated in several contexts, but it can also be used in all-words word sense disambiguation tasks.

2. Those in which we have a larger and general neural network that is able to manage all different words and assign a sense in the set of all existing sense in the dictionary used (Raganato et al., 2017b).

The advantage of the first branch of approaches is that in order to disambiguate a word, limiting our choice to one of its possible senses is computationally much easier than searching through all the senses of all words. To put things in perspective, the average number of senses of polysemous words in WordNet is approximately 3, whereas the total number of senses considering all words is 206 941.

The second approach, however, has an interesting property: all senses reside in the same vector space and hence share features in the hidden layers of the network. This allows the model to predict

an identical sense for two different words (i.e. synonyms), but it also offers the possibility to predict a sense for a word not present in the dictionary (e.g. neologism, spelling mistake...).

Finally, in two recent articles, Luo et al. (2018a) and Luo et al. (2018b) have proposed an improvement of these type of architectures, by computing an attention between the context of a target word and the gloss of its different senses. Thus, their work is one of the first to incorporate knowledge from WordNet into a WSD neural network.

2.3 Sense Clustering Methods

Several works exploit the idea of grouping together mutiple WordNet sense tags in order to create a coarser sense inventory which can potentially be more useful in some NLP tasks.

In the works of Ciaramita and Altun (2006), the authors propose a supervised system that learns and predicts "Supersense" tags, which belong to the set of the broad semantic categories of senses, organizing the sense inventory of WordNet. This tagset consists, in their work, of 26 categories for nouns (such as "food", "person" or "object"), and 15 categories for verbs (such as "emotion" or "weather"). By predicting supersense tags instead of the usual fine-grained sense tags of WordNet, the output vocabulary of their system is shrinked to only 41 different classes, and this leads to a small and easy-to-train model able to perform partial WSD, which could be useful and sufficient for other NLP tasks where the fine-grained distinction is not necessary.

In Izquierdo et al. (2007), the authors propose several methods for creating "Basic Level Concepts" (BLC), groups of related senses with a generally smaller size than supersenses, and which can be controlled by a threshold variable. Their methods rely on the semantic relationships between senses of WordNet, and, in the same way as Ciaramita and Altun (2006), they evaluated their clusters on a modified WSD task, where supersenses or BLC have to be predicted instead of the original sense tags from WordNet.

The main difference between our work and these works is that our end goal is to improve fine-grained WSD systems. Even though our methods generate clusters of related senses, we guarantee that two different senses of a lemma reside in two different clusters, so at the end, even if our supervised system produces a cluster tag for a target word, we are still able to find back the true sense tag, by simply keeping track of which sense key of its lemma belongs to the predicted group.

3 Sense Vocabulary Compression

Current state of the art supervised WSD systems such as Yuan et al. (2016), Raganato et al. (2017b), Luo et al. (2018a) and Le et al. (2018) are all confronted to the following issues:

1. Due to the small number of manually sense annotated corpora available, a target word may never be observed during the training, and therefore the system is not able to annotate it.
2. For the same reason, a word may have been observed, but not all of its senses. In this case the system is able to annotate the word, but if the expected sense has never been observed, the output will be wrong, regardless of the architecture of the supervised system.
3. Training a neural network to predict a tag which belongs to the set of all WordNet senses can become extremely slow and requires a lot of parameters with a large output vocabulary. And this vocabulary goes up to $206\,941$ if we consider all word-senses of WordNet.

In order to overcome all these issues, we propose a method for grouping together multiple sense tags that refer in fact to the same concept. In consequence, the output vocabulary decreases, the ability of the trained system to generalize improves, as well as its coverage.

3.1 From Senses to Synsets: A Vocabulary Compression Based on Synonymy

In the lexical database WordNet, senses are organized in sets of synonyms called synsets. A synset is technically a group of one or more word-senses that have the same definition and consequently the same meaning. For instance, the first senses of "eye", "optic" and "oculus" all refer to a common synset which definition is "the organ of sight".

Illustrated in Figure 1, the word-sense to synset mapping is hence a way of compressing the output vocabulary, and it is already applied in many works (Yuan et al., 2016; Le et al., 2018), while not being always explicitly stated. This method clearly helps to improve the coverage of supervised systems however. Indeed, if the verb "help" is observed in the annotated data in its first sense, the context surrounding the target word can be used to later annotate the verb "assist" or "aid" with the same valid synset tag.
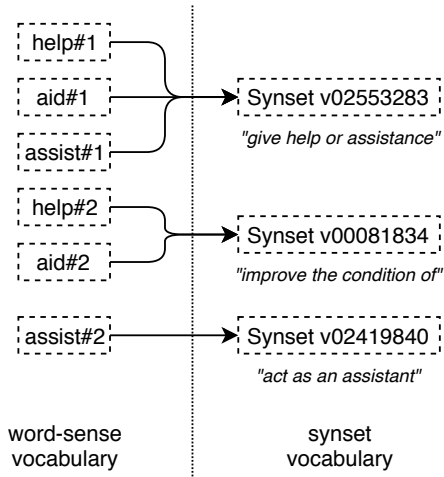
Figure 1: Word-sense to synset mapping (compression through synonymy) applied on the first two senses of the words "help", "aid" and "assist".

Figure 2: Sense vocabulary compression trough hypernymy hierarchy applied on the first and fourth sense of the word "mouse". Dashed arrows mean that some nodes are skipped for clarity.

Going further, other information from WordNet can help the system to generalize. Our first new method takes advantage of the hypernymy and hyponymy relationships to achieve the same idea.

3.2 Compression through Hypernymy and Hyponymy Relationships

According to Polguère (2003), hypernymy and hyponymy are two semantic relationships which correspond to a particular case of sense inclusion: the hyponym of a term is a specialization of this term, whereas its hypernym is a generalization. For instance, a "mouse" is a type of "rodent" which is in turn a type of "animal".

In WordNet, these relationships bind nearly every noun together in a tree structure[2] that goes from the generic root, the node "entity" to the most specific leaves, for instance the node "white-footed mouse". These relationships are also present on several verbs: for instance "add" is a way of "compute" which is a way of "reason".

For the sake of WSD, just like grouping together the senses of the same synset helps to better generalize, we hypothesize that grouping together the synsets of the same hypernymy relationship also helps in the same way. The general idea of our method is that the most specialized concepts in WordNet are often superfluous for WSD.

Indeed, considering a small subset of WordNet that only consists of the word "mouse", its first sense (the small rodent), its fourth sense (the elec-
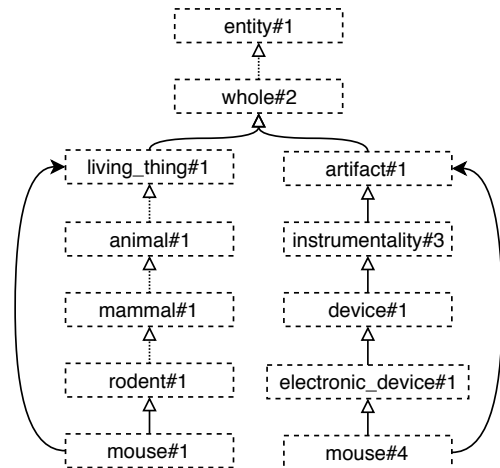
tronic device), and all of their hypernyms. This is illustrated in Figure 2. We can see that every concept that is more specialized than the concepts "artifact" and "living_thing" could be removed. We could map every tag of "mouse#1" to the tag of "living_thing#1" and we could still be able to disambiguate this word, but with a benefit: all other "living things" and animals in the sense annotated data could be tagged with the same sense. They would give examples of what is an animal and then show how to differentiate the small rodent from the hand-operated electronic device.

Therefore, the goal of our method is to map every sense of WordNet to its highest ancestor in the hypernymy hierarchy, but with the following constraints: First, this ancestor must discriminate all the different senses of the target word. Second, we need to preserve the hypernyms that are indispensable to discriminate the senses of the other words in the dictionary. For instance, we cannot map "mouse#1" to "living_thing#1", because the more specific tag "animal#1" is essential to distinguish the two senses of the word "prey" (one sense describes a person, the other describes an animal). Our method thus works in two steps:

1. We mark as "necessary" the children of the first common ancestor of every pair of senses of every word of WordNet.

2. We map every sense to its first ancestor in the hypernymy hierarchy that has been previously marked as "necessary".

---

[2]We computed that 41 607 on the 44 449 polysemous nouns of WordNet (94%) are part of this hierarchy.

As a result, the most specific synsets of the tree that are not indispensable for discriminating any word of the lexical inventory are automatically removed from the vocabulary. In other words, the set of synsets that is left in the vocabulary is the smallest subset of all synsets that are necessary to distinguish every sense of every word of WordNet, following the hypernym and hyponym links.

3.3 Compression through all semantic relationships

In addition to hypernymy and hyponymy, Word-Net contains several other relationships between synsets, such as the instance relationship (e.g. "Albert Einstein" is an instance of "physicist'), the meronymy (X is part of Y, or X is a member of Y) and its counterpart the holonymy, the antonymy (X is the opposite of Y), etc.

We hence propose a second method for sense vocabulary compression, that considers all the semantic relationships offered by WordNet, in order to form clusters of related synsets.

For instance, using all semantic relationships, we could form a cluster containing "physicist", "physics" (domain category), "Albert Einstein" (instance of), "astronomer" (hyponym), but also further related senses such as "photon", because it is a meronym of "radiation", which is a hyponym of "energy", which belongs to the same domain category of "physics".

Our method works by constructing these clusters iteratively: First, we initialize the set of clusters C with one synset in each cluster.

$$C = \{c_0, c_1, ..., c_n\} \qquad S = \{s_0, s_1, ..., s_n\}$$
$$C = \{\{s_0\}, \{s_1\}, ..., \{s_n\}\}$$

Then at each step, we sort C by sizes of clusters, and we peek the smallest one c_x and the smallest related cluster to c_x, c_y. We define a cluster being related to another if they contain at least one synset that have a semantic link together. We merge c_x and c_y together, and we verify that the operation still allows to discriminate the different senses of all words in the lexical database. If it is not the case, we cancel the merge and we try another semantic link. If no link is possible, we try to create one with the next smallest cluster, and if no further link can be created, the algorithm stops.

In Figure 3, we show a possible set of clusters that could result from our method, focusing on two senses of the word "Weber" and only on a few relationships.
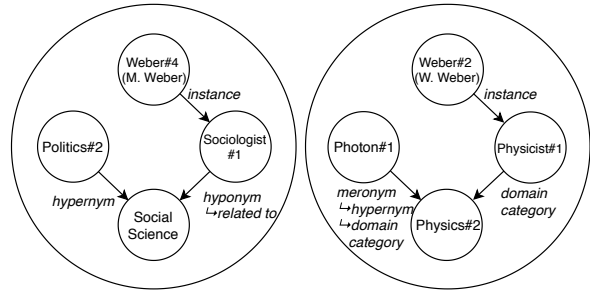

Figure 3: Example of clusters of sense that could result from our method, if we limit our view to two senses of the word "Weber" and only some relationship links.

This method produces clusters significantly larger than the method based on hypernyms. On average, a cluster has 5 senses with the hypernym method, whereas it has 17 senses with this method. This method, unlike the previous one, is also stochastic, because the formation of clusters depends on the underlying order of iteration when multiple clusters are the same size. However, because we always sort clusters by size before creating a link, we observed that the final vocabulary size (i.e. number of clusters) is always between 11 000 and 13 000. In the following, we consider a resulting mapping where the algorithm stopped after 105 774 steps.

Method	Vocabulary size	Compression rate	SemCor Coverage
No compression	206 941	0%	16%
Synonyms	117 659	43%	22%
Hypernyms	39 147	81%	32%
All relations	11 885	94%	39%

Table 1: Effects of the sense vocabulary compression on the vocabulary size and on the coverage of the SemCor.

In Table 1, we show the effect of the common compression through synonyms, our first proposed compression through hypernyms, and our second method of compression through all semantic relationships, on the size of the vocabulary of WordNet sense tags, and on the coverage of the SemCor corpus. As we can see, the sense vocabulary size is drastically decreased, and the coverage of the same corpus really improved.

4 Experiments

In order to evaluate our sense vocabulary compression methods, we applied them on a neural WSD system based on a softmax classifier capable of classifying a word in all possible synsets of WordNet (see subsection 2.2).

We implemented a system similar to Raganato et al. (2017b)'s BiLSTM but with some key differences. In particular, we used BERT contextualized word vectors (Devlin et al., 2018) in input of our network, Transformer encoder layers (Vaswani et al., 2017) instead of LSTM layers as hidden units, our output vocabulary only consists of sense tags seen during training (mapped according to the compression method used), and we ignore the network's predictions on words that are not annotated.

4.1 Implementation details

For BERT, we used the model named "bert-large-cased" of the PyTorch implementation[3], which consists of vectors of dimension 1024, trained on BooksCorpus and English Wikipedia.

Due to the fact that BERT's internal tokenizer sometimes split words in multiples tokens (i.e. ["rodent"] becomes ["rode", "##nt"]), we trained our system to predict a sense tag on the first token only of a splitted annotated word.

For the Transformer encoder layers, we used the same parameters as the "base" model of Vaswani et al. (2017), that is 6 layers with 8 attention heads, a hidden size of 2048, and a dropout of 0.1.

Finally, because BERT already encodes the position of the words inside their vectors, we did not add any positional encoding.

4.2 Training

We compared our sense vocabulary compression methods on two training sets: The SemCor, and the concatenation of the SemCor and the Princeton WordNet Gloss Corpus (WNGC). The latter is a corpus distributed as part of WordNet since its version 3.0, and it consists of the definitions (glosses) of every synset of WordNet, with words manually or semi-automatically sense annotated. We used the version of these corpora given as part of the UFSAC 2.1 resource[4] (Vial et al., 2018).

We performed every training for 20 epochs. At the beginning of each epoch, we shuffled the training set. We evaluated our model at the end of every epoch on a development set, and we kept only the one which obtained the best F1 WSD score. The development set was composed of 4 000 random sentences taken from the Princeton WordNet Gloss Corpus for the models trained on the SemCor, and 4 000 random sentences extracted from the whole training set for the other models.

For each training set, we trained three systems:

1. A "baseline" system that predicts a tag belonging to all the synset tags seen during training, thus using the common vocabulary compression through synonyms method.
2. A "hypernyms" system which applies our vocabulary compression through hypernyms algorithm on the training corpus.
3. A "all relations" system which applies our second vocabulary compression through all relations on the training corpus.

We trained with mini-batches of 100 sentences, truncated to 80 words, and we used Adam (Kingma and Ba, 2015) with a learning rate of 0.0001 as the optimization method.

System	SemCor	SemCor+WNGC
baseline	77.15M	120.85M
hypernyms	63.44M	79.85M
all relations	55.16M	60.27M

Table 2: Number of parameters of neural models.

All models have been trained on one Nvidia's Titan X GPU. The number of parameters of individual models are displayed in Table 2. As we can see, our compression methods drastically reduce the number of parameters, by a factor of 1.2 to 2.

4.3 Evaluation

We evaluated our models on all evaluation corpora commonly used in WSD, that is the English all-words WSD tasks of the evaluation campaigns SensEval/SemEval. We used the fine-grained evaluation corpora from the evaluation framework of Raganato et al. (2017a), which consists of SensEval 2 (Edmonds and Cotton, 2001), SensEval 3 (Snyder and Palmer, 2004), SemEval 2007 task 17 (Pradhan et al., 2007), SemEval 2013 task 12 (Navigli et al., 2013) and SemEval 2015 task 13 (Moro and Navigli, 2015), as well as the "ALL" corpus consisting of the concatenation of all pre-

---

System	SE2	SE3	SE07 17	SE13	SE15	ALL (concat. of previous tasks)					SE07 07
						nouns	verbs	adj.	adv.	total	
First sense baseline	65.6	66.0	54.5	63.8	67.1	67.7	49.8	73.1	80.5	65.5	78.9
HCAN (Luo et al., 2018a)	72.8	70.3	-	68.5	72.8	72.7	58.2	77.4	84.1	71.1	-
LSTMLP (Yuan et al., 2016)	73.8	71.8	63.5	69.5	72.6	†73.9	-	-	-	†71.5	83.6
SemCor, baseline	77.2	76.5	70.1	74.7	77.4	78.7	65.2	79.1	85.5	76.0	87.7
SemCor, hypernyms	77.5	77.4	69.5	76.0	78.3	79.6	65.9	79.5	85.5	76.7	87.6
SemCor, all relations	76.6	76.9	69.0	73.8	75.4	77.2	66.0	80.1	85.0	75.4	86.7
SemCor+WNGC, baseline	**79.7**	76.1	**74.1**	78.6	80.4	80.6	68.1	82.4	86.1	78.3	90.4
SemCor+WNGC, hypernyms	**79.7**	77.8	73.4	**78.7**	**82.6**	81.4	68.7	83.7	85.5	**79.0**	90.4
SemCor+WNGC, all relations	79.4	**78.1**	71.4	77.8	81.4	80.7	68.6	82.8	85.5	78.5	**90.6**

Table 3: F1 scores (%) on the English WSD tasks of the evaluation campaigns SensEval/SemEval. The task "ALL" is the concatenation of SE2, SE3, SE07 17, SE13 and SE15. The first sense is assigned on words for which none of its sense has been observed during the training. Results in **bold** are to our knowledge the best results obtained on the task. Scores prefixed by a dagger (†) are not provided by the authors but are deduced from their other scores.

vious ones. We also compared our result on the coarse-grained task 7 of SemEval 2007 (Navigli et al., 2007) which is not present in this framework.

For each evaluation, we trained 8 independent models, and we give the score obtained by an ensemble system that averages their predictions through a geometric mean.

System	No Backoff	Backoff on Monosemics
SemCor, baseline	93.23%	98.13%
SemCor, hypernyms	98.75%	99.68%
SemCor, all relations	99.67%	99.99%
SemCor+WNGC, baseline	98.26%	99.41%
SemCor+WNGC, hypernyms	99.83%	99.96%
SemCor+WNGC, all relations	99.99%	100%

Table 4: Coverage of our systems on the task "ALL". "Backoff on Monosemics" means that monosemic words are considered annotated.

In the results in Table 3, we first observe that our systems that use the sense vocabulary compression through hypernyms or through all relations obtain scores that are overall equivalent to the systems that do not use it.

Our methods greatly improves their coverage on the evaluation tasks however. As we can see in Table 4, on the total of 7 253 words to annotate for the corpus "ALL", the baseline system trained on the SemCor is not able to annotate 491 of them, while the vocabulary compression through hypernyms reduces this number to 91 and 24 for the

compression through all relations.

When adding the Princeton WordNet Gloss Corpus to the training set, only one word (the monosemic adjective "cytotoxic") cannot be annotated with the system that uses the compression through all relations because its sense has not been observed during training.

If we exclude the monosemic words, the system based on our compression method through all relations miss only one word (the adverb "eloquently") when trained on the SemCor, and has a coverage to 100% when the WNGC is addded.

In comparison to the other works, thanks to the Princeton WordNet Gloss Corpus added to the training data and the use of BERT as input embeddings, we outperform systematically the state of the art on every task.

4.4 Ablation Study

In order to give a better understanding of the origin of our scores, we provide a study of the impact of our main parameters on the results. In addition to the training corpus and the vocabulary compression method, we chose two parameters that differentiate us from the state of the art: the pre-trained word embeddings model and the ensembling method, and we have made them vary.

For the word embeddings model, we experimented with BERT (Devlin et al., 2018) as in our main results, with ELMo (Peters et al., 2018), and with GloVe (Pennington et al., 2014), the same pre-trained word embeddings used by Luo et al. (2018a). For ELMo, we used the model trained on

Training Corpus	Input Embeddings	Ensemble	F1 Score on task "ALL" (%)					
			Baseline		Hypernyms		All relations	
			$\bar{x}$	σ	$\bar{x}$	σ	$\bar{x}$	σ
SemCor+WNGC	BERT	Yes	78.27	-	79.00	-	78.48	-
SemCor+WNGC	BERT	No	76.97	±0.38	77.08	±0.17	76.52	±0.36
SemCor+WNGC	ELMo	Yes	75.16	-	74.65	-	70.58	-
SemCor+WNGC	ELMo	No	74.56	±0.27	74.36	±0.27	68.77	±0.30
SemCor+WNGC	GloVe	Yes	72.23	-	72.74	-	71.42	-
SemCor+WNGC	GloVe	No	71.93	±0.35	71.79	±0.29	69.60	±0.32
SemCor	BERT	Yes	76.02	-	76.73	-	75.40	-
SemCor	BERT	No	75.06	±0.26	75.59	±0.16	73.91	±0.33
SemCor	ELMo	Yes	72.55	-	73.09	-	69.43	-
SemCor	ELMo	No	72.21	±0.13	72.83	±0.24	68.74	±0.29
SemCor	GloVe	Yes	70.77	-	71.18	-	68.44	-
SemCor	GloVe	No	70.51	±0.16	70.77	±0.21	67.48	±0.55
HCAN (Luo et al., 2018a) (fully reproducible state of the art)								
SemCor+WordNet glosses	GloVe	No	71.1					
LSTMLP (Yuan et al., 2016) (state of the art scores but use private data)								
SemCor+1K (private)	private	No	71.5					

Table 5: Ablation study on the task "ALL" (i.e. the concatenation of all SensEval/SemEval tasks). For systems that do not use ensemble, we display the mean score ($\bar{x}$) of eight individually trained models along with its standard deviation (σ).

Wikipedia and the monolingual news crawl data from WMT 2008-2012.[5] For GloVe, we used the model trained on Wikipedia 2014 and Gigaword 5.[6] Due to the fact that GloVe embeddings do not encode the position of the words (a word has the same vector representation in any context), we used bidirectional LSTM cells of size 1 000 for each direction, instead of Transformer encoders for this set of experiments. In addition, because the vocabulary of GloVe is finite and all words are lowercased, we lowercased the inputs, and we assigned a vector filled with zeros to out-of-vocabulary words.

For the ensembling method, we either perform ensembling as in our main results, by averaging the prediction of 8 models trained separately or we give the mean and the standard deviation of the scores of the 8 models evaluated separately.

As we can see in Table 5, the additional training corpus (WNGC) and even more the use of BERT as input embeddings both have a major impact on our results and lead to scores above the state of the art. Using BERT instead of ELMo or GloVe improves respectively the score by approximately 3 and 5 points in every experiment, and adding the WNGC to the training data improves it by approximately 2 points. Finally, using ensembles adds roughly another 1 point to the final F1 score.

Finally, through the scores obtained by invidual models (without ensemble), we can observe on the standard deviations that the vocabulary compression method through hypernyms never impact significantly the final score. However, the compression method through all relations seems to negatively impact the results in some cases (when using ELMo or GloVe especially).

5 Conclusion

In this paper, we presented two new methods that improve the coverage and the capacity of generalization of supervised WSD systems, by narrowing down the number of different sense in WordNet in order to keep only the senses that are essential for differentiating the meaning of all words of the lexical database. On the scale of the whole lexical database, we showed that these methods can shrink the total number of different sense tags in WordNet to only 6% of the original size, and that the coverage of an identical training corpus has more than doubled. We implemented a state of the art WSD neural network and we showed that these methods compress the size of the underlying models by a factor of 1.2 to 2, and greatly improve their coverage on the evaluation tasks. As a result, we reach a coverage of 99.99% of the evaluation tasks (1 word missing on 7 253) when training a system on the SemCor only, and 100% when adding the WNGC to the training data, on the pol-

ysemic words. Therefore, the need for a backoff strategy is nearly eliminated. Finally, our method combined with the recent advances in contextualized word embeddings and with a training corpus composed of sense annotated glosses, our system achieves scores that considerably outperform the state of the art on all WSD evaluation tasks.

References

[Chan et al.2007] Yee Seng Chan, Hwee Tou Ng, and Zhi Zhong. 2007. Nus-pt: Exploiting parallel texts for word sense disambiguation in the english all-words tasks. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 253–256, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Ciaramita and Altun2006] Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 594–602, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Devlin et al.2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

[Edmonds and Cotton2001] Philip Edmonds and Scott Cotton. 2001. Senseval-2: Overview. In *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, SENSEVAL '01, pages 1–5, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Iacobacci et al.2016] Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 897–907, Berlin, Germany, August. Association for Computational Linguistics.

[Ide et al.2002] Nancy Ide, Tomaz Erjavec, and Dan Tufis. 2002. Sense discrimination with parallel corpora. In *Proceedings of the ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pages 61–66. Association for Computational Linguistics, July.

[Izquierdo et al.2007] Rubén Izquierdo, Armando Suárez, and German Rigau. 2007. Exploring the automatic selection of basic level concepts. In *Proceedings of RANLP*, volume 7. Citeseer.

[Kågebäck and Salomonsson2016] Mikael Kågebäck and Hans Salomonsson. 2016. Word sense disambiguation using a bidirectional lstm. In *5th Workshop on Cognitive Aspects of the Lexicon (CogALex)*. Association for Computational Linguistics.

[Kingma and Ba2015] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*.

[Le et al.2018] Minh Le, Marten Postma, Jacopo Urbani, and Piek Vossen. 2018. A deep dive into word sense disambiguation with lstm. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 354–365. Association for Computational Linguistics.

[Lesk1986] Michael Lesk. 1986. Automatic sense disambiguation using mrd: how to tell a pine cone from an ice cream cone. In *Proceedings of SIGDOC '86*, pages 24–26, New York, NY, USA. ACM.

[Luo et al.2018a] Fuli Luo, Tianyu Liu, Zexue He, Qiaolin Xia, Zhifang Sui, and Baobao Chang. 2018a. Leveraging gloss knowledge in neural word sense disambiguation by hierarchical co-attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1402–1411. Association for Computational Linguistics.

[Luo et al.2018b] Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018b. Incorporating glosses into neural word sense disambiguation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2473–2482. Association for Computational Linguistics.

[Miller et al.1993] George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the workshop on Human Language Technology*, HLT '93, pages 303–308, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Miller1995] George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

[Moro and Navigli2015] Andrea Moro and Roberto Navigli. 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 288–297, Denver, Colorado, June. Association for Computational Linguistics.

[Moro et al.2014] Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244.

[Navigli et al.2007] Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. 2007. Semeval-2007 task 07: Coarse-grained english all-words task. In *SemEval-2007*, pages 30–35, Prague, Czech Republic, June.

[Navigli et al.2013] Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 222–231.

[Pasini and Navigli2017] Tommaso Pasini and Roberto Navigli. 2017. Train-o-matic: Large-scale supervised word sense disambiguation in multiple languages without manual training data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 78–88. Association for Computational Linguistics.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

[Peters et al.2018] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

[Polguère2003] Alain Polguère. 2003. *Lexicologie et sémantique lexicale*. Les Presses de l'Université de Montréal.

[Pradhan et al.2007] Sameer S. Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task 17: English lexical sample, srl and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 87–92, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Raganato et al.2017a] Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017a. 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: Volume 1, Long Papers*, pages 99–110, Valencia, Spain, April. Association for Computational Linguistics.

[Raganato et al.2017b] Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017b. Neural sequence learning models for word sense disambiguation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1167–1178. Association for Computational Linguistics.

[Snyder and Palmer2004] Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*.

[Taghipour and Ng2015] Kaveh Taghipour and Hwee Tou Ng. 2015. One Million Sense-Tagged Instances for Word Sense Disambiguation and Induction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 338–344, Beijing, China, July. Association for Computational Linguistics.

[Vaswani et al.2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

[Vial et al.2018] Loïc Vial, Benjamin Lecouteux, and Didier Schwab. 2018. UFSAC: Unification of Sense Annotated Corpora and Tools. In *Language Resources and Evaluation Conference (LREC)*, Miyazaki, Japan, May.

[Yuan et al.2016] Dayu Yuan, Julian Richardson, Ryan Doherty, Colin Evans, and Eric Altendorf. 2016. Semi-supervised word sense disambiguation with neural models. In *COLING 2016*.

[Zhong and Ng2010] 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*, ACLDemos '10, pages 78–83, Stroudsburg, PA, USA. Association for Computational Linguistics.