CoKE : Word Sense Induction Using Contextualized Knowledge Embeddings

Sanjana Ramprasad
Mya Systems
sanjana.ramprasad@hiremya.com

James Maddox
Mya Systems
james.maddox@hiremya.com

Abstract

Word Embeddings can capture lexico-semantic information but remain flawed in their inability to assign unique representations to different senses of polysemous words. They also fail to include information from well-curated semantic lexicons and dictionaries. Previous approaches that obtain ontologically grounded word-sense representations learn embeddings that are superior in understanding contextual similarity but are outperformed on several word relatedness tasks by single prototype words. In this work, we introduce a new approach that can induce polysemy to any pre-defined embedding space by jointly grounding contextualized sense representations learned from sense-tagged corpora and word embeddings to a knowledge base. The advantage of this method is that it allows integrating ontological information while also readily inducing polysemy to pre-defined embedding spaces without the need for re-training. We evaluate our vectors on several word similarity and relatedness tasks, along with two extrinsic tasks and find that it consistently outperforms current state-of-the-art.

Introduction

Distributed representations of words (Mikolov et al. 2013b) has proven to be successful in addressing various drawbacks of symbolic representations which treat words as atomic units of meaning. By grouping similar words and capturing analogical and lexical relationships, they are a popular choice in several downstream NLP applications.

While these embeddings capture meaningful lexical relationships, they come with their own set of drawbacks. For instance, complete reliance on natural language corpora amplifies existing vocabulary bias that is inherent in datasets. Vocabulary bias is caused by words not seen in the training corpora and also extends to bias in word usage where some words, often morphologically complex words, are used less frequently than other words or phrases with the same meaning. Thus embeddings suffer from inaccurate modeling of less frequent words which is evident in the relatively lower performance of word embeddings on the rare word similarity task (Luong, Socher, and Manning 2013b). An approach by (Bojanowski et al. 2016a) propose using character n-gram representations to address the problem of out-of-vocabulary and rare words. (Faruqui et al. 2014) also proposed retrofitting vectors to an ontology to deal with inaccurate modeling of less frequent words. However, these methods don't account for polysemy.

Polysemy is an important feature of language which causes words to have a different meaning or "sense" based on the context in which they occur. For instance, the word *bank* can refer to a *financial institution* or *land on either side of a river*. A large body of work has gone into developing word sense disambiguation systems to identify the correct sense of a word based on its context. Word embeddings, on the other hand, assign a single vector representation to a word type, irrespective of polysemy. The availability of disambiguation systems coupled with the growing reliance of NLP systems on distributional semantics has led to an increasing interest in obtaining powerful sense representations.

Some of the previous work that has gone into learning sense representations includes unsupervised learning techniques to cluster contexts and learn multi prototype vectors((Reisinger and Mooney 2010) , (Huang et al. 2012) and (Wu and Giles 2015)). A common drawback with the cluster based approach is the difficulty in deciding the number of clusters apriori. ((Neelakantan et al. 2015) , (Tian et al. 2014) ,(Cheng and Kartsaklis 2015)) also learn multiple word embeddings by modifying the Skip-Gram model. These approaches yield to sense representations that are limited in terms of interpretability which makes it challenging to include in downstream tasks. To remedy this, (Iacobacci, Pilehvar, and Navigli 2015), (Chen, Liu, and Sun 2014) use sense-tagged corpora and Word2Vec modifications to obtain sense representations; however, they only make use of distributional semantics.

Previous work combining distributional semantics and knowledge bases include (Jauhar, Dyer, and Hovy 2015) and (Rothe and Schütze 2015) that grounding word embeddings to ontologies to obtain sense representations. As a result of grounding, these techniques drastically improved performance on several similarity tasks but an observed pattern is that this leads to compromised performance on word relatedness tasks((Faruqui et al. 2014), (Jauhar, Dyer, and Hovy

2015)).

In this work, we present a novel approach that uses knowledge bases and sense representations to directly induce polysemy to any pre-defined word embedding space. Our approach leads to interpretable, ontologically grounded sense representations that can easily be used with powerful disambiguation systems. The main contributions of this paper are a) Obtaining ontologically grounded sense representations that perform well on both similarity and relatedness tasks b) Automatic sense induction and integration of knowledge base information into any predefined embedding space without re-training c) Our embeddings also show performance benefits when used with transfer learning methods like CoVE (McCann et al. 2017) and ELMo (Peters et al. 2018) on extrinsic tasks. d) Furthermore, we propose methodologies for knowledge base augmentation along with an approach to learn more effective sense representations.

Methodology

In our approach we thus rely on a) Sense tagged corpora to obtain contextualized sense representations. The objective of which is to capture sense relations and interactions in naturally occurring corpora. The sense representations are interpretable and have lexical mappings to a knowledge base. We use them to induce polysemy in word embedding spaces. b) Pretrained word embeddings to capture beneficial lexical relationships that are inherent on account of being trained on large amounts of data. Sense representations do not adequately capture these relationships due to the limited size of sense-tagged corpora which is used to train them. c) Lastly, to account for the vocabulary bias in corpora which causes similar meaning words to be farther apart in embedding spaces, we use a knowledge base to jointly ground word and sense representations.

We thus describe our approach in three parts **a) Lexicon building b) Sense-Form Representations** and **c) Multi Word-Sense Representations**

a) Lexicon Building

For our Knowledge Base, we rely on WordNet (Miller 1995) and a Thesaurus[1]. WordNet(WN) is a large lexical database that groups synonyms to *synsets* and records relations between them in the form of synonyms, hypernyms, and hyponyms. The synsets are highly interpretable since they come with a gloss along with examples. A thesaurus, on the other hand, groups words into different clusters based on similarity of meaning.

Thesaurus Inclusion The structure of WordNet(WN) is such that it labels semantic relations among different synsets. While this structure helps determine the degree of similarity between synsets, it leads to a restricted set of synonyms that represent a *synset*. To best combine information from both resources, we augment the synonyms in a WordNet *synset* using a Thesaurus.

[1] https://www.thesaurus.com/

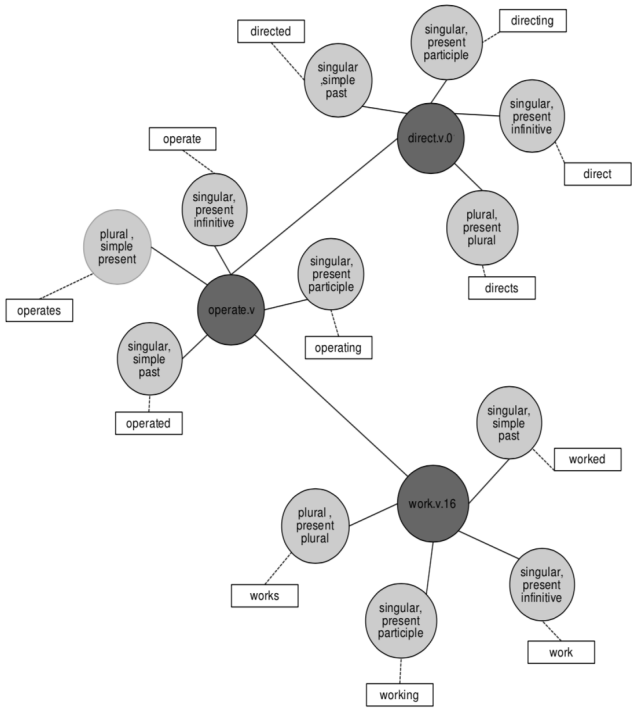

Figure 1: WordNet synset nodes split based on syntactic form information

Unlike WordNet(WN), the thesaurus does not have distinct labels for senses. Senses are instead represented by a group of words. Given a query word, the thesaurus returns clusters of words where each cluster represents some sense. Given a WN synset(s), we use the synset's headword to query the thesaurus and use a simple algorithm to map the most appropriate cluster to the corresponding WN synset by computing each cluster's probability with respect to (s).

Probabilities are assigned based on the words in a cluster and the WN structure. Thus if a thesaurus cluster has more words that are "closer" based on WN structure to the synset(s), it receives a higher probability. To measure "closeness", we use the path-similarity(p) metric of WN. Path-similarity(p) measures the similarity between two synsets by considering the distance between them. It ranges from $0-1$ with scores towards 1 denoting "closer" synsets. Since path-similarity(p) calculates similarity between two synsets, thus given a word (w) in a thesaurus cluster queried using the headword of the WN synset(s), we find the distance-based similarity $d_{w,s}$ between s and w by first obtaining all of the synsets(S_w) in WN for w and use it to calculate $d_{w,s}$ as follows.

$$d_{w,s} \leftarrow max\{p(s, s_i) \forall s_i \in S_w\}$$

If a word is not found in WN, we assign $d_{w,s}$ to 0.1 which is the lowest distance-based similarity implying it is "farthest" from the synset(s) in WN.

To account for varying cluster sizes in the thesaurus

Algorithm 1 Thesaurus Inclusion

Input: WordNet Synset (s), corresponding synonym set(S_w)
Output: Most probable cluster for a word C_{w_n} out of all possible clusters C_w found in Thesaurus for a word.

1: $C_w \leftarrow$ Thesaurus(w)
2: **if** length(C_w) = 1
3: n = 0
4: **else**
5: $p_c(w) \leftarrow \{p(cluster) \forall cluster \in C_w\}$
6: n $\leftarrow$ index($p_c(w)$, max($p_c(w)$))
7: **end if**
9: **return** C_{w_n}

and prevent larger clusters from invariably having bigger scores, we divide words in each cluster(c) into ten discrete bins($bins$) based on each word's d score. The bins are in an incremental range of 0.1((*[0-0.1 , 0.11- 0.2 ,...,0.91-1.0]*), with the highest score bin being 1. We then obtain cluster scores, $score_{cluster}$ as :

$$score_{cluster} = \sum_{bin \in bins} w_{bin} * count(bin)$$

We then get the probability of a cluster($p_{cluster}$) from $score_{cluster}$ by passing it through a sigmoid function.

$$p_{cluster} = \frac{exp(score_{cluster})}{exp(score_{cluster}) + 1}$$

The words in the thesaurus cluster with the highest probability is then picked to augmented into the synonym list of the respective WN synset(s) . We've outlined the procedure in Algorithm 1.

In the Table 1 we denote the vocabulary and synset cluster changes brought about by this step. The last column records the average number of synonyms linked with a synset in WordNet. Originally, owing to WordNet's stringent relation structure we see there are an average of approximately 2 synonyms within a synset. This number drastically increases using a thesaurus for augmentation.

	Words	Phrases	Average synonyms(per synset)
WordNet	147307	69408	1.75
Thesaurus(Introduced)	4026	500	7.37

Table 1: Vocabulary and synset cluster changes in WordNet through Thesaurus Inclusion.

WordNet Form Extension To obtain representations that cater to both similarity and relatedness, we modify the synset nodes in WordNet. A synset in WordNet is represented by a set of synonyms. We observe that these synonym sets include words of the same meaning without differentiating between their syntactic forms. For instance, consider the synset *operate.v.01*, defined as "direct or control; projects, businesses" , it has both *run* and *running* in its synonym sets. In practice, each syntactic form of a word has different semantic distributions. For instance, for this sense, *run* is found to most likely occur with words such as *lead* and *head* as compared to its alternate form *running* which is more likely to appear with words such as *managing*, *administrating* , *leading*. To account for this difference in semantics, we extend WordNet nodes to include the syntactic form information and call a synset, syntactic form pair "sense-form." To obtain different sense-form nodes, we make use of the OMSTI corpus and record different forms of a synset based on the different syntactic forms of words associated with the synset. Each "sense-form" is then linked to the corresponding syntactic form of synonyms. The extended WordNet(Ext-WN) sense-form nodes and synonyms are depicted in Figure 1.

b) Sense-Form Representations

To obtain sense-form representations, we use a sense-tagged corpus, OMSTI(Taghipour and Ng 2015). The corpus contains sense-tagged words based on WordNet. Each sense-tagged word is associated with the respective synset found in WN. We pre-process the corpora by replacing every word and synset pair as a sense-form based on the syntactic form of the tagged word and the synset. We then use the Word2Vec toolkit((Mikolov et al. 2013b)) with the Skip Gram objective function and Negative Sampling to obtain our contextualized "sense-form" representations.

c) Word-Sense Representation and Induction

We initialise each sense-form node in WN using the representations obtained from the sense-tagged corpora. Then, for each sense-form and the respective augmented synonym set, we obtain unique multi word-sense representations by jointly grounding the word and sense-form embeddings to WordNet. For a word(w) in synonym set of a sense(s), we obtain multi word-sense representations as follows:

$$v_{w,s} = \alpha_{w,s}([u_w, v_{s,form(s)}])$$

Where, u_w is the pre-trained word embedding , $v_{s,form(s)}$ is the contextualized sense-form representation of the node learned from sense-tagged corpora. For grounding, we use WordNet's synset rank information and graph structure to obtain the scaling factor, $\alpha_{w,s}$ for grounding as follows:

$$\alpha_{w,s} = 1 - clog(x), where$$
$$x = rank_{s,w} + d(w,s)$$

For word (w) in the w, s pair, WN which gives the list of senses(S_w) in decreasing order of likelihood. We use this to obtain the rank $rank_{s,w}$ of a sense$s$ with respect to w. The sense with rank 1 in S_w for a word is thus the most likely sense of the word. As outlined in our previous sections, we use an augmented synonym set by adding from a thesaurus for each synset node which means there are many word-sense pairs in our extended-WN not found in WN. For

example, the extended-WN includes "hold" as a synonym for sense "influence.n.01". This word and sense pair(hold, influence.n.01) is not found in WN. Thus "influence.n.01" is not part of S_{hold} in the original WN. If a word(w),sense(s) pair from our extended-WN is present in S_w, we use the rank directly. If not, we use the rank of the synset in S_w that is "closest" to the sense s in the word-sense pair. The WN path-similarity(p) metric is used to denote "closeness". We would also like to penalise senses s found in our extended-WN pairs more if they are farther in the WN graph structure to the original senses S_w given by WN for word w. The intuition is, the closer a sense is to a word in the WN graph, the more relevant it is to the word. The same intuition is followed in retrofitting vectors to lexicons as well(Faruqui et al. 2014). $d(S_w, s)$ is the penalizer in our equation which obtains the distance between a word and a sense as follows:

$$d(w, s) = min([1 - p(s, x) \forall x \in S_w])$$

Recall $p(s, x)$ is the path-similarity score with a higher score denoting closer pairs, implying closer pairs get assigned a lower penalizing distance. We use a monotonically decreasing distribution $1 - clog(x)$ with c as some constant in our probability distribution as found by (Arora et al. 2018). As a result, of feeding ranks and graph structure distances between w and s, to this distribution, the lower ranked(with one being the highest) and farther away synsets (or bigger d) get lower scaling scores. Senses similar in rank and distance thus get similar scaling scores.
We thus get grounded representations with the scaling factor $\alpha_{w,s}$ reflecting likelihood and ontology graph structure.

Experiments

In this section, we describe the experiments done to evaluate our multi word-sense word embeddings. We use an array of existing word similarity and relatedness datasets to conduct intrinsic evaluation and 4 datasets across 2 tasks for extrinsic evaluation.

Intrinsic Evaluation

We test our embeddings intrinsically on similarity, relatedness and contextual similarity datasets.

Word Representations To run our experiments,we pick two different embeddings of 300 dimension GLoVE(Pennington, Socher, and Manning 2014) ,and Skip-Gram(SG)(Mikolov et al. 2013a). We use these embeddings for word sense induction in our experiments because they are a popular choice for NLP systems at the time of writing the paper. The resulting CoKE embeddings after scaling and concatenation with word embeddings is 600 dimension.

Similarity Measures Given a pair of words w with M senses and $w^{'}$ with N senses, we use the following two metrics proposed by (Reisinger and Mooney 2010) for computing similarity scores without using context.

$$AvgSim(w, w^{'}) = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (cos(v_{w,i}, v_{w^{'},j}))$$

$$MaxSim(w, w^{'}) = \max_{1 \leq i \leq M, 1 \leq j \leq M} cos(v_{w,i}, v_{w^{'},j})$$

$AvgSim$ computes word similarity as the average similarity between all pairs of sense vectors. Whereas $MaxSim$ computes the maximum over all pairwise sense vector similarities.

Baselines We denote two baselines in Table 2. and Table 3., in addition to the baseline score of the single prototype word embeddings themselves. The first baseline we denote is to measure performance on concatenating sense embeddings learned from the OMSTI corpus along with word embeddings using WordNet to retrieve senses for a word. This baseline is to indicate scores on concatenating embeddings from two different sources. This is denoted as *+Synset(WN)* in the table. The second baseline, *+CoKE(Ext-WN)* is to track performance changes when splitting senses to sense-forms and grounding them to extended-WN. Finally, we show scores with *+CoKE(Thes+Ext-WN)* which reflects performance of grounded word-sense representations using sense-forms, extended-WordNet and the thesaurus.

Word Similarity We evaluate our embeddings on several standard word similarity datasets namely, SimLex (Hill, Reichart, and Korhonen 2015)(SL-999), WordSim-353(Gabrilovich and Markovitch) (WS-S), MC-30(Miller and Charles 1991) , RG-65 (Rubenstein and Goodenough 1965), YP-130 (Yang and Powers 2006),SimVerb(Gerz et al. 2016)(SV) and Rare Word(RW) similarity (Luong, Socher, and Manning 2013a).
Each dataset contains a list of word pairs with an individual score generated by humans of how similar the two words are. We calculate the Spearman correlation between the labels and the scores generated by our method. For similarity, we use $MaxSim$ as a metric to find the most similar pair among different senses of a word. The results are outlined in Table 2.

We observe that the lower performance for *Synset(WN)*, obtained by concatenating word with sense embeddings to get word-sense embeddings, is because of the limited number of synonyms for a synset recorded in WordNet along with the limited size of the dataset used to learn these embeddings.
The average improvement column in the table(Avg Improvement), shows a significant improvement in performance on splitting senses to sense-forms and grounding(*CoKE(Ext-WN)*). The benefits of this approach are reflected mainly on the SimVerb-3500 dataset. This is not a surprising result since words tend to have more syntactic forms when they occur as verbs. With distributional semantics, syntactic forms of verbs often remain close making it hard to capture differences. However drastic improvements can be seen through

Vector	WS-S	RG-65	RW	SL-999	YP	MC	SV-3500	Avg Improvement
SG	76.96	74.97	50.33	44.19	55.89	78.80	36.35	-
+Synset(WN)	-25.76	-11.85	-28.24	+0.59	+5.41	-11.44	+1.1	-10.02
+CoKE(Ext-WN)	-24.64	-7.96	-27.7	+4.04	+11.75	-9.48	+6.71	-6.75
+CoKE(Thes+Ext-WN)	**+0.21**	**+10.84**	**+1.72**	**+17.69**	**+11.69**	**+5.98**	**+13.51**	**+8.80**
Glove	79.43	76.15	45.78	40.82	57.08	78.60	28.32	-
+Synset(WN)	-23.05	-10.34	-23.03	+0.48	+0.26	-10.24	+0.47	-9.35
+CoKE(Ext-WN)	-22.11	-4.23	-25.38	+6.96	+7.02	-6.19	+8.06	-5.12
+CoKE(Thes+Ext-WN)	**+0.23**	**+11.6**	**+1.51**	**+18.29**	**+11.8**	**+7.27**	**+17.59**	**+9.75**

Table 2: Table showing performance difference using CoKE on similarity tasks. Baselines of scores of original pre-trained embeddings are included at the top. Synset(WN) indicates concatenation with synset embeddings using senses of a word from WordNet, CoKE(Ext-WN) represents CoKE obtained using extended-WordNet, and CoKE(Thes+Ext-WN) is CoKE obtained using the thesaurus augmented version of the extended-WordNet.

Vector	WS-R	MEN	MT-771	SGS	Avg Improvement
SG	61.75	73.59	67.71	56.61	-
+Synset(WN)	-12.37	-10.07	-6.15	-13.25	-10.46
+CoKE(Ext-WN)	-11.65	-8.38	-5.34	-15.72	-10.27
+CoKE(Thes+Ext-WN)	**+0.13**	**+0.71**	**+0.19**	**+8.51**	**+2.38**
Glove	66.92	79.88	71.57	58.34	-
+CoKE(WN)	-6.52	-11.31	-4.54	-14.36	-9.18
+CoKE(EXT-WN)	-6.78	-10.64	-3.8	-14.7	-8.98
+CoKE(Thes+Ext-WN)	**+0.2**	**+0.49**	**+0.47**	**+12.92**	**+3.52**

Table 3: Performance differences using CoKE on word relatedness tasks. Baselines of scores of original pre-trained embeddings are included at the top. Synset(WN) indicates concatenation with synset embeddings using senses of a word from WordNet, CoKE(Ext-WN) represents CoKE obtained using extended-WordNet, and CoKE(Thes+Ext-WN) is CoKE obtained using the thesaurus augmented version of the extended-WordNet.

Model	ρ **x 100**
(Jauhar, Dyer, and Hovy 2015)	61.3
(Iacobacci, Pilehvar, and Navigli 2015) , 2015	62.4
(Huang et al. 2012)	62.8
(Athiwaratkun and Wilson 2017)	65.5
(Chen, Liu, and Sun 2014)	66.2
CoKE + SG(Our model)	*67.3*
Rothe and Schutze (2015)	68.9

Table 4: Comparison of our multi word-sense representations with other state-of-the art representations on the Stanford Contextual Word Similarity(SCWS) dataset to evaluate polysemous word similarity.

thesaurus inclusion(*CoKE(Thes+Ext-WN)*), this is because using WordNet alone leads to limited lexemes on account of words being represented by fewer senses as opposed to a large number of senses captured for a word by word embeddings, as a result of being trained on large datasets. On including a thesaurus and augmenting the synonym set for synsets in WordNet, we see that the number of senses that represent a word drastically changes leading to more lexemes that closely reflect all possible senses of a word.

We also note that the improvements for WS-S is relatively lower; we suspect this is because the dataset is designed based on association rather than similarity alone. We also observe that as baselines of embedding spaces get higher for datasets, the performance gains reduces since most of the information is captured in the embedding spaces. The same trend is also observed in (Faruqui et al. 2014).

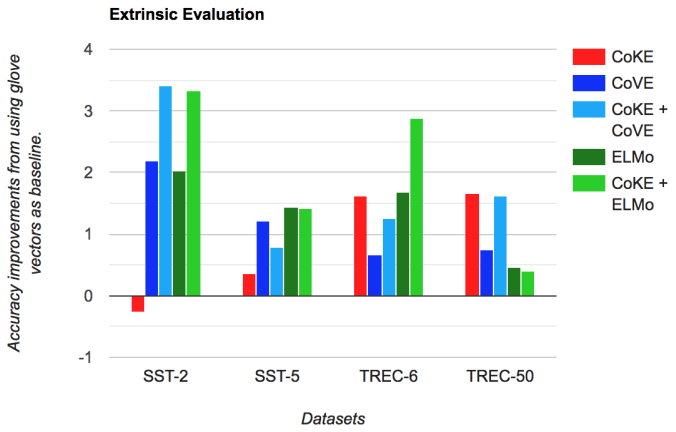

Table 5: Accuracy differences on sentiment analysis and classification tasks of CoKE, CoVE, CoVE+CoKE , ELMo, CoKE+ELMo with GLoVE as baseline.

Dataset	GloVe	CoKE	CoVE	CoKE(+CoVE)	ELMo	CoKE(+ELMo)
SST-2	85.99	85.72	88.18	**89.41**	88.02	89.32
SST-5	50.19	50.56	51.4	50.97	**51.62**	51.60
TREC-6	89.90	91.53	90.56	91.15	91.59	**92.78**
TREC-50	83.84	**85.5**	84.59	85.46	84.31	84.249

Table 6: CoKE improves performance when used alone as well as when used with a disambiguation system. Note, CoVE and ELMo are only used for disambiguation, their representations aren't included with CoKE

Word Relatedness Integration of our vectors also shows improvements in word relatedness tasks. As our benchmark, we evaluate on WS-R (relatedness) , MTurk(771) ((Halawi et al. 2012)), MEN((Bruni et al. 2012)), and on SGS130 ((Szumlanski, Gomez, and Sims 2013)) which includes phrases. We evaluate the performance of our method against standard pre-trained word embedding using Spearman correlation. We use $AvgSim$ as our metric to measure relatedness and report scores Table 3.

The baselines we use are the same as for word similarity as described above. We notice how performance improvements through sense-form splitting are not as drastic as for word similarity. This could be on account of word relatedness tasks more frequently checking for relatedness of objects rather than verbs; sense-form splitting is more beneficial to verbs than nouns on account of more varying forms of words as verbs.

We are not sure why the overall performance gains are not as high as for similarity, but the scores do reflect gains as opposed to retrofitting directly to lexicons which leads to a serious drop in relatedness. The big performance gains on SGS (Szumlanski, Gomez, and Sims 2013) is due to phrases present in the dataset. By using a thesaurus and WN, we learn multiple phrasal representations not found in the original word embedding space.

Word Similarity for Polysemous Words We use the SCWS dataset introduced by (Huang et al. 2012), where word pairs are chosen to have variations in meanings for polysemous and homonymous words. We compare our method with other state-of-the-art multi-prototype models.We find that our model performs competitively with previous models. We use the Skip-Gram(SG) word embedding with our method to allow for fair comparison, since previous work uses Skip-Gram for retrofitting to WordNet.The Spearman correlation between the labels and scores are indicated in Table 4.

Extrinsic Evaluation

A lot of the prior work on obtaining sense embeddings show performance improvements in intrinsic tasks, but leave out testing them on downstream tasks. It is thus difficult to judge the effectiveness of these representations. To bridge this gap, we run experiments on two tasks(Sentiment Analysis and Question Classification) across 4 datasets to provide some insight on the usefulness of our representations.

Datasets For sentiment analysis we use the Stanford Sentiment Treebank dataset(Socher et al. 2013). We train seperately and test on the Binary Version(SST-2) as well as the five class version(SST-5). For question classification, we evaluate performance on the TREC(Voorhees 2001) ques-

tion classification dataset which consists of open domain questions and semantic categories.

Performance Comparisons We first run experiments on CoKE by representing words as an average of their respective sense embeddings. It is a known fact that words are a weighted sum of their senses. Thus the intuition of using averaged embeddings is that having grounded word-sense representations should lead to better word representations through averaging.

Recent trends have also lead to an increasing interest in transfer learning for obtaining superior word representations. CoVE (McCann et al. 2017) and ELMo (Peters et al. 2018) show significant improvements in extrinsic tasks. CoVE uses word representations learned from a machine translation system in combination with GloVE embeddings. ELMo, on the other hand, uses a language model to obtain contextualised word representations. As shown by (Peters et al. 2018), these systems inherently act as word sense disambiguation and representation systems. They give word representations conditioned on the context it occurs in and perform on par with state-of-the-art word sense disambiguation systems, but it is unclear how informative the sense representations are. We thus hypothesise that the systems can benefit by using better sense representations.
Due to the promising performance of CoVE and ELMo as word sense disambiguation systems and increasing interest in using them in NLP tasks, we use them as disambiguation systems in our experiments to sense tag the four benchmark datasets. To get the disambiguated sense tags using CoVE or ELMo, we use the same approach as outlined in (Peters et al. 2018). We compute each word's representation in OMSTI using CoVE or ELMo and then use the average of all the representations obtained for a sense to get its respective sense representations. To disambiguate a sentence, we then run the sentence through the CoVE or ELMo architecture to get word representations and then tag the word by taking the nearest neighbour sense from the corresponding CoVE or ELMo computed sense representations. For ELMo, we use the last layer and the pre-trained version made available publicly.
In our experiments, we use the CoKE word-sense embeddings obtained by using GLoVE with the thesaurus and extended-WordNet for grounding.We pick CoKE with GLoVE embeddings to be fair in comparison with CoVE which is obtained by concatenation with GLoVE embeddings.
We thus compare performance using GLoVE, CoVE and ELMo independently, using an average of CoKE representations to get word representations, and also using ELMo/CoVE as disambiguation systems with sense-tagged words represented with CoKE embeddings(CoKE+(CoVE), CoKE(+ELMo)). Note if a word is not sense-tagged we use vanilla GLoVE vectors concatenated with an unknown vector.

Training Details To test for performance of different embeddings on datasets, we implement a single-layer LSTM(Hochreiter and Schmidhuber 1997) with a hidden size of 300 and run our experiments. Parameters were fine-tuned specifically for each task and embedding type.

Results As shown in Table 6. using CoKE shows more significant improvements with Classification as opposed to Sentiment Analysis. This is an expected outcome since our approach focuses on ontology grounding without considering polarity of words which is the primary goal of Sentiment Analysis. On the other hand, Classification as a task is more sensitive to representations that cater to similarity and relatedness between sentences. Significant improvements can be seen on classification tasks even by using averaged CoKE embeddings without disambiguation.

Qualitative Analysis

In this section, we look at some visualisations of senses induced and show how they are easily interpretable. Since sense tags have lexical mappings to an ontology, they can be looked up to find meanings. Moreover, the semantic distribution of the word-senses also plays a role in obtaining meaningful sense clusters. We analyse two things 1) Sense clusters induced 2) How using different sense forms affect representations and sense interactions in their respective word forms. For all our analysis, we use the concatenated version of CoKE + GLoVE embeddings and use Principle Component Analysis to perform dimensionality reduction.

Sense Clusters

We look at the sense clusters formed by our word specific senses embeddings for the word "rock".

The clusters for the word "rock" is depicted in Figure 2. The multiple fine-grained word-sense embeddings for the word "rock" cluster to form 5 basic senses. We see three distinct clusters that dominate. "Cluster#2" can be interpreted as all synsets that speak of rock as a "substance". In, "Cluster#3", the synsets cluster together to speak of rock as "music". An interesting property can be observed comparing "Cluster#1" and "Cluster#5". The senses found in both of these clusters interpret "rock" as "movement/motion". However, the two distinct clusters also capture the kind of motion. For instance ,the senses $roll.v.13$ and $rock.v.01$ in "Cluster#5" map specifically "sideways movement". While the senses in "Cluster#1" map to glosses "sudden movements"(convulse , lurch,move , tremble) and "back and forth movements(wobble , rock)". Another interesting property is depicted by "Cluster#4", although they are more synonymous in meaning to rock as a "substance", the senses for gravel cluster very closely to senses mapping to gloss "jerking" movements capturing deeper relations between senses.

Sense Forms

In this section, we analyse how different sense-form representations interact for synonyms within a synset. We do so by considering the word-forms "plan" and "planning" both of which are synonyms of their respective sense-forms of

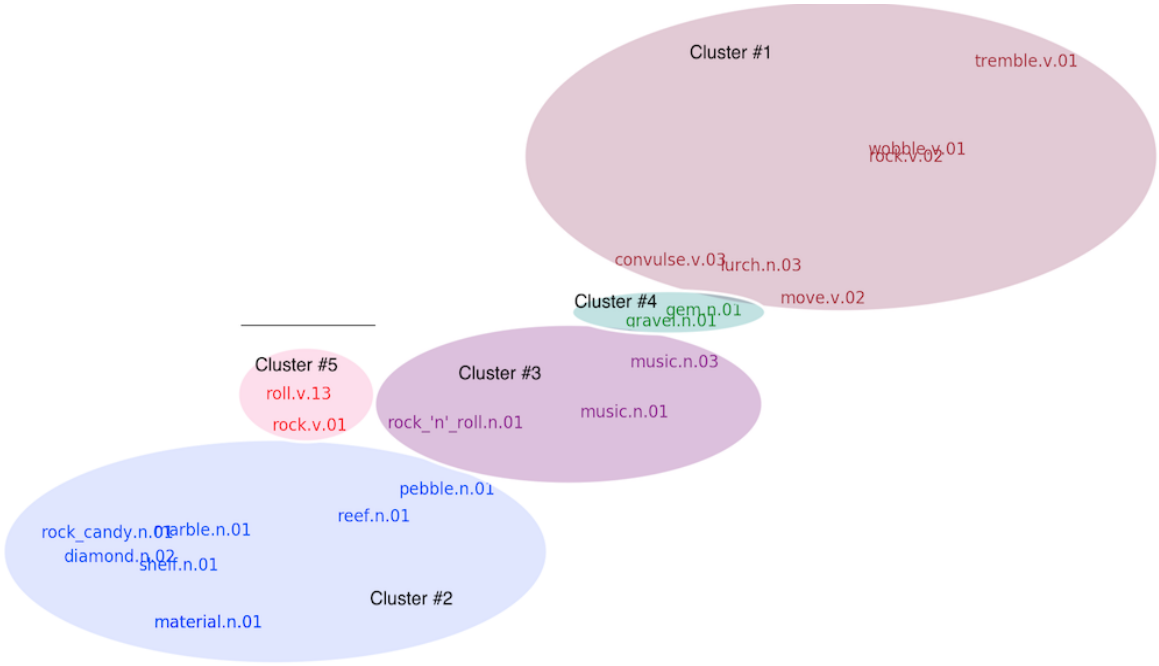

Figure 2: Sense clusters for the word ”rock”, visualized using PCA.

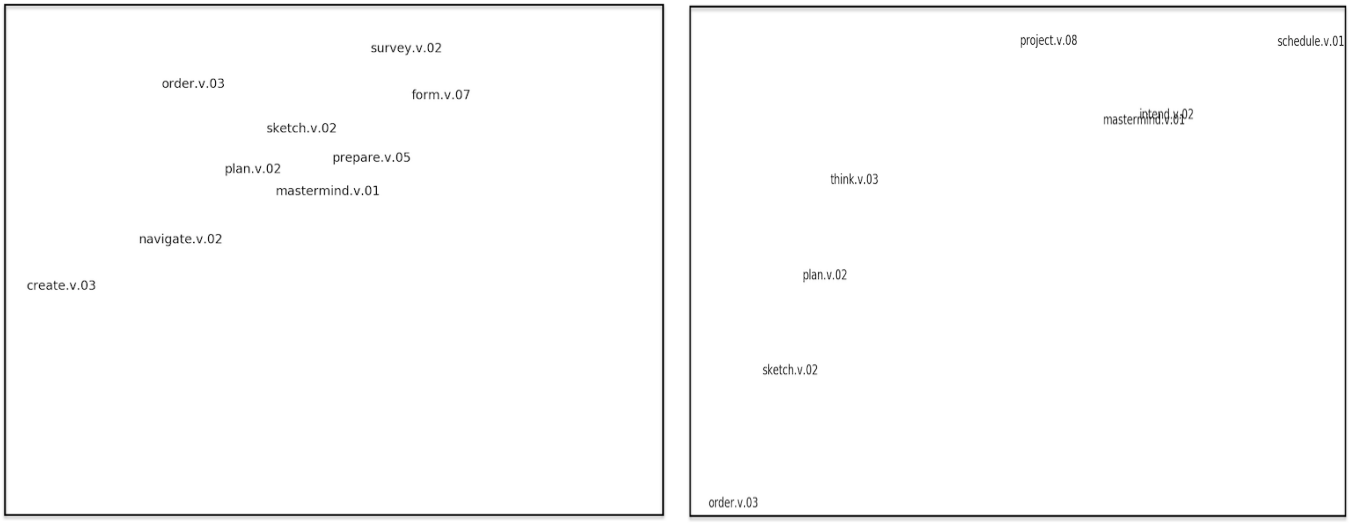

Figure 3: a) Interactions between different senses of the word ”plan” b) Interactions between different senses of the word ”planning”

“mastermind.v.01” (Gloss: plan and direct, a complex undertaking).

In order to observe the difference in sense-form relationships of word-forms, we consider only common synsets in “plan” and “planning” for visualisation and observe the interactions with each other. For the word “plan” as shown in Figure 3.a), we observe that the synset “mastermind” is closer in proximity to synsets that map to words like “plan”, “sketch”, “prepare”. In contrast, the same synset in the embedding space for ”planning” as shown in Figure 3.b) interacts closely with synsets that are analogous to “project planning”, “scheduling”, “organising”. This shows how using different sense-form representations, leads to different and unique interactions among the same group of synsets for each word.

Conclusion

In our work, we explore the possibility of obtaining multi word-sense representations and sense induction to embedding spaces by using distributional semantics and a knowledge base. The prototypes allow ease of use with WSD systems, can easily be used in downstream applications since they are portable and are flexible to use in a wide variety of tasks. Previous work on obtaining sense representations falls under three distinct clusters - Unsupervised methods, Supervised resource-specific methods and ontology grounding. By using pre-trained unsupervised embeddings, supervised sense embeddings and jointly grounding them in an ontology, ours is the first approach that lies in the intersection of all three approaches. The code and vectors will be made available publicly as well.

References

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association of Computational Linguistics* 6:483–495.

Athiwaratkun, B., and Wilson, A. G. 2017. Multimodal word distributions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1645–1656.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2016a. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2016b. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Bruni, E.; Boleda, G.; Baroni, M.; and Tran, N.-K. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, 136–145. Association for Computational Linguistics.

Chen, X.; Liu, Z.; and Sun, M. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1025–1035.

Cheng, J., and Kartsaklis, D. 2015. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. *arXiv preprint arXiv:1508.02354*.

Faruqui, M.; Dodge, J.; Jauhar, S. K.; Dyer, C.; Hovy, E.; and Smith, N. A. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Gabrilovich, E., and Markovitch, S. Computing semantic relatedness using wikipedia-based explicit semantic analysis.

Gerz, D.; Vulić, I.; Hill, F.; Reichart, R.; and Korhonen, A. 2016. Simverb-3500: A large-scale evaluation set of verb similarity. *arXiv preprint arXiv:1608.00869*.

Halawi, G.; Dror, G.; Gabrilovich, E.; and Koren, Y. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 1406–1414. ACM.

Hill, F.; Reichart, R.; and Korhonen, A. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics* 41(4):665–695.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Huang, E. H.; Socher, R.; Manning, C. D.; and Ng, A. Y. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, 873–882. Association for Computational Linguistics.

Iacobacci, I.; Pilehvar, M. T.; and Navigli, R. 2015. Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, 95–105.

Jauhar, S. K.; Dyer, C.; and Hovy, E. 2015. Ontologically grounded multi-sense representation learning for semantic vector space models. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 683–693.

Luong, M.-T.; Socher, R.; and Manning, C. D. 2013a. Better word representations with recursive neural networks for morphology. In *CoNLL*.

Luong, T.; Socher, R.; and Manning, C. 2013b. Better word representations with recursive neural networks for morphology. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*.

McCann, B.; Bradbury, J.; Xiong, C.; and Socher, R. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, 6297–6308.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Miller, G. A., and Charles, W. G. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes* 6(1):1–28.

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

Neelakantan, A.; Shankar, J.; Passos, A.; and McCallum, A. 2015. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark,

C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Reisinger, J., and Mooney, R. J. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 109–117. Association for Computational Linguistics.

Rothe, S., and Schütze, H. 2015. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. *arXiv preprint arXiv:1507.01127*.

Rubenstein, H., and Goodenough, J. B. 1965. Contextual correlates of synonymy. *Communications of the ACM* 8(10):627–633.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1631–1642.

Szumlanski, S.; Gomez, F.; and Sims, V. K. 2013. A new set of norms for semantic relatedness measures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, 890–895.

Taghipour, K., and Ng, H. T. 2015. One million sense-tagged instances for word sense disambiguation and induction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, 338–344.

Tian, F.; Dai, H.; Bian, J.; Gao, B.; Zhang, R.; Chen, E.; and Liu, T.-Y. 2014. A probabilistic model for learning multi-prototype word embeddings. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 151–160.

Voorhees, E. M. 2001. The trec question answering track. *Natural Language Engineering* 7(4):361–378.

Wu, Z., and Giles, C. L. 2015. Sense-aaware semantic analysis: A multi-prototype word representation model using wikipedia. In *AAAI*, 2188–2194.

Yang, D., and Powers, D. M. 2006. *Verb similarity on the taxonomy of WordNet*. Masaryk University.