Modeling Word Meaning in Context with Substitute Vectors

Oren Melamud
Computer Science Dept.
Bar-Ilan University
melamuo@cs.biu.ac.il

Ido Dagan
Computer Science Dept.
Bar-Ilan University
dagan@cs.biu.ac.il

Jacob Goldberger
Faculty of Engineering
Bar-Ilan University
goldbej@eng.biu.ac.il

Abstract

Context representations are a key element in distributional models of word meaning. In contrast to typical representations based on neighboring words, a recently proposed approach suggests to represent a context of a target word by a *substitute vector*, comprising the potential fillers for the target word slot in that context. In this work we first propose a variant of substitute vectors, which we find particularly suitable for measuring context similarity. Then, we propose a novel model for representing word meaning in context based on this context representation. Our model outperforms state-of-the-art results on lexical substitution tasks in an unsupervised setting.

1 Introduction

Following the distributional hypothesis (Firth, 1957), distributional models represent the meaning of a word type as an aggregation of its contexts. A recent line of work addresses polysemy of word types by representing the meaning (or sense) of each word instance individually as induced by its particular context. The context-sensitive meaning of a word instance is commonly called the word meaning *in context*, as opposed to the word meaning *out-of-context* of a word type.

A key element of distributional models is the choice of context representation. A context of a word instance is typically represented by an unordered collection of its first-order neighboring words, called *bag-of-words* (BOW). In contrast, Yatbaz et al. (2012) proposed to represent this context as a second-order *substitute vector*. Instead of the neighboring words themselves, a substitute vector

our proposals will unlock __new__ ways of raising the finance needed to develop businesses.	
representations of context	
BOW	proposals, raising, unlock, ways
substitutes	alternate, proposing, infinite, various
representation of word meaning in context	
paraphrases	new, innovative, different, alternative, novel

Table 1: Example for BOW and substitute vector representations for a context of the target word *new*. The paraphrase vector is the representation learned by our model for the meaning of *new* in this context. Only the first few entries for each vector are shown.

includes the potential filler words for the target word slot, weighted according to how 'fit' they are to fill the target slot given the neighboring words. For example, the substitute vector representing the context "I __ my smartphone." (target slot underlined), would typically include potential slot fillers such as *love*, *lost*, *upgraded*, etc.

Melamud et al. (2014) argued that substitute vectors are potentially more informative than traditional context representations since the fitness of the fillers is estimated using an n-gram language model, thereby capturing information embedded in the neighboring word order. They showed promising results on measuring word similarity out-of-context with a distributional model based on this approach.

In this paper we first propose a variant of substitute vectors as a context representation, which we find particularly suitable for measuring context similarity. Then, we extend the work in Melamud et al. (2014) by proposing a novel distributional model for representing word meaning in context, based on this context representation. Like substitute vectors, the word representations learned by our model are second-order vectors, which we call *paraphrase vectors*. Table 1 illustrates the difference between BOW and substitute vector context representations, as well as our representation for a word in context. Our model outperforms state-of-the-art results on lexical substitution tasks while learning from plain text in an unsupervised setting. [1]

2 Background

A key element in distributional models of word meaning is the choice of context representation. Traditional out-of-context distributional models aggregate the observed contexts of a target word type to derive its representation. More recent models of word meaning in context typically bias such a word type representation towards the context of each particular word instance using context similarity measures. The typical choice for context representation is a *bag-of-words* (BOW) context vector. In this vector each neighboring word type is assigned with a weight, such as its count (Erk and Padó, 2010) or *tf-idf* (Reisinger and Mooney, 2010). A more recent variant of this approach is the *continuous bag-of-words* (CBOW) vector, where context is represented as an average, or *tf-idf* weighted average, of dense low dimensional vector representations of the neighboring words (Huang et al., 2012). Context similarity is typically computed using vector Cosine.

Several types of models of word meaning in context were recently proposed in the literature, mostly based on variants of BOW context representations. Thater et al. (2011) aggregate the contexts of a target word type into a sparse syntax-based context feature vector. Then, they generate a biased vector representation by reducing the weight of each context feature the less similar it is to the context of the given word instance. Reisinger and Mooney (2010) and Huang et al. (2012) use context clustering to induce multiple word senses for a target word type, where each sense is represented by a different context feature vector. Then, they choose for each word instance the sense vector most similar to its given context. Ó Séaghdha and Korhonen (2014) use LDA (Blei et al., 2003) to aggregate word collocations to distributions over topics. Then, word meaning is represented by distributions that can be conditioned on (and hence biased towards) the given context.

3 Substitute Vectors

In this work we propose to use a little-known context modeling paradigm, representing a context word window as a *substitute vector* (Yatbaz et al., 2012). Unlike traditional context representations, a substitute vector does not comprise the first-order neighboring words of the target word. Instead it includes the second-order potential fillers for the target word slot, weighted according to how 'fit' they are to fill the target slot given the neighboring words. More formally, we denote a word window context around a target word slot as c, and the substitute vector representing c as $\vec{s}_c$. $\vec{s}_c[v]$ is the fitness weight of word v to fill the target slot in context c, for every word v in the target word vocabulary. For example, the substitute vector $\vec{s}_c$ = [big 0.35, good 0.28, bold 0.05, ...] may represent the context c = "It's a __ move.".

Yatbaz et al. (2012) used a Kneser-Ney smoothed n-gram language model (Kneser and Ney, 1995) to estimate conditional probability as the fitness weight $\vec{s}_c[v] = p(v|c)$. Using a smoothed language model is essential since context-word collocation counts are too sparse when the context considered is an entire word window. We note that given the nature of n-gram language models this representation is sensitive to word order. This property makes it appealing as it is potentially more informative than the traditional unordered BOW context representations.

4 A Model for Word Meaning in Context

The main contribution of this paper is in proposing a model for word meaning in context, which is based on substitute vector context representations instead of the traditional bag-of-words representations. To achieve this we first propose a variant of substitute vectors as our context representation. Then, we consider an out-of-context representation for a word type as the average of the substitute vector representations of its observed contexts. Finally, the in-context representation for a given word instance is a weighted average of its observed contexts. In this weighted average, contexts are weighted higher the more similar they are to the given context, using

[1]Our source code is publicly available at: `www.cs.biu.ac.il/nlp/resources/downloads/word2parvec/`

... the very essence or heart of being a __*coach*__.	
c	... the very essence or heart of being a ____.
$\vec{s}_c$	christian, non-smoker, traitor, grandparent
$\vec{p}_u$	coach, bus, train, boat
$\vec{p}_{u,c}$	coach, teacher, writer, manager

Table 2: The substitute vector $\vec{s}_c$ for context c, the out-of-context paraphrase vector $\vec{p}_u$ for the word type *u* = *coach*, and the in-context paraphrase vector $\vec{p}_{u,c}$ for *u* in context c, as learned by our model. Only the first few vector entries (weights omitted) are shown. $\vec{s}_c$, $\vec{p}_u$ and $\vec{p}_{u,c}$ are defined formally in sections 4.1 and 4.2.

a substitute vector similarity measure. Intuitively, with this weighting scheme, we wish to consider mostly the word type contexts that induce a sense similar to that of the given word instance, under the premise that similar contexts induce similar word senses. The resulting word representation is biased towards the given context on one hand due to the context weighting scheme, and is bounded to the target word type spectrum of meanings on the other hand as only contexts of that word type are taken into consideration.

Table 2 exemplifies a context substitute vector and both out-of-context and in-context word representations learned by our model for a word instance. It is evident in this case that our in-context representation comprises suitable paraphrases in contrast to the out-of-context representation. We evaluate these word representations quantitatively in Section 6. We next describe our model in more detail.

4.1 Context representation

We wish the context representation in our model to be optimized for measuring context similarity, which is typically used to bias in-context word representations towards a given context.

For the purpose of measuring similarity between contexts, we consider in this section the contexts as 'targets'. Accordingly, we observe that the substitute vector of a word window context c can be considered as a vector of first-order co-occurrence features of c, as it consists of slot filler words that are likely to co-occur with this context. Hence, we follow prior work and propose to use *Positive PMI* (PPMI) as our substitute vector feature weights, instead of the conditional probabilities used by Yatbaz et al. (2012), and vector Cosine as our context

Q: the transcendental meditation people advertised this: meditation can __fix__ many sicknesses.	
substitutes	relieve, circumvent, alleviate
R$_{sub}$: use the results of your analysis to suggest design changes that would __fix__ these problems.	
substitutes	overcome, solve, alleviate
R$_{cbow}$: __fix__ in your mind a picture of heavenly worship that is real and eternal.	
substitutes	echoing, send, stick

Table 3: Example for a context of the word *fix*, Q, and the two contexts of *fix*, R$_{sub}$ and R$_{cbow}$, most similar to it, based on substitute vector and CBOW similarity, respectively. The substitute vectors are illustrated below each context (selected substitutes in the top-10 entries shown).

similarity function (Bullinaria and Levy, 2007):

$$\vec{s}_c[v] = PPMI(c, v) = \max(0, PMI(c, v)) \quad (1)$$

$$sim(c, c') = cos(\vec{s}_c, \vec{s}_{c'}) \quad (2)$$

where $\vec{s}_c[v]$ is the fitness weight for word v in the substitute vector of context c, PMI is point-wise mutual information (Church and Hanks, 1990), and $sim(c, c')$ is our context similarity measure. We note that a context c in our setting stands for an entire word window rather than a single context word. We therefore follow Yatbaz et al. (2012) using an n-gram language model to estimate $PMI(c, v)$, as detailed in Section 5.

Table 3 illustrates an example of a given context and the contexts most similar to it, as retrieved by our substitute vector and continuous bag-of-words context similarity measures. It is evident in this case that our measure correlates with the induced senses better than the bag-of-words measure. We suggest this context similarity measure as a standalone contribution, which may be useful in other settings as well. We evaluate it quantitatively in Section 5.

4.2 Modeling word meaning

Word meaning out-of-context We first define our *out-of-context* representation for target word type u, as an average of the substitute vectors of its contexts:

$$\vec{p}_u = \frac{1}{|C_u|} \sum_{i \in C_u} \vec{s}_i \quad (3)$$

where C_u is a collection of the contexts observed for target word type u in a learning corpus, and $\vec{s}_i$ are their substitute vectors.

Word meaning in context Next, following Erk and Padó (2010), to represent the meaning of word u in context c, we would like to alter the out-of-context representation by theoretically averaging only over contexts that induce a word sense similar to that of the given context. To approximate this objective we use a weighted average of all contexts of u, where contexts are weighted according to their similarity to the given context:

$$\vec{p}_{u,c} = \frac{1}{Z} \sum_{i \in C_{cu}} sim(c,i) \cdot \vec{s}_i \tag{4}$$

where $C_{cu} = C_u \cup c$ (u's corpus contexts plus the given context) and Z is a normalization factor.

$\vec{p}_{u,c}[v] = \frac{1}{Z} \sum_{i \in C_{cu}} sim(c,i) \cdot \vec{s}_i[v]$ is the average fitness of v within the contexts of u, biased to those similar to c. We consider this a context-sensitive similarity score, indicative of the likelihood of v to be a paraphrase of u in context c.[2] Thus, we name our in-context representation for a word instance as its *paraphrase vector*.

Finally, $\vec{p}^{\,m}_{u,c}$ denotes a word representation as defined in Equation (4), where only the top-m percent of the contexts in C_{cu} most similar to c are averaged. Using low values for m means injecting a stronger bias in our model towards the given context.

5 Evaluating Context Representations

As described in Section 4, our model for word meaning in context utilizes a context similarity measure under the premise that similar contexts induce similar target word senses. In this section we describe a focused evaluation of our proposed similarity measure and prior methods with respect to this objective. This evaluation suggests that our measures may be useful as a component in other models as well.

5.1 Task description

Given a word window context c of a target word u, we wish to evaluate context similarity measures on their ability to retrieve other contexts of u from C_u that induce a similar sense. To perform such an evaluation we want a dataset of target words with thousands of sense tagged contexts in C_u for each target word u. Since available manually sense-tagged datasets, such as SemCor (Mihalcea, 1998), are not large enough for this purpose, we adopted a pseudo-word approach, with which we can automatically generate as many tagged contexts as we wish.

Pseudo-word methods consider a set of real words as pseudo-senses of an artificial pseudo-word (Pilehvar and Navigli, 2014). Specifically, we adopted a simple approach following Otrusina and Smrz (2010) to generate our pseduo-words. First, we sampled 100 words randomly from our learning corpus, ukWaC (Ferraresi et al., 2008). Then we constructed a pseudo-word based on each of these words as follows. We used WordNet (Fellbaum, 2010) to identify all of the word's synsets. Next, for each synset we chose the surface word which is the least polysemous yet occurs in our learning corpus at least 1,000 times, as a *representative* for this synset. Then, we created a pseudo-word whose pseudo-senses are the set of the representative words. For example, the pseudo-word that was generated based on the word *promote* is *elevate_encourage_advertise*. Finally, we sampled from our learning corpus 1,000 contexts for each pseudo-sense word, and for each pseudo-word we mixed together all contexts of its pseudo-sense words. The original pseudo-sense word for each context was recorded as its sense tag.

Next, for each pseudo-word, we sampled a single *query context* from all of its mixed contexts and then ranked the remaining contexts according to each of the compared context similarity measures. We computed precision at top-1, top-1% and average precision for the ranked lists, where a true-positive is a context with an identical sense tag as the query context. We repeated this procedure, sampling 100 different query contexts and computed the mean precision values. Finally, we report for each compared method, the average of the mean precision values for all 100 pseudo-words. Our pseudo-word dataset consists on average around 4.5 senses and 4,500 tagged contexts per pseudo-word. [3]

5.2 Compared methods

All compared methods are unsupervised and use the plain text of ukWaC (Ferraresi et al., 2008), a two

[2]This can be considered an in-context extension of the out-of-context similarity score proposed by Melamud et al. (2014).

[3]Our pseudo-word dataset is available at: `www.cs.biu.ac.il/nlp/resources/downloads/word2parvec/`

billion word web corpus, as their learning corpus. We converted every word that occurs less than 100 times in the corpus to a special rare-word token, and all numbers to a special number token, obtaining a vocabulary of a little under 200K word types.

5.2.1 Substitute vector similarity

We learned a 5-gram Kneser-Ney language model from our learning corpus using KenLM (Heafield et al., 2013). Following Yatbaz et al. (2012), we used FASTSUBS (Yuret, 2012) with this language model to efficiently generate substitute vectors pruned to their top-n substitutes, $v_1..v_n$, and normalized such that $\sum_{i=1..n} p(v_i|c) = 1$. In order to make our substitute vectors compatible with the pseudo-word setting, for each substitute vector we replaced the entries of all of the pseudo-sense words with a single pseudo-word entry, and assigned it with the sum of the conditional probabilities of the pseudo-sense words. Next, we computed the Positive PMI weights for the substitutes, $\vec{s}_c[v_i] = PPMI(v_i, c) = \max(0, \log(\frac{p(v_i|c)}{p(v_i)}))$, where $p(v_i)$ is the unigram probability of the word v_i in our learning corpus. The unigram probability of a pseudo-word is the sum of the probabilities of its pseudo-sense words. Finally, we computed context similarity as substitute vector Cosine.

$SUB_{weight,n}$ denotes our similarity measure, where n is the pruning factor and $weight \in \{cond, ppmi\}$ denotes conditional probabilities and PPMI fitness weights, respectively. We note that by using a 5-gram language model we consider a context word window of 4 words on each side of the target word.

5.2.2 Bag-of-words similarity

Bag-of-words similarity between two context word windows is computed as vector Cosine between their bag-of-words vector representations. We use $BOW_{weight,l}$ to denote these context similarity measures, where l is the size of the context word window on each side of the target word (we use *sent* to denote the entire sentence), and *weight* $\in$ {*tf*, *tfidf*} stands for term frequency and *tf-idf* weights, respectively. $CBOW_{weight,l}$ is used to denote the same for the continuous bag-of-words method, which is based on averaging the dense vector word representations. We used *word2vec* (Mikolov et al., 2013) to learn these dense vectors.[4]

Method	P@1	P@1%	AvgPrec
$SUB \cdot CBOW$	**80.6**	**67.1**	**44.5**
$SUB_{ppmi,1000}$	73.1	60.0	**44.0**
$SUB_{ppmi,100}$	**74.5**	**61.0**	42.8
$CBOW_{tfidf,8w}$	68.4	58.0	43.5
$SUB_{cond,1000}$	63.6	53.0	38.6
$SUB_{cond,100}$	63.1	52.7	38.5
$BOW_{tfidf,sent}$	62.8	51.3	34.6
Random	30.4	30.4	30.4

Table 4: Precision values for compared context similarity measures. Only the best performing configurations for BOW and CBOW are shown.

5.3 Results

The results presented in Table 4 support our hypothesis that our proposed substitute vector similarity measure is particularly suitable for measuring context similarity, at least in our setting. Our similarity measures outperform both CBOW and BOW baselines on P@1 and P@1%, with statistical significance at $p < 0.01$ for $SUB_{ppmi,100}$, and $p < 0.05$ for $SUB_{ppmi,1000}$, on a paired t-test. On average precision they perform similarly to CBOW. Within the substitute vector measures, our proposed PPMI weights significantly outperform the previously used conditional probabilities, and the choice of pruning factor has a small impact. Within the bag-of-words measures, the CBOW measure significantly outperforms the BOW measure, with an optimal window size of 8 (on each side). This suggests that CBOW's ability to capture context word similarities via its dense word representations is beneficial.

Finally, *SUB·CBOW* denotes a combined similarity measure, which is the geometrical mean between the scores of the best configurations of the respective methods. We see that this combination yields substantial improvement, outperforming all other baselines across all precision categories, with $p < 0.0001$ for P@1 and P@1%. We hypothesize that this is due to the synergy between the word order sensitivity of *SUB* and the word similarities and larger window size captured by *CBOW*.

[4]We experimented with various parameters of word2vec, observing small differences in performance. We report here the results with the best configuration (cbow 1, negative sampling 15, window size 8).

6 Evaluating Word Representations

Models of word meaning in context are commonly evaluated in lexical substitution tasks on predicting paraphrases of a target word that preserve its meaning in a given context. Conveniently, our paraphrase vector representations for words include exactly these predictions. We evaluated our model on two lexical substitution datasets under two types of tasks and compared it to the state-of-the-art as described next.

6.1 Lexical substitution datasets

The dataset introduced in the lexical substitution task of SemEval 2007 (McCarthy and Navigli, 2007), denoted here LS07, is the most widely used for the evaluation of lexical substitution. It consists of 10 sentences extracted from a web corpus for each of 201 target words (nouns, verbs, adjectives and adverbs), or altogether 2,010 word instances in sentential context, split into 300 trial sentences and 1,710 test sentences. The gold standard provided with this dataset is a weighted lemmatized paraphrase list for each word instance, based on manual annotations.

A more recent dataset (Kremer et al., 2014), denoted LS14, provides the same kind of data as LS07, but instead of target words that were specifically selected to be ambiguous as in LS07, the target words here are simply all the content words in text documents extracted from news and fiction corpora. LS14 is also much larger than LS07 with over 15K target word instances.

6.2 Predicting lexical substitutions

6.2.1 Task description

In SemEval 2007 the lexical substitution task organizers evaluated participant systems on their ability to predict the paraphrases in the gold standard of the LS07 test-set in a few subtasks (1) *best* and *best-mode* - evaluate the quality of the best predictions (2) *oot* and *oot-mode* (out of ten) - evaluate the coverage of the gold paraphrase list by the top ten best predictions.[5] We performed this evaluation on both the LS07 and LS14 datasets.

[5]For brevity we do not describe the details of these subtasks. We report only *recall* scores as in this task *recall=precision* for all methods that predict paraphrases to all of the instances in the dataset, as we did.

6.2.2 Compared methods

We used the same learning corpus and substitute vector generation procedure as described in Section 5. For every target word type u (not lemmatized) in the LS07 and LS14 datasets, we sampled a collection of 20K sentence contexts from our learning corpus (or less for word types with lower frequency), denoted C_u.[6] Next, we generated substitute vectors, pruned to top-100 entries, for these contexts. For every target word type u, we discarded the contexts in C_u where u itself is not in the top-100 predicted substitutes, assuming that either these contexts are not typical to u, or that the quality of the predicted substitutes is low. This omits approximately 25% of all contexts. Finally, we generated top-100 and top-1000 substitute vectors for all of the instances in the LS07 and LS14 datasets.

We used a generalization of PPMI, called *Shifted PPMI* (Levy and Goldberg, 2014), as our substitute vector fitness weights: $SPPMI(v, c; s) = \max(0, PPMI(v, c) - s)$, where s is a global shift constant. Levy and Goldberg (2014) showed that SPPMI outperformed PPMI on various semantic tasks. We tuned the value of $s \in \{0.0, 1.0, 2.0, 3.0\}$ on the trial portion of the LS07 dataset and used the best value, 2.0, on the LS07 test set and on LS14. We omit results based on conditional probability weights for brevity as they were substantially worse. Next, for every LS07/LS14 instance of word u in context c we generated a paraphrase vector according to Equation (4). We used the paraphrase vectors sorted by entry scores as our paraphrase predictions. P_n^{in} (in-context) denotes this method, where n is the pruning factor used for generating the substitute vectors of the lexical substitution datasets.[7]

As baseline, we generated our meaning out-of-context paraphrase vectors according to Equation (3), denoted P^{out}. We also used *word2vec* (Mikolov et al., 2013) to generate dense word vectors for all word types in our learning corpus.[8] Para-

[6]In general, we observed that the results improve the more contexts are sampled up to $\sim 10K$ contexts per word type.

[7]For the learning corpus contexts we always used top-100 substitute vectors to reduce computational complexity.

[8]We experimented with 600-dimension vectors, negative sampling value 15, both skip and cbow options, and various window sizes, and tuned these parameters on the trial portion of the LS07 dataset.

phrase predictions for this baseline, denoted $w2v^{out}$, were computed as the words most similar to the target word based on dense vector Cosine similarities, ignoring the context (out-of-context).

In the *best* subtasks we used only the top ranked lemmatized paraphrase (best prediction) suggested by each of the compared methods. In the *oot* subtasks we used the top-10 lemmatized paraphrases.

To the best of our knowledge, Biemann and Riedl (2013) is the only prior work that attempted to perform the original SemEval 2007 task on the LS07 dataset, learning only from corpus data like we do. They merged Gigaword (Parker et al., 2011) and LLC (Richter et al., 2006) as their learning corpus, which is similar in size to ours. We denote by *Biemann*^{in} and *Biemann*^{out} the reported results for their in-context and out-of-context methods, respectively. There is no previously reported result for this task on LS14.

6.2.3 Results

The results are shown in Table 5. First, we note that our out-of-context method significantly outperforms the out-of-context word2vec baseline on all subtasks in both LS07 and LS14, showing that our model performs well on predicting paraphrases even out-of-context. Furthermore, our meaning in-context methods show significant additional gains in performance on LS07, with top-1000 pruning performing a little better than top-100. On LS14 we see smaller gains, which may be due to the fact that its target words are less ambiguous by construction. This behavior is consistent with similar findings in Kremer et al. (2014). Finally, both *Biemann*^{out} and *Biemann*^{in} exhibit substantially lower performance on LS07 than our methods, achieving scores that are close to the word2vec baseline.

We note that all ten systems that participated in the original SemEval 2007 task on the LS07 dataset followed a two-step scheme (1) generating paraphrase candidates using a manually constructed thesaurus, such as WordNet; (2) ranking the candidates according to the given context based on data from various learning corpora. We stress that in contrast to all these systems, our model does not utilize manually constructed thesauri, and therefore addresses a much harder problem of predicting paraphrase substitutes out of the entire vocabulary, rather than merely ranking a small given set of candidates. Even so, in the *best* subtasks we achieve top results with respect to the reported score range of these systems, 2.98-12.90 for *best*, and 4.72-20.73 for *best-mode*. In comparison to the reported *oot* score range, our results are lower than average.

Method	best	best-m	oot	oot-m
LS07 test-set				
P^{in}_{1000}	**12.72**	**21.71**	**36.37**	**52.03**
P^{in}_{100}	12.25	20.73	35.54	50.98
P^{out}	10.68	18.29	32.58	46.34
$w2v^{out}_{skip,4w}$	8.25	13.41	29.27	39.92
Biemann^{in}	n/a	n/a	27.48	37.19
Biemann^{out}	n/a	n/a	27.02	37.35
LS14 all				
P^{in}_{1000}	**8.07**	**17.37**	**26.67**	**46.23**
P^{in}_{100}	7.93	16.97	26.24	45.58
P^{out}	7.80	16.90	25.57	44.66
$w2v^{out}_{skip,4w}$	5.99	12.21	22.66	36.98

Table 5: *best* and *oot* subtasks scores for all compared methods. best-m and oot-m stand for the *mode* scores.

6.3 Ranking lexical substitutions

6.3.1 Task description

Most works that used the LS07 dataset after SemEval 2007, as well as the results reported for LS14, focused only on candidate ranking. Instead of using a thesaurus, they obtained the set of paraphrase candidates for each target type by pooling the annotated gold-standard paraphrases from all of its instances.[9] The quality of the rankings with respect to the gold standard was measured using Generalized Average Precision (GAP) (Kishida, 2005). Furthermore, all of the works compared in this section discarded multi-word expression substitutes from the original gold standard, and omitted instances who thus remained with no gold paraphrases. We follow the same evaluation settings for this task.

6.3.2 Compared methods

We observe that in this task, the objective is to rank candidates that are known to be semantically similar to the target word in some context. Therefore, we hypothesize that possibly more focus should be given in this case to assessing the

[9] A target type is defined as the pair (word lemma, pos), where pos $\in$ {noun, verb, adjective, adverb}.

compatibility between the candidates and the given context versus their semantic similarity to the target word. To this end, we explore strategies with different points of balance between these two factors. On one hand we evaluate the scores assigned to the candidates by our out-of-context paraphrase vector representation, which is based only on semantic similarity. Similarly, we also evaluate rankings based on word2vec similarity scores, with a range of learning parameters (same range as used in Section 6.2) and report the best results that we were able to obtain. On the other hand, we ignore the target word identity and consider only context compatibility by ranking candidates based on their conditional probabilities to fill the target word slot, as reflected in the respective context substitute vector representation, denoted $S_{cond,1000}$.

Finally, we rank the candidates using the scores in our in-context paraphrase vectors from Section 6.2. However, this time we check the effect of injecting a stronger bias towards the given context c, by averaging only the top-m percent contexts most similar to c, for $m \in \{1\%, 5\%, 10\%, 100\%\}$, as described in Section 4.2. We denote this as $P_n^{in,m}$. We do not report results based on conditional probability weights, as they perform substantially worse than our SPPMI weights. We also report the most recent state-of-the-art results on both LS07 and LS14. On LS07, we report our results both on the test-set and on the entire dataset (trial+test).

6.3.3 Results

The results are shown in Table 6. Looking first at the results on the LS07 dataset, we see that not surprisingly both our out-of-context method, P^{out}, and the word2vec baseline, $w2v^{out}_{skip,2w}$, which ignore the given context, achieve relatively low results. $S_{cond,1000}$ that considers only the context compatibility performs a little better. Next, we see that our in-context method, $P^{in,100\%}_{1000}$ outperforms all of the above, but $P^{in,5\%}_{1000}$, which is more strongly biased to context compatibility performs even better. For brevity, we report only the results for $m = 5\%$, which performed best on the trial portion of LS07, but the results are almost as good for $m = 1\%$ and $m = 10\%$. This supports our hypothesis that in the ranking task more focus is to be given to context compatibility. As in the prediction task in Section 6.2, the pruning factor of 100 performed slightly (up to half a point) worse than 1000.

Method	Resources	LS07 test	LS07 all	LS14 all
$P^{in,5\%}_{1000}$	UW	**55.2**	**55.1**	**50.2**
$P^{in,100\%}_{1000}$		52.0	51.7	50.0
$S_{cond,1000}$	UW	48.6	48.4	46.4
P^{out}		46.6	45.9	47.9
$w2v^{out}_{skip,2w}$		45.2	45.2	46.5
Random		29.7	30.0	33.8
Kremer, 2014†	GW	n/a	52.5	47.8
Thater, 2011	GW	n/a	51.7	n/a
Séaghdha, 2014	WP,BN	n/a	49.5	n/a
Moon, 2013	UW,BN,WN	n/a	47.1	n/a
	GW,WN	n/a	46.7	n/a
Szarvas, 2013b	LLC,WN	n/a	55.0*	n/a
Szarvas, 2013a	LLC,WN	n/a	52.4*	n/a

Table 6: GAP scores for compared methods. UW = ukWaC; GW = Gigaword; WP = Wikipedia; WN = WordNet; BN = British National Corpus (Aston and Burnard, 1998).
† A re-implementation of the model in Thater, 2011.
* Obtained by a supervised method.

In comparison to previous results, our method achieves the best reported GAP score to date, on par with Szarvas et al. (2013b). However, we note that both Szarvas et al. (2013b) and Szarvas et al. (2013a) follow a supervised approach, training on the LS07 gold standard with 10-fold cross validation, as well as incorporate features from WordNet. Therefore, they cannot be directly compared with unsupervised models, such as our own. Our model and previous works used different learning corpora that are similar in size. Moon and Erk (2013) reported results for both Gigaword and ukWaC, showing minor differences in performance.

The results on LS14 exhibit a similar behavior with our method outperforming the state-of-the-art. However, as also reported in Kremer et al. (2014), the performance gain achieved by taking the given context into consideration is smaller than in LS07. Again, this seems to be due to the nature of LS14, which is not biased to ambiguous target words.

Benchmark	Orig	1000	100
best	12.7	12.1	11.2
best-m	21.7	20.9	19.7
oot	36.3	34.8	32.5
oot-m	52.0	50.4	46.4
GAP test	55.2	52.0	50.9
GAP all	55.1	51.8	50.7

Table 7: The effect of context clustering on our model's performance on LS07. *Orig* stands for best configuration of our model with no clustering, and 1000/100 stand for the same configuration with that number of clusters.

6.4 Computational efficiency and clustering

To generate our in-context paraphrase predictions for an instance of a target word u, our model performs a weighted average over all of its context substitute vectors in C_u. The run-time complexity of this procedure is reasonably efficient at $O(|C_u| \cdot n)$, where n is the vector pruning factor. This is comparable to the complexity of the state-of-the-art algorithm before this work (Thater et al., 2011) and even to word2vec's dense vector computations. As a point of reference, in our experiments it took $\sim$300 msec to generate an in-context paraphrase vector for a given word instance on a modest single core, which was only about 3 times slower than the word2vec computation.

Memory consumption of our model is not an issue when operating in an 'offline' mode. In this mode all the target word instances in a test set (such as LS07 or LS14), can first be sorted according to their word type. Then, while processing all instances of the same word type u one after the other, only the substitute vectors in C_u need to be loaded into memory. In contrast, in an 'online' mode, to be ready for any arbitrary word instance input, our model would need to keep in memory substitute vectors for all the word types in the vocabulary V. The space complexity in this case is $O(|C_u| \cdot n \cdot |V|)$, which can easily reach memory consumptions in the order of $\sim$100 GB or more, requiring a large-scale server.

To address this challenge, we present a more coarse-grained variant of our model, where for each word type u we keep only k substitute vectors instead of all individual context vectors, thereby bounding the memory consumption to $O(k \cdot n \cdot |V|)$. To this end, for each word type u we used spherical k-means to cluster the $\sim$20,000 substitute vectors in C_u into either 100 or 1000 clusters. Then, instead of C_u, we used the collection of its cluster centroids, pruned to their top-100 entries. Table 7 shows the results when applying this to our best performing configurations on the LS07 dataset. The results show relative performance degradation when fewer clusters are used, indicating that some relevant information may be lost in this process. However, absolute performance remains competitive, suggesting that this is a viable option when memory consumption is a concern. The results on the LS14 dataset show similar trends.

7 Discussion and Future Work

We proposed a model for word meaning in context whose main novelty is in representing contexts as substitute vectors. Our model outperformed state-of-the-art baselines in both predicting and ranking paraphrases of words in context in two different lexical substitution tasks. As another potential contribution, the context similarity measures used in our model performed well on a targeted evaluation, suggesting that they may be useful as a component in other applications as well.

Substitute vectors were successfully used earlier for performing part-of-speech and word sense induction tasks (Baskaya et al., 2013; Yatbaz et al., 2014), not addressed in this work. These works took a different approach, embedding words in a low dimensional space, based on target-substitute pairs sampled from substitute vectors. It would be interesting to explore how our approach applies to these tasks.

Finally, a preliminary qualitative analysis showed that low quality substitute vectors may be a factor limiting our model's performance. This suggests that generating substitute vectors with better language models, such as neural language models, is a potential path to further improvements.

Acknowledgments

We thank our reviewers for their helpful remarks. This work was partially supported by the Israel Science Foundation grant 880/12, the German Research Foundation through the German-Israeli Project Cooperation (DIP, grant DA 1600/1-1), and the European Community's Seventh Framework Programme (FP7/2007-2013) grant 287923 (EXCITEMENT).

References

Guy Aston and Lou Burnard. 1998. *The BNC handbook: exploring the British National Corpus with SARA*. Capstone.

Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. Ai-ku: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Proceedings of the SemEval*.

Chris Biemann and Martin Riedl. 2013. Text: Now in 2d! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

John A. Bullinaria and Joseph P. Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39(3):510–526.

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of ACL (short papers)*.

Christiane Fellbaum. 2010. *WordNet*. Springer.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4)*.

John R. Firth. 1957. A synopsis of linguistic theory 1930-1955. *Studies in linguistic analysis*, pages 1–32.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of ACL*.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*.

Kazuaki Kishida. 2005. *Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments*. National Institute of Informatics Tokyo, Japan.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of ICASSP*.

Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What substitutes tell us-analysis of an all-words lexical substitution corpus. In *Proceedings of EACL*.

Omer Levy and Yoav Goldberg. 2014. Neural word embeddings as implicit matrix factorization. In *Proceedings of NIPS*.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of SemEval*.

Oren Melamud, Ido Dagan, Jacob Goldberger, Idan Szpektor, and Deniz Yuret. 2014. Probabilistic modeling of joint-context in distributional similarity. In *Proceedings of CoNLL*.

Rada Mihalcea. 1998. Semcor semantically tagged corpus. *Unpublished manuscript*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Taesun Moon and Katrin Erk. 2013. An inference-based model of word meaning in context as a paraphrase distribution. *ACM Trans. Intell. Syst. Technol.*, 4(3):42:1–42:28.

Diarmuid Ó Séaghdha and Anna Korhonen. 2014. Probabilistic distributional semantics with latent variable models. *Computational Linguistics*, 40(3):587–631.

Lubomir Otrusina and Pavel Smrz. 2010. A new approach to pseudoword generation. In *Proceedings of LREC*.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition, june. *Linguistic Data Consortium, LDC2011T07*.

Mohammad Taher Pilehvar and Roberto Navigli. 2014. A large-scale pseudoword-based evaluation framework for state-of-the-art word sense disambiguation. *Computational Linguistics*, 40(4):837–881.

Joseph Reisinger and Raymond J. Mooney. 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*.

Matthias Richter, Uwe Quasthoff, Erla Hallsteinsdóttir, and Chris Biemann. 2006. Exploiting the Leipzig corpora collection. *Proceesings of the IS-LTC*.

György Szarvas, Chris Biemann, Iryna Gurevych, et al. 2013a. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of HLT-NAACL*.

György Szarvas, Róbert Busa-Fekete, and Eyke Hüllermeier. 2013b. Learning to rank lexical substitutions. In *Proceedings of EMNLP*.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of IJCNLP*.

Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of EMNLP*.

Mehmet Ali Yatbaz, Enis Rıfat Sert, and Deniz Yuret. 2014. Unsupervised instance-based part of speech induction using probable substitutes. In *Proceedings of COLING*.

Deniz Yuret. 2012. FASTSUBS: An efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. *Signal Processing Letters, IEEE*, 19(11):725–728.