Contextualized Word Representations for Multi-Sense Embedding

Kazuki Ashihara
Osaka University
ashihara.kazuki@ist.osaka-u.ac.jp

Tomoyuki Kajiwara
Osaka University
kajiwara@ids.osaka-u.ac.jp

Yuki Arase
Osaka University
arase@ist.osaka-u.ac.jp

Satoru Uchida
Kyushu University
uchida@flc.kyushu-u.ac.jp

Abstract

Distributed word representations are used in many natural language processing tasks. When dealing with ambiguous words, it is desired to generate multi-sense embeddings, *i.e.*, multiple representations per word. Therefore, several methods have been proposed to generate different word representations based on parts of speech or topic, but these methods tend to be too coarse to deal with ambiguity. In this paper, we propose methods to generate multiple word representations for each word based on dependency structure relations. In order to deal with the data sparseness problem due to the increase in the size of vocabulary, the initial value for each word representations is determined using pre-trained word representations. It is expected that the representations of low frequency words will remain in the vicinity of the initial value, which will in turn reduce the negative effects of data sparseness. Extensive evaluation results confirm the effectiveness of our methods that significantly outperformed state-of-the-art methods for multi-sense embeddings. Detailed analysis of our method shows that the data sparseness problem is resolved due to the pre-training.

1 Introduction

Distributed word representations by neural networks (Mikolov et al., 2013a; Mikolov et al., 2013b; Pennington et al., 2014; Levy and Goldberg, 2014; Bojanowski et al., 2017), which are one of the implementations of the distributional hypothesis (Harris, 1954), have shown surprising effectiveness in representing semantic or syntactic information of words by dense vectors. They have become the standard language resource for many natural language processing tasks, such as machine translation (Sutskever et al., 2014), text classification (Mikolov and Com, 2014) and lexical substitution (Melamud et al., 2015). Widely used methods for word embedding including CBOW (Continuous Bag-of-Words) (Mikolov et al., 2013b) and SGNS (Skip-gram with Negative Sampling) (Mikolov et al., 2013a) generate a single word representation per word. Hence several meanings of the word are mixed in the representation, which significantly affect the downstream usability.

To solve this problem, several methods (Neelakantan et al., 2014; Paetzold and Specia, 2016; Fadaee et al., 2017; Athiwaratkun and Wilson, 2017) to generate multi-sense embeddings, *i.e.*, representations for each word sense are proposed. Because word sense disambiguation itself is still an open problem, these studies use approximate approaches. Paetzold and Specia (2016) use parts of speech to distinguish functionally different word senses, while Fadaee et al. (2017) propose to use topics. However, either parts of speech or topics are too coarse to distinguish word senses. In the following examples, both sentences have the topic of food, and the parts of speech of the words `soft` are both adjective.[1]

ex.1) *I ate a **soft** cheese.*

ex.2) *I drank **soft** drinks.*

Although the previous studies regard the senses of

[1] We use the `typewriter` font to indicate examples.

32nd Pacific Asia Conference on Language, Information and Computation
Hong Kong, 1-3 December 2018

words `soft` as the same, they are indeed different; the word `soft` in ex.1) expresses the meaning of tender but in ex.2) it denotes the meaning of non-alcoholic. In order to capture word senses at a finer-grained level, we propose a method to consider context words as a clue to distinguish word senses and generate representations for each pair of word and its context word.

Specifically, we regard words with dependency relations in a sentence as context words. For example, the `soft` in ex.1) has a representation of `soft_cheese` while that in ex.2) has `soft_drink`. In our approach, data sparseness is a challenge because a word has as many representations as the number of its context words. To deal with the data sparseness problem, we lemmatize each word and conduct pre-training of word representations using a conventional algorithm and use them as initial weights of the representations. It is expected that the representations of low frequency word pairs will remain in the vicinity of the initial value, which will in turn reduce the negative effects of data sparseness.

Evaluation on the Context-Aware Word Similarity Task (Huang et al., 2012) and Lexical Substitution Task (McCarthy and Navigli, 2007; Kremer et al., 2014) are conducted to investigate the effects of the generated word representations in downstream tasks. The results show that our method significantly outperforms the state-of-the-art methods for multi-sense embeddings. In addition, detailed analysis confirms that the data sparseness problem has been effectively resolved by our methods due to pre-training.

2 Related Work

Li and Jurafsky (2015) show that generating multiple word representations per word contributes to improving the performance of downstream tasks such as parts of speech tagging and semantic relation identification.

There are several previous studies (Neelakantan et al., 2014; Paetzold and Specia, 2016; Fadaee et al., 2017; Athiwaratkun and Wilson, 2017) that generate multiple word representations per word. Athiwaratkun and Wilson (2017) assume that all words have a predetermined number of word senses

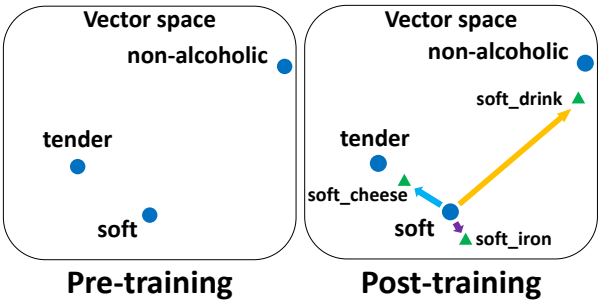

Figure 1: Outline of the proposed method.

and generate multiple word representations for each word. They assume two or three as the number of word senses, but it varies from word to word. Neelakantan et al. (2014) group word senses into clusters based on the similarity of the context and generate several word representations for each cluster. Word sense disambiguation is itself a difficult task, hence the quality of the clustering significantly affects the learning of word representations. Paetzold and Specia (2016) generate multiple word representations based on parts of speech. Because the error rate of POS tagging is low, this approach seems to be stable in quality, but there are some words with ambiguity among the same parts of speech. Fadaee et al. (2017) generate different word representations for each topic. Topics can distinguish meanings with finer granularity compared to the approach using parts of speech, but there are cases where topics are still coarse in distinguishing the meanings of polysemous words as shown in ex.1) and ex.2).

3 Proposed Method

In the present study, we aim to generate representations at a finer-grained level for each word that can capture differences in word senses. As a clue to distinguish word senses, we focus on words with dependency relations in a sentence, which are referred to as *context-word* hereafter. We assume that each word w_i has different meanings that are characterized by *context-words* C and generate $|C|$ sets of word representations for word w_i. The number of *context-words* should be larger than that of the word senses, but we assume that each representation does not interfere each other and can be independently used for downstream tasks.

Algorithm 1 Word representation generation with *context-words*

Input: Set of training sentences S, Window size W in CBOW($\cdot$)
for all $s \in S$ **do**
  $D \leftarrow$ DEPPARSE(s) ▷ dependency parsing
  for $i \leftarrow 1, n$ **do** ▷ n is length of s
    $l_c \leftarrow s[i-W:i]$ ▷ slice left W words
    $r_c \leftarrow s[i:i+W]$ ▷ slice right W words
    $C \leftarrow$ GETCONTEXTW($s[i], D$)
    for $c \in C$ **do**
      $w_c \leftarrow s[i] + "\_" + c$ ▷ concatenate with its *context-word*
      CBOW(w_c, l_c, r_c)
    end for
  end for
end for
function GETCONTEXTW(w, D)
  $C \leftarrow \emptyset$
  for all $h, d \in D$ **do** ▷ h is head, d is dependent word
    if $h = w$ **then**
      $C \leftarrow C + d$
    else if $d = w$ **then**
      $C \leftarrow C + h$
    end if
  end for
  return C
end function

Algorithm 1 shows the pesudo-code of our method. For each sentence s in the training data S, it obtains dependency relations D by dependency parsing. Then each word $s[i] \in s$ is processed to collect C in GETCONTEXTW($s[i], D$). Finally, $s[i]$ is concatenated with one of its *context-words* c as w_c. Then w_c is the input to CBOW(w_c, l_c, r_c) together with left and right W words, l_c and r_c, respectively, as the same manner with the plain training of CBOW algorithm, where W is the window size defined in CBOW.

In the example shown in Figure 1, since there are `cheese`, `drink` and `iron` as *context-words* of the target word `soft`, our approach generates word representations for each pair such as `soft_cheese`, `soft_drink`, and `soft_iron`.

In the proposed method, we need to deal with the data sparseness problem because there are a number of word combinations among target and *context-words*. Specifically, we take the following two measures to address this problem.

1. Changing each word into lemma using a lemmatizer.

2. Setting the initial value for each word representations using the pre-trained word representations.

In measure (1), we can reduce the size of vocabulary to that of lemmas. In measure (2), we first pre-train vanilla word representations (without considering the *context-word*) using CBOW and use them to initialize our representations based on *context-word* (post-training). For example, we obtain word representations of `soft` through normal CBOW algorithm. Using this as an initial value for `soft_cheese` and `soft_drink`, we train their representations in the post-training.

It is supposed that if a certain word representation is not learned well due to its low frequency in a training data, word representations in such cases are expected to remain in the vicinity of the initial value by measure (2) and it will in turn reduce the negative effect of data sparseness. Figure 1 illustrates how the word representations of `soft_cheese`, `soft_drink`, and `soft_iron` change from the initial representation of the word `soft` as a starting point. As a result of the post-training, `soft_cheese` whose meaning is close to the main meaning of the word `soft` is located near the initial value of `soft` and `soft_drink` which has different meaning is located near the position of `non-alcoholic`. Note that `soft_iron` with a low frequent *context-word* remains in the vicinity of the initial value of `soft`.

4 Evaluation Setting

In order to evaluate the effects of the proposed method in downstream tasks, we conducted experiments in Context-Aware Word Similarity Task and Lexical Substitution Task. Both tasks require considering word senses, hence it is especially important to handle word representations for sense disambiguation.

4.1 Preprocessing

We use 988M sentences extracted from main contents of English Wikipedia[2] articles as training data. We lemmatize each word using Stanford Parser (Manning et al., 2014) and replace words with frequency of 200 or less to ⟨unk⟩ tag to reduce the size of the vocabulary, which results in 112,087 words. Dependency relations are also extracted using Stanford Parser. In order to avoid the data sparseness problem, parts of speech of *context-word* is limited to content word (i.e. noun, verb, adjective and adverb). For pre-training and post-training of word representations, we set the window size to 5 and dimensions of representation to 300 in CBOW algorithm.

4.2 Baseline Models

We compared our method to a baseline that generates a single word representation as well as methods that generate multiple word representations as below.

SGNS (Mikolov et al., 2013a)
Generates one word representation per word, which is the baseline in this experiment.

CBOW (Mikolov et al., 2013b)
Generates one word representation per word.

MSSG (Neelakantan et al., 2014)
Generates multiple word representations for each word by clustering based on the context similarity.

POS (Paetzold and Specia, 2016)
Generates multiple word representations for each word based on parts of speech.

TOPIC (Fadaee et al., 2017)
Generates multiple word representations for each word based on topics.

In addition, we evaluate variations of our method that use the lemma of a *context-word* and that use both the lemmas and parts of speech inspired by (Paetzold and Specia, 2016). For example, the representation of `soft` with the *context-word* of `cheese` is generated as `soft_adj_cheese_n`.

[2]https://dumps.wikimedia.org/enwiki/20170601/

snippet1	In 1955 the Soviet Union forwarded $ 100 million in **credit** to Afghanistan, which financed public transportation, airports, etc.
snippet2	Only congress has the authority to coin this **money** that should be used by the States.

Table 1: An example of Context-Aware Word Similarity Task. Words in **bold** are the targets.

5 Context-Aware Word Similarity Task

In order to evaluate the performance of word representations in a text analysis that needs to take ambiguity into consideration, we use the dataset of Stanford Contextual Word Similarity（SCWS) (Huang et al., 2012)[3]. In this dataset, a pair of target words and its surrounding snippets (50 preceding and following words) are given. The task is to estimate the similarity between targets in given snippets. Ten annotators were employed through Amazon Mechanical Turk and annotated similarity to 2,003 pairs of targets. As shown in Table 1, target pairs that are semantically close or distant in given snippets are both included in the dataset, so in this task, it is necessary to estimate similarities of targets considering their senses.

5.1 Evaluating Appropriateness of Word Similarity

First, we extract the *context-words* C_i and C_j of each target of w_i and w_j, respectively, from dependency relations in given snippets. Next, we obtain the representations of w_i and w_j, namely V_i and V_j, respectively, using the *context-words* of $c_i \in C_i$ and $c_j \in C_j$. Finally, the similarity is calculated between the two representations of targets using two measures as shown below.

Avg
In *Avg*, the average of the cosine similarity scores is used as the similarity measure between the target words. If there is more than one context-word, we can obtain several word

[3]`http://www-nlp.stanford.edu/~ehhuang/SCWS.zip`

representations for each *context-word* that potentially represent different word senses. We take all the word senses into account by taking the average of their similarity scores:

$$S_{\text{avg}} = \frac{1}{|V_i||V_j|} \sum_{v_i \in V_i, v_j \in V_j} \cos(v_i, v_j).$$

where $|\cdot|$ computes the size of a set and $\cos(\cdot, \cdot)$ calculates the cosine similarity between two vectors.

Max

In *Max*, the highest score among the cosine similarity scores which are calculated using each word representation is employed as the similarity measure between the target words. If a word has more than one context-word, the representations could be a mix of different senses. Hence, we use only the most similar representations indicated by the highest similarity score:

$$S_{\max} = \max_{v_i \in V_i, v_j \in V_j} \cos(v_i, v_j).$$

When any word representations given by *context-words* do not exist in our vocabulary, the similarity score cannot be obtained. In case no *context-word* is identified or when word representations given by *context-words* do not exist in our vocabulary, we return to using the cosine similarity between target words.

5.2 Results

Table 2 shows Spearman's rank correlation coefficient between each models and the manual scores in SCWS. Here, *Ours* uses only lemmas of *context-words* and *Ours+POS* uses both lemmas and parts of speech. *CBOW* and *POS* are our reproduction of CBOW algorithm and Paetzold and Specia (2016), respectively, trained with the same lemmatized training data as our methods. It is observed that the scores of the model using *context-words* are slightly higher than the models without *context-words*.

Table 3 shows the coverage of target word representations (with *context-words*) in our vocabulary. It is clear from this table that more than 80% of necessary representations are covered in our model even when combining parts of speech.

Model	Spearman's ρ
SGNS (Fadaee et al., 2017)	0.59
MSSG (Fadaee et al., 2017)	0.61
TOPIC (Fadaee et al., 2017)	0.61
CBOW	0.63
POS	0.64
Ours ($S_{\mathbf{avg}}$)	0.64
Ours ($S_{\mathbf{max}}$)	0.64
Ours+POS ($S_{\mathbf{avg}}$)	0.65
Ours+POS ($S_{\mathbf{max}}$)	0.64

Table 2: Spearman's rank correlation coefficient on SCWS dataset.

	In vocabulary / All tokens	
Ours	7297 / 8772	83.2%
Ours+POS	7058 / 8772	80.5%

Table 3: Coverage of target representations in our vocabulary on SCWS dataset.

6 Lexical Substitution Task

In order to evaluate the performance of word representations in a text generation that needs to take ambiguity into consideration, we use the following two datasets [4] for Lexical Substitution Task. In this task, in addition to the target word and its context, substitution candidates are given. The task is to rank candidates from the viewpoint of paraphrasability into the target word in a given context. Hence, this task also needs to capture word senses of word representation.

LS-SE

This is a dataset (McCarthy and Navigli, 2007) used in the Lexical Substitution Task of SemEval-2007. For each target word (201 types of targets in total), five annotators produce up to three types of substitutions in 10 different contexts.

LS-CIC

This is a large-scale dataset (Kremer et al., 2014) for Lexical Substitution Task. For

[4] https://github.com/stephenroller/naacl2016

$15,629$ target words, six annotators produce up to five types of substitutions under given context. Unlike LS-SE, three sentences are given as context; a sentence containing the target as well as its preceding and following sentences.

Following the previous studies (Melamud et al., 2015; Roller and Erk, 2016; Fadaee et al., 2017), substitution candidates are the union set of substitutions in each context with same target word. Therefore, these candidates are semantically and syntactically close to the target word in a certain context. Hence, it is important for this task to distinguish the meaning of target word using a given context.

We use Generalized Average Precision (GAP) (Kishida, 2005) as an evaluation metric. GAP is commonly used in evaluation of Lexical Substitution Task that calculates ranking accuracy by considering the weight of correct examples.

6.1 Evaluating Appropriateness of Lexical Substitution

In this task, we need to evaluate the appropriateness of substituting a target w_t with a candidate w_c considering the sentence s in which the target appears. We assume that candidates with representations that are substantially similar to those of the target are more appropriate for substitution. First, we identify *context-words* C_t of w_t based on dependency relations. Next, we obtain the representations of w_t as V_t for each *context-word* $c_t \in C_t$. We then obtain the representations of w_c as V_c by retrieving representations of w_c_c_t *s.t.* $c_t \in C_t$. Finally, we calculate the similarity between representations of $v_t \in V_t$ and $v_c \in V_c$ to rank the candidates using the following metrics:

Cos

This metric computes the cosine similarity of representations as $\cos(v_t, v_c)$.

balAddCos (Melamud et al., 2015)

This metric considers not only the similarity between v_t and v_c, but also between v_c and representations of words in s:

$$|s| \cos(v_t, v_c) + \sum_{w_i \in s} \cos(v_c, w_i)$$

At this stage, we ignore cases where representations of the words are out of the vocabulary.

When there are several context-words, we can obtain representations of w_t and w_c for each *context-word*. The straight-forward ways to compute the final similarity given multiple *context-words* are *Avg* and *Max* described in Section 5.1, which consider all possible combinations of representations.

In the Context-Aware Word Similarity Task, *context-words* of an input pair of words are completely independent. Unlike this, the *context-words* in Lexical Substitution Task are common between the target and candidate words. Hence, we expect that more accurate similarity can be computed by restricting pairs of representations to these based on the same *context-word*. Specifically, *Avg* is reformalized as *Avgc* as:

$$S_{\text{avgc}} = \frac{1}{|C_t|} \sum_{c_t \in C_t} \text{sim}(\text{vec}(w_t, c_t), \text{vec}(w_c, c_t)),$$

where $\text{sim}(\cdot, \cdot)$ is either *Cos* or *balAddCos*. The function $\text{vec}(\cdot, \cdot)$ obtains a word representation given a *context-word*. Similarly, *Max* is reformalized as *Maxc* as:

$$S_{\text{maxc}} = \max_{c_t \in C_t} \text{sim}(\text{vec}(w_t, c_t), \text{vec}(w_c, c_t)).$$

Based on the final similarity scores, we rank the candidates in descending order of the similarities.

6.2 Results

Table 4 shows GAP scores on LS-SE and LS-CIC datasets. When using *Cos* as a similarity metric, all of the proposed methods significantly outperformed previous methods. These results indicate that our approach generating multiple word representations per word using *context-words* is effective to capture word senses. Table 5 shows example results in which the meanings of a polysemous word `hard` were successfully captured by considering *context-words*. We can also see that the top-rated words in the output have the largest number of annotators who listed these words as substitutes. Figure 2 visualizes the actual word representations in the output of Table 5, where the vector dimensions are reduced to 2 by Principal Component Analysis. Circles represent the embeddings of pre-training and triangles represent the embeddings generated by post-training. It shows that embeddings

Model	LS-SE		LS-CIC	
	Cos	balAddCos	Cos	balAddCos
SGNS (Fadaee et al., 2017)	40.5	40.9	32.1	36.1
MSSG (Fadaee et al., 2017)	41.1	NA	37.8	NA
TOPIC (Fadaee et al., 2017)	NA	42.8	NA	40.9
CBOW	41.0	40.1	44.1	44.4
POS	41.8	42.1	46.5	46.7
Ours ($S_{\mathbf{avg}}$)	47.0	47.2	45.7	45.9
Ours ($S_{\mathbf{max}}$)	**48.4**	**48.5**	**46.4**	**46.5**
Ours+POS ($S_{\mathbf{avg}}$)	46.7	47.0	45.7	45.9
Ours+POS ($S_{\mathbf{max}}$)	47.7	48.1	46.3	**46.5**
Ours ($S_{\mathbf{avgc}}$)	47.4	47.7	46.1	46.2
Ours ($S_{\mathbf{maxc}}$)	**48.4**	**48.5**	46.4	46.5
Ours+POS ($S_{\mathbf{avgc}}$)	46.9	47.4	47.7	47.8
Ours+POS ($S_{\mathbf{maxc}}$)	47.7	48.0	**48.1**	**48.2**

Table 4: GAP scores on LS-SE and LS-CIC datasets, where highest scores are indicated in **bold**.

Input	... you are carrying on two conversations at once and you are required to *listen* **hard**.
Output	carefully (4), intensively (0), closely(1), intently(1), seriously (0), ...
Input	One event in particular *hits* the platoon **hard** : the death of its platoon leader, ...
Output	badly (3), heavily (0), strongly (0), severely (1), firmly (0), ...

Table 5: Example outputs in a Lexical Substitution Task. The target words in the input are presented in **bold** and *context-words* are presented in *italic*. The outputs are the ranked list of candidates, where the numbers in parentheses show the number of annotators who generated the word as one of the candidates for substitution.

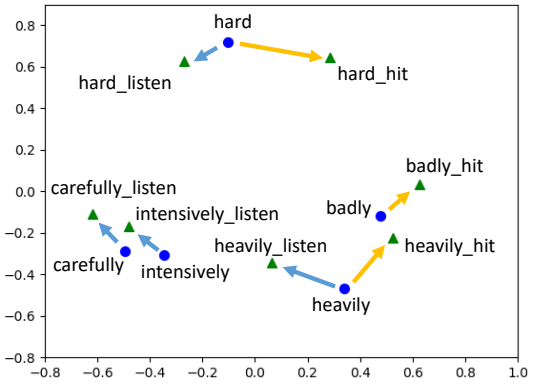

Figure 2: Visualization of word representations

of `hard_listen` and `carefully_listen` (appropriate substitution) get closer by deviating from the embeddings of `hard` and `carefully`, respectively. Similarly, embeddings of `hard_hit` and `badly_hit` get closer after post-training. Obviously, `hard_listen` and `hard_hit` have distinct embeddings, which demonstrates that our model successfully generates representations capturing different word senses.

When using *balAddCos* as a similarity metric, our method outperformed TOPIC. This result confirms that *context-words* are more effective to generate word representations considering word senses than topics.

When multiple *context-words* are available, *Maxc* performed best for both Ours and Ours+POS across LS-SE and LS-CIC. These results indicate that the similarity can be more accurately estimated using representations under the same *context-word* rather than considering any representations under diverse *context-words*. Another reason is that the effect of each representation can be diminished by taking the average of similarities. Even though there is a

		Coverage
LS-SE	Ours	82.2%
	Ours+POS	79.7%
LS-CIC	Ours	73.4%
	Ours+POS	71.7%

Table 6: Coverage of representations by our vocabulary

context-word that can pin-point the sense of the target word, its effect is blurred due to averaging.

When comparing the results of LS-SE and LS-CIC, the improvements in LS-SE is larger than that of the GAP score when using *context-words*. The primary factor of this difference is the number of polysemous words in the datasets. In LS-SE, we need to conduct lexical substitution for the same target word in 10 different contexts, which capturing of subtler differences in word senses compared to LS-CIC. Hence, the superior performance of our model on LS-SE shows its sensitivity representing word senses. Also, the coverage of target and candidate representations (with *context-words*) in our vocabulary is another factor for our different performances in LS-SE and LS-CIC. As shown in Table 6, LS-SE has higher vocabulary coverage than LS-CIC. In LS-SE dataset with higher vocabulary coverage, using context words greatly improved substitution performance.

Next, we examined the effects of *context-word* frequencies in the training data on the quality of generated word representations. We used *balAddCos* combined with *Maxc* as the similarity measure that showed the highest GAP scores in both datasets. We generated another two word representations; one generates representations using *context-words* whose frequency is less than 5, and the other generates representations using those of more than 100 in the training dataset. Table 7 shows GAP scores in LS-SE and LS-CIC using different representations, where CBOW generates plain word representations (without *context-words*) and POS generates representations combined with parts of speech. The GAP score improves even when using representations generated with *context-words* whose frequency is 5 or less. The improvement is larger when using representations generated with more frequent *context-words*. These results confirm the effectiveness of post-training in our method to alleviate the data sparseness problem. Moreover, the GAP score dramatically improved when using the representations generated considering all *context-words* regardless of their frequency. This indicates that our approach can effectively make use of *context-words* of low-frequency to generate representations to capture word senses.

		LS-SE	LS-CIC
CBOW		40.1	44.4
Ours	5 or less	41.6	44.3
	100 or more	43.3	45.6
	All	**48.5**	**46.5**
POS		42.1	46.7
Ours+POS	5 or less	42.8	46.7
	100 or more	43.6	47.6
	All	**48.0**	**48.2**

Table 7: The effect of *context-word* frequency, where highest scores are indicated in **bold**.

7 Conclusion

In this paper, we proposed methods that generate multiple word representations per word in order to capture its senses. Specifically, we assume that the senses of a word can be captured by *context-words* with dependency relations in a sentence. The extensive evaluation in Context-Aware Word Similarity Task and Lexical Substitution Task confirms the effectiveness of our methods and the detailed analysis in revealing their characteristics.

One of our future tasks is to use the entire surrounding context of a word, rather than single words using bi-directional recurrent neural networks. Also, we will extend our method to generate representations for multi-word expressions and phrases. In addition, we will apply our methods to diverse languages.

Acknowledgments

This research has been supported by the KDDI Foundation. We thank Professor Christopher G. Haswell for his valuable comments and discussions with us.

References

Ben Athiwaratkun and Andrew Gordon Wilson. 2017. Multimodal Word Distributions. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1645–1656.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, pages 135–146.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Learning Topic-Sensitive Word Representations. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 441–447.

Zellig S. Harris. 1954. Distributional Structure. *Word*, pages 146–162.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving Word Representations via Global Context and MultipleWord Prototypes. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 873–882.

Kazuaki Kishida. 2005. Property of Average Precision and its Generalization: An Examination of Evaluation Indicator for Information Retrieval Experiments. *NII Technical Reports*, pages 1–19.

Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What Substitutes Tell Us - Analysis of an"All-Words"Lexical Substitution Corpus. *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 540–549.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 302–308.

Jiwei Li and Dan Jurafsky. 2015. Do Multi-Sense Embeddings Improve Natural Language Understanding? *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1722–1732.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 55–60.

Diana McCarthy and Roberto Navigli. 2007. SemEval-2007 Task 10: English Lexical Substitution Task. *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 48–53.

Oren Melamud, Omer Levy, and Ido Dagan. 2015. A Simple Word Embedding Model for Lexical Substitution. *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7.

Tomas Mikolov and Tmikolov Google Com. 2014. Distributed Representations of Sentences and Documents. *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Distributed Representations of Words and Phrases and their Compositionality. *In Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Efficient Estimation of Word Representations in Vector Space. *Proceeding of the International Conference on Learning Representations*, pages 1–12.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space. *In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1059–1069.

Gustavo Henrique Paetzold and Lucia Specia. 2016. Unsupervised Lexical Simplification for Non-Native Speakers. *In Proceedings of AAAI Conference on Artificial Intelligence*, pages 3761–3767.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Stephen Roller and Katrin Erk. 2016. PIC a Different Word: A Simple Model for Lexical Substitution in Context. *The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1121–1126.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. *In Advances in Neural Information Processing Systems*, pages 1–9.