Composing Distributed Representations of Relational Patterns

Sho Takase **Naoaki Okazaki** **Kentaro Inui**
Graduate School of Information Sciences, Tohoku University
{takase, okazaki, inui}@ecei.tohoku.ac.jp

Abstract

Learning distributed representations for relation instances is a central technique in downstream NLP applications. In order to address semantic modeling of relational patterns, this paper constructs a new dataset that provides multiple similarity ratings for every pair of relational patterns on the existing dataset (Zeichner et al., 2012). In addition, we conduct a comparative study of different encoders including additive composition, RNN, LSTM, and GRU for composing distributed representations of relational patterns. We also present Gated Additive Composition, which is an enhancement of additive composition with the gating mechanism. Experiments show that the new dataset does not only enable detailed analyses of the different encoders, but also provides a gauge to predict successes of distributed representations of relational patterns in the relation classification task.

1 Introduction

Knowledge about entities and their relations (relation instances) are crucial for a wide spectrum of NLP applications, e.g., information retrieval, question answering, and recognizing textual entailment. Learning distributed representations for relation instances is a central technique in downstream applications as a number of recent studies demonstrated the usefulness of distributed representations for words (Mikolov et al., 2013; Pennington et al., 2014) and sentences (Sutskever et al., 2014; Cho et al., 2014; Kiros et al., 2015).

In particular, semantic modeling of relations and their textual realizations (*relational patterns* hereafter) is extremely important because a relation (e.g., causality) can be mentioned by various expressions (e.g., "*X* cause *Y*", "*X* lead to *Y*", "*Y* is associated with *X*"). To make matters worse, relational patterns are highly productive: we can produce a emphasized causality pattern "*X* increase the severe risk of *Y*" from "*X* increase the risk of *Y*" by inserting *severe* to the pattern.

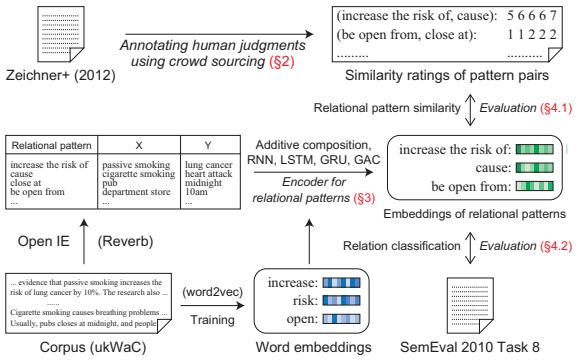

Figure 1: Overview of this study.

To model the meanings of relational patterns, the previous studies built a co-occurrence matrix between relational patterns (e.g., "*X* increase the risk of *Y*") and entity pairs (e.g., "*X*: smoking, *Y*: cancer") (Lin and Pantel, 2001; Nakashole et al., 2012). Based on the distributional hypothesis (Harris, 1954), we can compute a semantic vector of a relational pattern from the co-occurrence matrix, and measure the similarity of two relational patterns as the cosine similarity of the vectors. Nowadays, several studies adopt distributed representations computed by neural networks for semantic modeling of relational patterns (Yih et al., 2014; Takase et al., 2016).

Notwithstanding, the previous studies paid little attention to explicitly evaluate semantic modeling of relational patterns. In this paper, we construct a new dataset that contains a pair of relational patterns with five similarity ratings judged by human annotators. The new dataset shows a

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 2276–2286,
Berlin, Germany, August 7-12, 2016.

high inter-annotator agreement, following the annotation guideline of Mitchell and Lapata (2010). The dataset is publicly available on the Web site[1].

In addition, we conduct a comparative study of different encoders for composing distributed representations of relational patterns. During the comparative study, we present Gated Additive Composition, which is an enhancement of additive composition with the gating mechanism. We utilize the Skip-gram objective for training the parameters of the encoders on a large unlabeled corpus. Experiments show that the new dataset does not only enable detailed analyses of the different encoders, but also provides a gauge to predict successes of distributed representations of relational patterns in another task (relation classification). Figure 1 illustrates the overview of this study.

2 Data Construction

2.1 Target relation instances

We build a new dataset upon the work of Zeichner et al. (2012), which consists of relational patterns with semantic inference labels annotated. The dataset includes 5,555 pairs[2] extracted by Reverb (Fader et al., 2011), 2,447 pairs with inference relation and 3,108 pairs (the rest) without one.

Initially, we considered using this high-quality dataset as it is for semantic modeling of relational patterns. However, we found that inference relations exhibit quite different properties from those of semantic similarity. Take a relational pattern pair "*X* be the part of *Y*" and "*X* be an essential part of *Y*" filled with "*X* = the small intestine, *Y* = the digestive system" as an instance. The pattern "*X* be the part of *Y*" does not entail "*X* be an essential part of *Y*" because the meaning of the former does not include 'essential'. Nevertheless, both statements are similar, representing the same relation (PART-OF). Another uncomfortable pair is "*X* fall down *Y*" and "*X* go up *Y*" filled with "*X* = the dude, *Y* = the stairs". The dataset indicates that the former entails the latter probably because falling down from the stairs requires going up there, but they present the opposite meaning. For this reason, we decided to re-annotate semantic similarity judgments on every pair of relational patterns on the dataset.

2.2 Annotation guideline

We use instance-based judgment in a similar manner to that of Zeichner et al. (2012) to secure a high inter-annotator agreement. In instance-based judgment, an annotator judges a pair of relational patterns whose variable slots are filled with the same entity pair. In other words, he or she does not make a judgment for a pair of relational patterns with variables, "*X* prevent *Y*" and "*X* reduce the risk of *Y*", but two instantiated statements "Cephalexin prevent the bacteria" and "Cephalexin reduce the risk of the bacteria" ("*X* = Cephalexin, *Y* = the bacteria"). We use the entity pairs provided in Zeichner et al. (2012).

We asked annotators to make a judgment for a pair of relation instances by choosing a rating from 1 (dissimilar) to 7 (very similar). We provided the following instructions for judgment, which is compatible with Mitchell and Lapata (2010): (1) rate 6 or 7 if the meanings of two statements are the same or mostly the same (e.g., "Palmer team with Jack Nicklaus" and "Palmer join with Jack Nicklaus"); (2) rate 1 or 2 if two statements are dissimilar or unrelated (e.g., "the kids grow up with him" and "the kids forget about him"); (3) rate 3, 4, or 5 if two statements have some relationships (e.g., "Many of you know about the site" and "Many of you get more information about the site", where the two statements differ but also reasonably resemble to some extent).

2.3 Annotation procedure

We use a crowdsourcing service CrowdFlower[3] to collect similarity judgments from the crowds. CrowdFlower has the mechanism to assess the reliability of annotators using Gold Standard Data (Gold, hereafter), which consists of pairs of relational patterns with similarity scores assigned. Gold examples are regularly inserted throughout the judgment job to enable measurement of the performance of each worker[4]. Two authors of this paper annotated 100 pairs extracted randomly from 5,555 pairs, and prepared 80 Gold examples showing high agreement. Ratings of the Gold examples were used merely for quality assessment of the workers. In other words, we discarded the

[1] http://github.com/takase/relPatSim

[2] More precisely, the dataset includes 1,012 *meaningless* pairs in addition to 5,555 pairs. A pair of relational patterns was annotated as *meaningless* if the annotators were unable to understand the meaning of the patterns easily. We ignore the *meaningless* pairs in this study.

[3] http://www.crowdflower.com/

[4] We allow ± 1 differences in rating when we measure the performance of the workers.

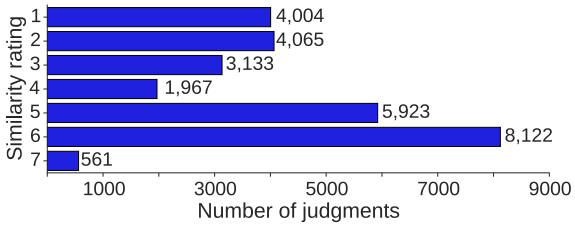

Figure 2: Number of judgments for each similarity rating. The total number of judgments is $27,775$ ($5,555$ pairs $\times$ 5 workers).

similarity ratings of the Gold examples, and used those judged by the workers.

To build a high quality dataset, we use judgments from workers whose confidence values (reliability scores) computed by CrowdFlower are greater than 75%. Additionally, we force every pair to have at least five judgments from the workers. Consequently, 60 workers participated in this job. In the final version of this dataset, each pair has five similarity ratings judged by the five most reliable workers who were involved in the pair.

Figure 2 presents the number of judgments for each similarity rating. Workers seldom rated 7 for a pair of relational patterns, probably because most pairs have at least one difference in content words. The mean of the standard deviations of similarity ratings of all pairs is 1.16. Moreover, we computed Spearman's ρ between similarity judgments from each worker and the mean of five judgments in the dataset. The mean of Spearman's ρ of workers involved in the dataset is 0.728. These statistics show a high inter-annotator agreement of the dataset.

3 Encoder for Relational Patterns

The new dataset built in the previous section raises two new questions — *What is the reasonable method (encoder) for computing the distributed representations of relational patterns? Is this dataset useful to predict successes of distributed representations of relational patterns in real applications?* In order to answer these questions, this section explores various methods for learning distributed representations of relational patterns.

3.1 Baseline methods without supervision

A naïve approach would be to regard a relational pattern as a single unit (word) and to train word/pattern embeddings as usual. In fact, Mikolov et al. (2013) implemented this approach as a preprocessing step, mining phrasal expressions with strong collocations from a training corpus. However, this approach might be affected by data sparseness, which lowers the quality of distributed representations.

Another simple but effective approach is *additive composition* (Mitchell and Lapata, 2010), where the distributed representation of a relational pattern is computed by the mean of embeddings of constituent words. Presuming that a relational pattern consists of a sequence of T words $w_1, ..., w_T$, then we let $x_t \in \mathbb{R}^d$ the embedding of the word w_t. This approach computes $\frac{1}{T}\sum_{t=1}^{T} x_t$ as the embedding of the relational pattern. Muraoka et al. (2014) reported that the additive composition is a strong baseline among various methods.

3.2 Recurrent Neural Network

Recently, a number of studies model semantic compositions of phrases and sentences by using (a variant of) Recurrent Neural Network (RNN) (Sutskever et al., 2014; Tang et al., 2015). For a given embedding x_t at position t, the vanilla RNN (Elman, 1990) computes the hidden state $h_t \in \mathbb{R}^d$ by the following recursive equation[5],

$$h_t = g(W_x x_t + W_h h_{t-1}). \quad (1)$$

Here, W_x and W_h are $d \times d$ matrices (parameters), $g(.)$ is the elementwise activation function ($\tanh$). We set $h_0 = 0$ at $t = 1$. In essence, RNN computes the hidden state h_t based on the one at the previous position (h_{t-1}) and the word embedding x_t. Applying Equation 1 from $t = 1$ to T, we use h_T as the distributed representation of the relational pattern.

3.3 RNN variants

We also employ Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014) as an encoder for relational patterns. LSTM has been applied successfully to various NLP tasks including word segmentation (Chen et al., 2015), dependency parsing (Dyer et al., 2015), machine translation (Sutskever et al., 2014), and sentiment analysis (Tai et al., 2015). GRU is also successful in machine translation (Cho et al., 2014) and various

[5]We do not use a bias term in this study. We set the number of dimensions of hidden states identical to that of word embeddings (d) so that we can adapt the objective function of the Skip-gram model for training (Section 3.5).

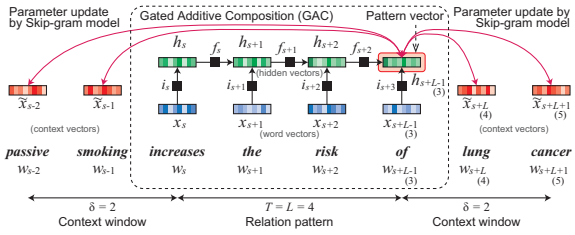

Figure 3: Overview of GAC trained with Skip-gram model. GAC computes the distributed representation of a relational pattern using the input gate and forget gate, and learns parameters by predicting surrounding words (Skip-gram model).

tasks including sentence similarity, paraphrase detection, and sentiment analysis (Kiros et al., 2015).

LSTM and GRU are similar in that the both architectures have gates (input, forget, and output for LSTM; reset and update for GRU) to remedy the gradient vanishing or explosion problem in training RNNs. Although some researchers reported that GRU is superior to LSTM (Chung et al., 2014), we have no consensus about the superiority. Besides, we are not sure whether LSTM or GRU is really necessary for relational patterns, which ususlly consist of a few words. Thus, we compare RNN, LSTM, and GRU empirically with the same training data and the same training procedure. Similarly to RNN, we use the hidden state h_T of LSTM[6] or GRU as the distributed representation of a relation pattern.

3.4 Gated Additive Composition (GAC)

In addition to the gradient problem, LSTM or GRU may be suitable for relational patterns, having the mechanism of adaptive control of gates for input words and hidden states. Consider the relational pattern "*X* have access to *Y*", whose meaning is mostly identical to that of "*X* access *Y*". Because 'have' in the pattern is a light verb, it may be harmful to incorporate the semantic vector of 'have' into the distributed representation of the pattern. The same may be true for the functional word 'to' in the pattern. However, the additive composition nor RNN does not have a mechanism to ignore the semantic vectors of these words. It is interesting to explore a method somewhere between additive composition and LSTM/GRU: additive composition with the gating mechanism.

For this reason, we present an another variant of RNN in this study. Inspired by the input and forget gates in LSTM, we compute the input gate $i_t \in \mathbb{R}^d$ and forget gate $f_t \in \mathbb{R}^d$ at position t. We use them to control the amount to propagate to the hidden state h_t from the current word x_t and the previous state h_{t-1}.

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1}) \quad (2)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1}) \quad (3)$$

$$h_t = g(f_t \odot h_{t-1} + i_t \odot x_t) \quad (4)$$

Here, W_{ix}, W_{ih}, W_{fx}, W_{fh} are $d \times d$ matrices. Equation 4 is interpreted as a weighted additive composition between the vector of the current word x_t and the vector of the previous hidden state h_{t-1}. The elementwise weights are controlled by the input gate i_t and forget gate f_t; we expect that input gates are closed (close to zero) and forget gates are opened (close to one) when the current word is a control verb or function word. We name this architecture *gated additive composition* (GAC).

3.5 Parameter estimation: Skip-gram model

To train the parameters of the encoders (RNN, LSTM, GRU, and GAC) on an unlabeled text corpus, we adapt the Skip-gram model (Mikolov et al., 2013). Formally, we designate an occurrence of a relational pattern p as a subsequence of L words $w_s, ..., w_{s+L-1}$ in a corpus. We define δ words appearing before and after pattern p as the context words, and let $C_p = (s-\delta, ..., s-1, s+L, ..., s+L+\delta)$ denote the indices of the context words. We define the log-likelihood of the relational pattern l_p, following the objective function of Skip-gram with negative sampling (SGNS) (Levy and Goldberg, 2014).

$$l_p = \sum_{\tau \in C_p} \left(\log \sigma(h_p^\top \tilde{x}_\tau) + \sum_{k=1}^{K} \log \sigma(-h_p^\top \tilde{x}_{\tilde{\tau}}) \right) \quad (5)$$

In this formula: K denotes the number of negative samples; $h_p \in \mathbb{R}^d$ is the vector for the relational pattern p computed by each encoder such as RNN; $\tilde{x}_\tau \in \mathbb{R}^d$ is the context vector for the word w_τ[7]; $x_{\tilde{\tau}'} \in \mathbb{R}^d$ is the context vector for the word

[6] We omitted peephole connections and bias terms.

[7] The Skip-gram model has two kinds of vectors x_t and $\tilde{x}_t$ assigned for a word w_t. Equation 2 of the original paper (Mikolov et al., 2013) denotes x_t (word vector) as v (input vector) and $\tilde{x}_t$ (context vector) as v' (output vector). The word2vec implementation does not write context (output) vectors but only word (input) vectors to a model file. Therefore, we modified the source code to save context vectors, and use them in Equation 5. This modification ensures the consistency of the entire model.

that were sampled from the unigram distribution[8] at every iteration of $\sum_k$.

At every occurrence of a relational pattern in the corpus, we use Stochastic Gradient Descent (SGD) and backpropagation through time (BPTT) for training the parameters (matrices) in encoders. More specifically, we initialize the word vectors x_t and context vectors $\tilde{x}_t$ with pre-trained values, and compute gradients for Equation 5 to update the parameters in encoders. In this way, each encoder is trained to compose a vector of a relational pattern so that it can predict the surrounding context words. An advantage of this parameter estimation is that the distributed representations of words and relational patterns stay in the same vector space. Figure 3 visualizes the training process for GAC.

4 Experiments

In Section 4.1, we investigate the performance of the distributed representations computed by different encoders on the pattern similarity task. Section 4.2 examines the contribution of the distributed representations on SemEval 2010 Task 8, and discusses the usefulness of the new dataset to predict successes of the relation classification task.

4.1 Relational pattern similarity

For every pair in the dataset built in Section 2, we compose the vectors of the two relational patterns using an encoder described in Section 3, and compute the cosine similarity of the two vectors. Repeating this process for all pairs in the dataset, we measure Spearman's ρ between the similarity values computed by the encoder and similarity ratings assigned by humans.

4.1.1 Training procedure

We used ukWaC[9] as the training corpus for the encoders. This corpus includes the text of 2 billion words from Web pages crawled in the .uk domain. Part-of-speech tags and lemmas are annotated by TreeTagger[10]. We used lowercased lemmas throughout the experiments. We apply word2vec to this corpus to pre-train word vectors x_t and context vectors $\tilde{x}_t$. All encoders use word vectors x_t to compose vectors of relational patterns; and the Skip-gram model uses context vectors $\tilde{x}_t$ to compute the objective function and gradients. We fix the vectors x_t and $\tilde{x}_t$ with pre-trained values during training.

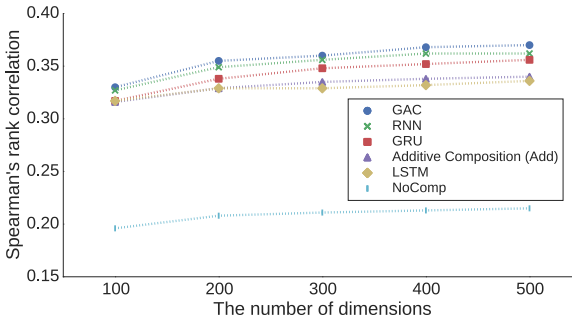

Figure 4: Performance of each method on the relational pattern similarity task with variation in the number of dimensions.

We used Reverb (Fader et al., 2011) to the ukWaC corpus to extract relational pattern candidates. To remove unuseful relational patterns, we applied filtering rules that are compatible with those used in the publicly available extraction result[11]. Additionally, we discarded relational patterns appearing in the evaluation dataset throughout the experiments to assess the performance under which an encoder composes vectors of unseen relational patterns. This preprocessing yielded $127,677$ relational patterns.

All encoders were implemented on Chainer[12], a flexible framework of neural networks. The hyperparameters of the Skip-gram model are identical to those in Mikolov et al. (2013): the width of context window $\delta = 5$, the number of negative samples $K = 5$, the subsampling of 10^{-5}. For each encoder that requires training, we tried 0.025, 0.0025, and 0.00025 as an initial learning rate, and selected the best value for the encoder. In contrast to the presentation of Section 3, we compose a pattern vector in backward order (from the last to the first) because preliminary experiments showed a slight improvement with this treatment.

4.1.2 Results and discussions

Figure 4 shows Spearman's rank correlations of different encoders when the number of dimensions of vectors is 100–500. The figure shows that GAC achieves the best performance on all dimensions.

Figure 4 includes the performance of the naïve approach, "NoComp", which regards a relational pattern as a single unit (word). In this approach,

[8] We use the probability distribution of words raised to the 3/4 power (Mikolov et al., 2013).

[9] http://wacky.sslmit.unibo.it

[10] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

[11] http://reverb.cs.washington.edu/

[12] http://chainer.org/

Length	#	NoComp	Add	LSTM	GRU	RNN	GAC
1	636	**0.324**	**0.324**	**0.324**	**0.324**	**0.324**	**0.324**
2	1,018	0.215	0.319	0.257	0.274	0.285	**0.321**
3	2,272	0.234	0.386	0.344	0.370	0.387	**0.404**
4	1,206	0.208	0.306	0.314	**0.329**	0.319	0.323
> 5	423	0.278	0.315	0.369	0.384	**0.394**	0.357
All	5,555	0.215	0.340	0.336	0.356	0.362	**0.370**

Table 1: Spearman's rank correlations on different pattern lengths (number of dimensions $d = 500$).

we allocated a vector h_p for each relational pattern p in Equation 5 instead of the vector composition, and trained the vectors of relational patterns using the Skip-gram model. The performance was poor for two reasons: we were unable to compute similarity values for 1,744 pairs because relational patterns in these pairs do not appear in ukWaC; and relational patterns could not obtain sufficient statistics because of data sparseness.

Table 1 reports Spearman's rank correlations computed for each pattern length. Here, the length of a relational-pattern pair is defined by the maximum of the lengths of two patterns in the pair. In length of 1, all methods achieve the same correlation score because they use the same word vector x_t. The table shows that additive composition (Add) performs well for shorter relational patterns (lengths of 2 and 3) but poorly for longer ones (lengths of 4 and 5+). GAC also exhibits the similar tendency to Add, but it outperforms Add for shorter patterns (lengths of 2 and 3) probably because of the adaptive control of input and forget gates. In contrast, RNN and its variants (RNN, GRU, and LSTM) enjoy the advantage on longer patterns (lengths of 4 and 5+).

To examine the roles of input and forget gates of GAC, we visualize the moments when input/forget gates are wide open or closed. More precisely, we extract the input word and scanned words when $|i_t|_2$ or $|f_t|_2$ is small (close to zero) or large (close to one) on the relational-pattern dataset. We restate that we compose a pattern vector in backward order (from the last to the first): GAC scans 'of', 'author', and 'be' in this order for composing the vector of the relational pattern 'be author of'.

Table 2 displays the top three examples identified using the procedure. The table shows two groups of tendencies. Input gates open and forget gates close when scanned words are only a preposition and the current word is a content word. In these situations, GAC tries to read the semantic vector of the content word and to ignore the semantic vector of the preposition. In contrast, input gates close and forget gates open when the current word is 'be' or 'a' and scanned words form a noun phrase (e.g., "charter member of"), a complement (e.g., "eligible to participate in"), or a passive voice (e.g., "require(d) to submit"). This behavior is also reasonable because GAC emphasizes informative words more than functional words.

	w_t	w_{t+1} w_{t+2} ...
large i_t (input open)	reimburse payable liable	for in to
small i_t (input close)	a a be	charter member of valuable member of an avid reader of
large f_t (forget open)	be be be	eligible to participate in require to submit request to submit
small f_t (forget close)	coauthor capital center	of of of

Table 2: Prominent moments for input/forget gates.

4.2 Relation classification

4.2.1 Experimental settings

To examine the usefulness of the dataset and distributed representations for a different application, we address the task of relation classification on the SemEval 2010 Task 8 dataset (Hendrickx et al., 2010). In other words, we explore whether high-quality distributed representations of relational patterns are effective to identify a relation type of an entity pair.

The dataset consists of $10,717$ relation instances ($8,000$ training and $2,717$ test instances) with their relation types annotated. The dataset

Method	Feature set	F1
SVM	BoW, POS	77.3
SVM + NoComp	embeddings, BoW, POS	79.9
SVM + LSTM	embeddings, BoW, POS	81.1
SVM + Add	embeddings, BoW, POS	81.1
SVM + GRU	embeddings, BoW, POS	81.4
SVM + RNN	embeddings, BoW, POS	81.7
SVM + GAC	embeddings, BoW, POS	82.0
	+ dependency, WordNet, NE	83.7
Ranking loss + GAC w/ fine-tuning	embeddings, BoW, POS	
	+ dependency, WordNet, NE	84.2
SVM (Rink and Harabagiu, 2010)	BoW, POS, dependency, Google n-gram, etc.	82.2
MV-RNN (Socher et al., 2012)	embeddings, parse trees	79.1
	+ WordNet, POS, NE	82.4
FCM (Gormley et al., 2015) w/o fine-tuning	embeddings, dependency	79.4
	+ WordNet	82.0
w/ fine-tuning	embeddings, dependency	82.2
	+ NE	83.4
RelEmb (Hashimoto et al., 2015)	embeddings	82.8
	+ dependency, WordNet, NE	83.5
CR-CNN (dos Santos et al., 2015) w/ Other	embeddings, word position embeddings	82.7
w/o Other	embeddings, word position embeddings	84.1
depLCNN (Xu et al., 2015)	embeddings, dependency	81.9
	+ WordNet	83.7
depLCNN + NS	embeddings, dependency	84.0
	+ WordNet	85.6

Table 3: F1 scores on the SemEval 2010 dataset.

defines 9 directed relations (e.g.,CAUSE-EFFECT) and 1 undirected relation OTHER. Given a pair of entity mentions, the task is to identify a relation type in 19 candidate labels (2×9 directed + 1 undirected relations). For example, given the pair of entity mentions e_1 = 'burst' and e_2 = 'pressure' in the sentence "The *burst* has been caused by water hammer *pressure*", a system is expected to predict CAUSE-EFFECT(e_2, e_1).

We used Support Vector Machines (SVM) with a Radial Basis Function (RBF) kernel implemented in libsvm[13]. Basic features are: part-of-speech tags (predicted by TreeTagger), surface forms, lemmas of words appearing between an entity pair, and lemmas of the words in the entity pair. Additionally, we incorporate distributed representations of a relational pattern, entities, and a word before and after the entity pair (number of dimensions $d = 500$). In this task, we regard words appearing between an entity pair as a relational pattern. We compare the vector representations of relational patterns computed by the five encoders presented in Section 4.1: additive composition, RNN, GRU, LSTM, and GAC. Hyperparameters related to SVM were tuned by 5-fold cross validation on the training data.

[13] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

4.2.2 Results and discussions

Table 3 presents the macro-averaged F1 scores on the SemEval 2010 Task 8 dataset. The first group of the table provides basic features and enhancements with the distributed representations. We can observe a significant improvement even from the distributed representation of NoComp (77.3 to 79.9). Moreover, the distributed representation that exhibited the high performance on the pattern similarity task was also successful on this task; GAC, which yielded the highest performance on the pattern similarity task, also achieved the best performance (82.0) of all encoders on this task.

It is noteworthy that the improvements brought by the different encoders on this task roughly cor-

respond to the performance on the pattern similarity task. This fact implies two potential impacts. First, the distributed representations of relational patterns are useful and easily transferable to other tasks such as knowledge base population. Second, the pattern similarity dataset provides a gauge to predict successes of distributed representations in another task.

We could further improve the performance of SVM + GAC by incorporating external resources in the similar manner as the previous studies did. Concretely, SVM + GAC achieved 83.7 F1 score by adding features for WordNet, named entities (NE), and dependency paths explained in Hashimoto et al. (2015). Moreover, we could obtain 84.2 F1 score, using the ranking based loss function (dos Santos et al., 2015) and fine-tuning of the distributed representations initially trained by GAC. Currently, this is the second best score among the performance values reported in the previous studies on this task (the second group of Table 3). If we could use the negative sampling technique proposed by Xu et al. (2015), we might improve the performance further[14].

5 Related Work

Mitchell and Lapata (2010) was a pioneering work in semantic modeling of short phrases. They constructed the dataset that contains two-word phrase pairs with semantic similarity judged by human annotators. Korkontzelos et al. (2013) provided a semantic similarity dataset with pairs of two words and a single word. Wieting et al. (2015) annotated a part of PPDB (Ganitkevitch et al., 2013) to evaluate semantic modeling of paraphrases. Although the target unit of semantic modeling is different from that for these previous studies, we follow the annotation guideline and instruction of Mitchell and Lapata (2010) to build the new dataset.

The task addressed in this paper is also related to the Semantic Textual Similarity (STS) task (Agirre et al., 2012). STS is the task to measure the degree of semantic similarity between two sentences. Even though a relational pattern appears as a part of a sentence, it may be difficult to transfer findings from one to another: for example, the encoders of RNN and its variants explored in this study may exhibit different characteristics, influenced by the length and complexity of input text expressions.

In addition to data construction, this paper addresses semantic modeling of relational patterns. Nakashole et al. (2012) approached the similar task by constructing a taxonomy of relational patterns. They represented a vector of a relational pattern as the distribution of entity pairs co-occurring with the relational pattern. Grycner et al. (2015) extended Nakashole et al. (2012) to generalize dimensions of the vector space (entity pairs) by incorporating hyponymy relation between entities. They also used external resources to recognize the transitivity of pattern pairs and applied transitivities to find patterns in entailment relation. These studies did not consider semantic composition of relational patterns. Thus, they might suffer from the data sparseness problem, as shown by NoComp in Figure 4.

Numerous studies have been aimed at encoding distributed representations of phrases and sentences from word embeddings by using: Recursive Neural Network (Socher et al., 2011), Matrix Vector Recursive Neural Network (Socher et al., 2012), Recursive Neural Network with different weight matrices corresponding to syntactic categories (Socher et al., 2013) or word types (Takase et al., 2016), RNN (Sutskever et al., 2011), LSTM (Sutskever et al., 2014), GRU (Cho et al., 2014), PAS-CLBLM (Hashimoto et al., 2014), etc. As described in Section 3, we applied RNN, GRU, and LSTM to compute distributed representations of relational patterns because recent papers have demonstrated their superiority in semantic composition (Sutskever et al., 2014; Tang et al., 2015). In this paper, we presented a comparative study of different encoders for semantic modeling of relational patterns.

To investigate usefulness of the distributed representations and the new dataset, we adopted the relation classification task (SemEval 2010 Task 8) as a real application. On the SemEval 2010 Task 8, several studies considered semantic composition. Gormley et al. (2015) proposed Feature-rich Compositional Embedding Model (FCM) that can combine binary features (e.g., positional indicators) with word embeddings via outer products. dos Santos et al. (2015) addressed the task using Convolutional Neural Network (CNN). Xu et al. (2015) achieved a higher performance than dos

[14] In fact, we made substantial efforts to introduce the negative sampling technique. However, Xu et al. (2015) omits the detail of the technique probably because of the severe page limit of short papers. For this reason, we could not reproduce their method in this study.

Santos et al. (2015) by application of CNN on dependency paths.

In addition to the relation classification task, we briefly describe other applications. To populate a knowledge base, Riedel et al. (2013) jointly learned latent feature vectors of entities, relational patterns, and relation types in the knowledge base. Toutanova et al. (2015) adapted CNN to capture the compositional structure of a relational pattern during the joint learning. For open domain question answering, Yih et al. (2014) proposed the method to map an interrogative sentence on an entity and a relation type contained in a knowledge base by using CNN.

Although these reports described good performance on the respective tasks, we are unsure of the generality of distributed representations trained for a specific task such as the relation classification. In contrast, this paper demonstrated the contribution of distributed representations trained in a generic manner (with the Skip-gram objective) to the task of relation classification.

6 Conclusion

In this paper, we addressed the semantic modeling of relational patterns. We introduced the new dataset in which humans rated multiple similarity scores for every pair of relational patterns on the dataset of semantic inference (Zeichner et al., 2012). Additionally, we explored different encoders for composing distributed representations of relational patterns. The experimental results shows that Gated Additive Composition (GAC), which is a combination of additive composition and the gating mechanism, is effective to compose distributed representations of relational patterns. Furthermore, we demonstrated that the presented dataset is useful to predict successes of the distributed representations in the relation classification task.

We expect that several further studies will use the new dataset not only for distributed representations of relational patterns but also for other NLP tasks (e.g., paraphrasing). Analyzing the internal mechanism of LSTM, GRU, and GAC, we plan to explore an alternative architecture of neural networks that is optimal for relational patterns.

Acknowledgments

We thank the reviewers and Jun Suzuki for valuable comments. This work was partially supported by Grant-in-Aid for JSPS Fellows Grant no. 26.5820, JSPS KAKENHI Grant number 15H05318, and JST, CREST.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *The First Joint Conference on Lexical and Computational Semantics (*SEM 2012)*, pages 385–393.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1197–1206.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1724–1734.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.

Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 626–634.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 334–343.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1535–1545.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, pages 758–764.

Matthew R. Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1774–1784.

Adam Grycner, Gerhard Weikum, Jay Pujara, James Foulds, and Lise Getoor. 2015. Relly: Inferring hypernym relationships between relational phrases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 971–981.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2014. Jointly learning word representations and composition functions using predicate-argument structures. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1544–1555.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2015. Task-oriented learning of word embeddings for semantic relation classification. In *Proceedings of the 19th Conference on Computational Natural Language Learning (CoNLL 2015)*, pages 268–278.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, pages 3276–3284.

Ioannis Korkontzelos, Torsten Zesch, Fabio Massimo Zanzotto, and Chris Biemann. 2013. Semeval-2013 task 5: Evaluating phrasal semantics. In *Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)*, pages 39–47.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 2177–2185.

Dekang Lin and Patrick Pantel. 2001. Dirt – discovery of inference rules from text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 01)*, pages 323–328.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, pages 3111–3119.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1439.

Masayasu Muraoka, Sonse Shimaoka, Kazeto Yamamoto, Yotaro Watanabe, Naoaki Okazaki, and Kentaro Inui. 2014. Finding the best model among representative compositional models. In *Proceedings of the 28th Pacific Asia Conference on Language, Information, and Computation (PACLIC 28)*, pages 65–74.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: A taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1135–1145.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, pages 74–84.

Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 256–259.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine learning (ICML 2011)*, pages 129–136.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 1201–1211.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, pages 455–465.

Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pages 1017–1024.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2015)*, pages 1556–1566.

Sho Takase, Naoaki Okazaki, and Kentaro Inui. 2016. Modeling semantic compositionality of relational patterns. *Engineering Applications of Artificial Intelligence*, 50(C):256–264.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1422–1432.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1499–1509.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics (TACL 2015)*, 3:345–358.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 536–540.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 643–648.

Naomi Zeichner, Jonathan Berant, and Ido Dagan. 2012. Crowdsourcing inference-rule evaluation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, pages 156–160.