Learning Word Representations from Relational Graphs

Danushka Bollegala   Takanori Maehara   Yuichi Yoshida   Ken-ichi Kawarabayashi

The University of Liverpool, Liverpool, L69 3BX, United Kingdom.
National Institute of Informatics, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan.
JST, ERATO, Kawarabayashi Large Graph Project.
Preferred Infrastructure, Inc., Hongo 2-40-1, Bunkyo-ku, Tokyo 113-0033, Japan.

Abstract

Attributes of words and relations between two words are central to numerous tasks in Artificial Intelligence such as knowledge representation, similarity measurement, and analogy detection. Often when two words share one or more attributes in common, they are connected by some semantic relations. On the other hand, if there are numerous semantic relations between two words, we can expect some of the attributes of one of the words to be inherited by the other. Motivated by this close connection between attributes and relations, given a relational graph in which words are interconnected via numerous semantic relations, we propose a method to learn a latent representation for the individual words. The proposed method considers not only the co-occurrences of words as done by existing approaches for word representation learning, but also the semantic relations in which two words co-occur. To evaluate the accuracy of the word representations learnt using the proposed method, we use the learnt word representations to solve semantic word analogy problems. Our experimental results show that it is possible to learn better word representations by using semantic semantics between words.

Introduction

The notions of attributes and relations are central to Artificial Intelligence. In Knowledge Representation (KR) (Brachman and Levesque 2004), a concept is described using its attributes and the relations it has with other concepts in a domain. If we already know a particular concept such as *pets*, we can describe a new concept such as *dogs* by stating the semantic relations that the new concept shares with the existing concepts such as *dogs* belongs-to *pets*. Alternatively, we could describe a novel concept by listing all the attributes it shares with existing concepts. In our example, we can describe the concept *dog* by listing attributes such as *mammal*, *carnivorous*, and *domestic animal* that it shares with another concept such as the *cat*. Therefore, both attributes and relations can be considered as alternative descriptors of the same knowledge. This close connection between attributes and relations can be seen in knowledge representation schemes such as predicate logic, where attributes

are modelled by predicates with a single argument whereas relations are modelled by predicates with two or more arguments.

Learning representations of words is an important task with numerous applications (Bengio *et al.* 2013). Better representations of words can improve the performance in numerous natural language processing tasks that require word representations such as language modelling (Collobert *et al.* 2011; Bengio *et al.* 2003), part-of-speech tagging (Zheng *et al.* 2013), sentiment classification (Socher *et al.* 2011b), and dependency parsing (Socher *et al.* 2013a; 2011a). For example, to classify a novel word into a set of existing categories one can measure the cosine similarity between the words in each category and the novel word to be classified. Next, the novel word can be assigned to the category of words that is most similar to (Huang *et al.* 2012). However, existing methods for learning word representations only consider the co-occurrences of two words within a short window of context, and ignore any semantic relations that exist between the two words.

Considering the close connection between attributes and relations, an obvious question is, *can we learn better word representations by considering the semantic relations that exist among words?* More importantly whether word representations learnt by considering the semantic relations among words could outperform methods that focus only on the co-occurrences of two words, ignoring the semantic relations. We study these problems in this paper and arrive at the conclusion that it is indeed possible to learn better word representations by considering the semantic relations between words. Specifically, given as input a relational graph, a directed labelled weighted graph where vertices represent words and edges represent numerous semantic relations that exist between the corresponding words, we consider the problem of learning a vector representation for each vertex (word) in the graph and a matrix representation for each label type (pattern). The learnt word representations are evaluated for their accuracy by using them to solve semantic word analogy questions on a benchmark dataset.

Our task of learning word attributes using relations between words is challenging because of several reasons. First, there can be multiple semantic relations between two words. For example, consider the two words *ostrich* and *bird*. An ostrich is-a-large bird as well as is-a-flightless bird. In this

regard, the relation between the two words *ostrich* and *bird* is similar to the relation between *lion* and *cat* (lion is a large cat) as well as to the relation between *penguin* and *bird* (penguin is a flightless bird). Second, a single semantic relation can be expressed using multiple lexical patterns. For example, the two lexical patterns **X** *is a large* **Y** and *large* **Y**s *such as* **X** represent the same semantic relation is-a-large. Beside lexical patterns, there are other representations of semantic relations between words such as POS patterns and dependency patterns. An attribute learning method that operates on semantic relations must be able to handle such complexities inherent in semantic relation representations. Third, the three-way co-occurrences between two words and a pattern describing a semantic relation is much sparser than the co-occurrences between two words (ignoring the semantic relations) or occurrences of individual words. This is particularly challenging for existing methods of representation learning where one must observe a sufficiently large number of co-occurrences to learn an accurate representation for words.

Given a *relational graph*, a directed labeled weighted graph describing various semantic relations that exist between words denoted by the vertices of the graph, we propose an unsupervised method to factorise the graph and assign latent attributional vectors $\boldsymbol{x}(u)$ for each vertex (word) u and a matrix $\mathbf{G}(l)$ to each label (pattern) l in the graph. Then, the co-occurrences between two words u and v in a pattern l that expresses some semantic relation is modelled as the scalar product $\boldsymbol{x}(u)^\top \mathbf{G}(l)\boldsymbol{x}(v)$. A regression model that minimises the squared loss between the predicted co-occurrences according to the proposed method and the actual co-occurrences in the corpus is learnt. In the relational graphs we construct, the edge between two vertices is labeled using the patterns that co-occur with the corresponding words, and the weight associated with an edge represents the strength of co-occurrence between the two words under the pattern.

Our proposed method does not assume any particular pattern extraction method or a co-occurrence weighting measure. Consequently, the proposed method can be applied to a wide-range of relational graphs, both manually created ones such as ontologies, as well as automatically extracted ones from unstructured texts. For concreteness of the presentation, we consider relational graphs where the semantic relations between words are represented using lexical patterns, part-of-speech (POS) patterns, or dependency patterns. Moreover, by adjusting the dimensionality of the decomposition, it is possible to obtain word representations at different granularities. Our experimental results show that the proposed method obtain robust performances over a wide-range of relational graphs constructed using different pattern types and co-occurrence weighting measures. It learns compact and dense word representations with as low as 200 dimensions. Unlike most existing methods for word representation learning, our proposed method considers the semantic relations that exist between two words in their co-occurring contexts. To evaluate the proposed method, we use the learnt word representations to solve semantic analogy problems in a benchmark dataset (Mikolov *et al.* 2013a).

Related Work

Representing the semantics of a word is a fundamental step in many NLP tasks. Given word-level representations, numerous methods have been proposed in compositional semantics to construct phrase-level, sentence-level, or document-level representations (Grefenstette 2013; Socher *et al.* 2012). Existing methods for creating word representations can be broadly categorised into two groups: *counting-based* methods, and *prediction-based* methods.

Counting-based approaches follow the distributional hypothesis (Firth 1957) which says that the meaning of a word can be represented using the co-occurrences it has with other words. By aggregating the words that occur within a pre-defined window of context surrounding all instances of a word from a corpus and by appropriately weighting the co-occurrences, it is possible to represent the semantics of a word. Numerous definitions of co-occurrence such as within a proximity window or involved in a particular dependency relation etc. and co-occurrence measures have been proposed in the literature (Baroni and Lenci 2010). This counting-based bottom-up approach often results in sparse word representations. Dimensionality reduction techniques such as the singular value decomposition (SVD) have been employed to overcome this problem in tasks such as measuring similarity between words using the learnt word representations (Turney and Pantel 2010).

Prediction-based approaches for learning word representations model the problem of representation learning as a prediction problem where the objective is to predict the presence (or absence) of a particular word in the context of another word. Each word w is assigned a feature vector $\boldsymbol{v}_w$ of fixed dimensionality such that the accuracy of predictions of other words made using $\boldsymbol{v}_w$ is maximised. Different objective functions for measuring the prediction accuracy such as perplexity or classification accuracy, and different optimisation methods have been proposed. For example, Neural Network Language Model (NLMM) (Bengio *et al.* 2003) learns word representations to minimise perplexity in a corpus. The Continuous Bag-Of-Words model (CBOW) (Mikolov *et al.* 2013b) uses the representations of all the words c in the context of w to predict the existence of w, whereas the skip-gram model (Mikolov *et al.* 2013c; 2013a) learns the representation of a word w by predicting the words c that appear in the surrounding context of w. Noise contrastive estimation has been used to speed-up the training of word occurrence probability models to learn word representations (Mnih and Kavukcuoglu 2013).

Given two words u and v represented respectively by vectors $\boldsymbol{x}(u)$ and $\boldsymbol{x}(v)$ of equal dimensions, GloVe (Pennington *et al.* 2014) learns a linear regression model to minimise the squared loss between the inner-product $\boldsymbol{x}(u)^\top \boldsymbol{x}(v)$ and the logarithm of the co-occurrence frequency of u and v. They show that this minimisation problem results in vector spaces that demonstrate linear relationships observed in word analogy questions. However, unlike our method, GloVe *does not* consider the semantic relations that exist between two words when they co-occur in a corpus. In particular, GloVe can be seen as a special case of our proposed method when we replace all patterns by a single pattern that indicates co-

occurrence, ignoring the semantic relations.

Our work in this paper can be categorised as a prediction-based method for word representation learning. However, prior work in prediction-based word representation learning have been limited to considering the co-occurrences between two words, ignoring any semantic relations that exist between the two words in their co-occurring context. On the other hand, prior studies on counting-based approaches show that specific co-occurrences denoted by dependency relations are particularly useful for creating semantic representations for words (Baroni and Lenci 2010). Interestingly, prediction-based approaches have shown to outperform counting-based approaches in comparable settings (Baroni *et al.* 2014). Therefore, it is natural for us to consider the incorporation of semantic relations between words into prediction-based word representation learning. However, as explained in the previous section, three-way co-occurrences of two words and semantic relations expressed by contextual patterns are problematic due to data sparseness. Therefore, it is non-trivial to extend the existing prediction-based word representation methods to three-way co-occurrences.

Methods that use matrices to represent adjectives (Baroni and Zamparelli 2010) or relations (Socher *et al.* 2013b) have been proposed. However, high dimensional representations are often difficult to learn because of their computational cost (Paperno *et al.* 2014). Although we learn matrix representations for patterns as a byproduct, our final goal is the vector representations for words. An interesting future research direction would be to investigate the possibilities of using the learnt matrix representations for related tasks such as relation clustering (Duc *et al.* 2010).

Learning Word Representations from Relational Graphs

Relational Graphs

We define a *relational graph* $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as a directed labelled weighted graph where the set of vertices $\mathcal{V}$ denotes words in the vocabulary, and the set of edges $\mathcal{E}$ denotes the co-occurrences between word-pairs and patterns. A pattern is a predicate of two arguments and expresses a semantic relation between the two words. Formally, an edge $e \in \mathcal{E}$ connecting two vertices $u, v \in \mathcal{V}$ in the relational graph $\mathcal{G}$ is a tuple $(u, v, l(e), w(e))$, where $l(e)$ denotes the label type corresponding to the pattern that co-occurs with the two words u and v in some context, and $w(e)$ denotes the co-occurrence strength between $l(e)$ and the word-pair (u, v). Each word in the vocabulary is represented by a unique vertex in the relational graph and each pattern is represented by a unique label type. Because of this one-to-one correspondence between words and vertices, and patterns and labels, we will interchange those terms in the subsequent discussions. The direction of an edge e is defined such that the first slot (ie. **X**) matches with u, and the second slot (ie. **Y**) matches with v in a pattern $l(e)$. Note that multiple edges can exist between two vertices in a relational graph corresponding to different patterns. Most manually created as well as
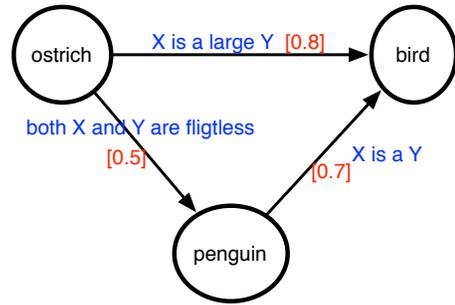

Figure 1: A relational graph between three words.

automatically extracted ontologies can be represented as relational graphs.

Consider the relational graph shown in Figure 1. For example, let us assume that we observed the context *ostrich is a large bird that lives in Africa* in a corpus. Then, we extract the lexical pattern ***X is a large Y*** between *ostrich* and *bird* from this context and include it in the relational graph by adding two vertices each for *ostrich* and *bird*, and an edge from *ostrich* to *bird*. Such lexical patterns have been used for related tasks such as measuring semantic similarity between words (Bollegala *et al.* 2007). The co-occurrence strength between a word-pair and a pattern can be computed using an association measure such as the positive pointwise mutual information (PPMI). Likewise, observation of the contexts *both ostrich and penguin are flightless birds* and *penguin is a bird* will result in the relational graph shown in Figure 1.

Learning Word Representations

Given a relational graph as the input, we learn d dimensional vector representations for each vertex in the graph. The dimensionality d of the vector space is a pre-defined parameter of the method, and by adjusting it one can obtain word representations at different granularities. Let us consider two vertices u and v connected by an edge with label l and weight w. We represent the two words u and v respectively by two vectors $\boldsymbol{x}(u), \boldsymbol{x}(v) \in \mathbb{R}^d$, and the label l by a matrix $\mathbf{G}(l) \in \mathbb{R}^{d \times d}$. We model the problem of learning optimal word representations $\hat{\boldsymbol{x}}(u)$ and pattern representations $\hat{\mathbf{G}}(l)$ as the solution to the following squared loss minimisation problem

$$\underset{\boldsymbol{x}(u) \in \mathbb{R}^d, \mathbf{G}(l) \in \mathbb{R}^{d \times d}}{\operatorname{argmin}} \frac{1}{2} \sum_{(u, v, l, w) \in \mathcal{E}} \left(\boldsymbol{x}(u)^\top \mathbf{G}(l) \boldsymbol{x}(v) - w \right)^2. \quad (1)$$

The objective function given by Eq. 1 is jointly non-convex in both word representations $\boldsymbol{x}(u)$ (or alternatively $\boldsymbol{x}(v)$) and pattern representations $\mathbf{G}(l)$. However, if $\mathbf{G}(l)$ is positive semidefinite, and one of the two variables is held fixed, then the objective function given by Eq. 1 becomes convex in the other variable. This enables us to use Alternating Least Squares (ALS) (Boyd *et al.* 2010) method to solve the optimisation problem. To derive the stochastic gradient descent (SGD) updates for the parameters in the model, let us denote the squared loss associated with a single edge $e = (u, v, l, w)$ in the relational graph $\mathcal{G}$ by $E(e)$, given

by,

$$E = \frac{1}{2} \sum_{(u,v,l,w) \in \mathcal{E}} (\boldsymbol{x}(u)^\top \mathbf{G}(l)\boldsymbol{x}(v) - w)^2. \quad (2)$$

In Eq. 2. The gradient of the error w.r.t. $\boldsymbol{x}(u)$ and $\mathbf{G}(l)$ are given by,

$$\nabla_{\boldsymbol{x}(u)} E = \sum_{(u,v,l,w) \in \mathcal{E}} (\boldsymbol{x}(u)^\top \mathbf{G}(l)\boldsymbol{x}(v) - w)\mathbf{G}(l)\boldsymbol{x}(v) + \sum_{(v,u,l,w) \in \mathcal{E}} (\boldsymbol{x}(v)^\top \mathbf{G}(l)\boldsymbol{x}(u) - w)\mathbf{G}(l)^\top \boldsymbol{x}(u) \quad (3)$$

$$\nabla_{\mathbf{G}(l)} E = \sum_{(u,v,l,w) \in \mathcal{E}} (\boldsymbol{x}(u)^\top \mathbf{G}(l)\boldsymbol{x}(v) - w)\boldsymbol{x}(v)\boldsymbol{x}(u)^\top \quad (4)$$

In Eq. 4, $\boldsymbol{x}(v)\boldsymbol{x}(u)^\top$ denotes the outer-product between the two vectors $\boldsymbol{x}(v)$ and $\boldsymbol{x}(u)$, which results in a $d \times d$ matrix. Note that the summation in Eq. 3 is taken over the edges that contain either u as a start or an end point, and the summation in Eq. 4 is taken over the edges that contain the label l.

The SGD update for the i-th dimension of $\boldsymbol{x}(u)$ is given by,

$$\boldsymbol{x}(u)_{(i)}^{(t+1)} = \boldsymbol{x}(u)_{(i)}^{(t)} - \frac{\eta_0}{\sqrt{\sum_{t'=1}^{t} \left(\nabla_{\boldsymbol{x}(u)^{(t')}} E\right)_{(i)}^2}} \left(\nabla_{\boldsymbol{x}(u)^{(t)}} E\right)_{(i)}. \quad (5)$$

Here, $\left(\nabla_{\boldsymbol{x}(u)^{(t)}} E\right)_{(i)}$ denotes the i-th dimension of the gradient vector of E w.r.t. $\boldsymbol{x}(u)^{(t)}$, and the superscripts (t) and $(t+1)$ denote respectively the current and the updated values. We use adaptive subgradient method (AdaGrad) (Duchi *et al.* 2011) to schedule the learning rate. The initial learning rate, η_0 is set to 0.0001 in our experiments.

Likewise, the SGD update for the (i,j) element of $\mathbf{G}(l)$ is given by,

$$\mathbf{G}(l)_{(i,j)}^{(t+1)} = \mathbf{G}(l)_{(i,j)}^{(t)} - \frac{\eta_0}{\sqrt{\sum_{t'=1}^{t} \left(\nabla_{\mathbf{G}(l)^{(t')}} E\right)_{(i,j)}^2}} \left(\nabla_{\mathbf{G}(l)^{(t)}} E\right)_{(i,j)}. \quad (6)$$

Recall that the positive semidefiniteness of $\mathbf{G}(l)$ is a requirement for the convergence of the above procedure. For example, if $\mathbf{G}(l)$ is constrained to be diagonal then this requirement can be trivially met. However, doing so implies that $\boldsymbol{x}(u)^\top \mathbf{G}(l)\boldsymbol{x}(v) = \boldsymbol{x}(v)^\top \mathbf{G}(l)\boldsymbol{x}(u)$, which means we can no longer capture asymmetric semantic relations in the model. Alternatively, without constraining $\mathbf{G}(l)$ to diagonal matrices, we numerically guarantee the positive semidefiniteness of $\mathbf{G}(l)$ by adding a small noise term $\delta\mathbf{I}$ after each update to $\mathbf{G}(l)$, where $\mathbf{I}$ is the $d \times d$ identity matrix and $\delta \in R$ is a small perturbation coefficient, which we set to 0.001 in our experiments.

Pseudo code for the our word representation learning algorithm is shown in Algorithm 1. Algorithm 1 initialises word and pattern representations randomly by sampling from the zero-mean and unit variance Gaussian distribution. Next, SGD updates are performed alternating between updates for $\boldsymbol{x}(u)$ and $\mathbf{G}(l)$ until a pre-defined number of maximum epochs is reached. Finally, the final values of $\boldsymbol{x}(u)$ and $\mathbf{G}(l)$ are returned.

Algorithm 1 Learning word representations.

Input: Relational graph $\mathcal{G}$, dimensionality d of the word representations, maximum epochs T, initial learning rate η_0.
Output: Word representations $\boldsymbol{x}(u)$ for words $u \in \mathcal{V}$.

1: **Initialisation:** For each vertex (word) $u \in \mathcal{V}$, randomly sample d dimensional real-valued vectors from the normal distribution. For each label (pattern) $l \in \mathcal{L}$, randomly sample $d \times d$ dimensional real-valued matrices from the normal distribution.

2: **for** $t = 1, \ldots, T$ **do**
3:     **for** edge $(u, v, l, w) \in \mathcal{G}$ **do**
4:         Update $\boldsymbol{x}(u)$ according to Eq. 5
5:         Update $\mathbf{G}(l)$ according to Eq. 6
6:     **end for**
7: **end for**
8: **return** $\boldsymbol{x}(u), \mathbf{G}(l)$

Experiments

Creating Relational Graphs

We use the English ukWaC[1] corpus in our experiments. ukWaC is a 2 billion token corpus constructed from the Web limiting the crawl to **.uk** domain and medium-frequency words from the British National Corpus (BNC). The corpus is lemmatised and Part-Of-Speech tagged using the TreeTagger[2]. Moreover, MaltParser[3] is used to create a dependency parsed version of the ukWaC corpus.

To create relational graphs, we first compute the co-occurrences of words in sentences in the ukWaC corpus. For two words u and v that co-occur in more than 100 sentences, we create two word-pairs (u, v) and (v, u). Considering the scale of the ukWaC corpus, low co-occurring words often represents misspellings or non-English terms.

Next, for each generated word-pair, we retrieve the set of sentences in which the two words co-occur. For explanation purposes let us assume that the two words u and v co-occur in a sentence s. We replace the occurrences of u in s by a slot marker **X** and v by **Y**. If there are multiple occurrences of u or v in s, we select the closest occurrences u and v measured by the number of tokens that appear in between the occurrences in s. Finally, we generate lexical patterns by limiting prefix (the tokens that appears before **X** in s), midfix (the tokens that appear in between **X** and **Y** in s), and suffix (the tokens that appear after **Y** in s) each separately to a maximum length of 3 tokens. For example, given the sentence *ostrich is a large bird that lives in Africa*, we will extract the lexical patterns **X** *is a large* **Y**, **X** *is a large* **Y** *that*, **X** *is a large* **Y** *that lives*, and **X** *is a large* **Y** *that lives in*. We select lexical patterns that co-occur with at least two word pairs for creating a relational graph.

In addition to lexical patterns, we generate POS patterns by replacing each lemma in a lexical pattern by its POS tag. POS patterns can be considered as an abstraction of the lexical patterns. Both lexical and POS patterns are unable to capture semantic relations between two words if those words

---

[1]http://wacky.sslmit.unibo.it/doku.php?id=corpora
[2]www.cis.uni-muenchen.de/~schmid/tools/TreeTagger
[3]www.maltparser.org

are located beyond the extraction window. One solution to this problem is to use the dependency path between the two words along the dependency tree for the sentence. We consider pairs of words that have a dependency relation between them in a sentence and extract dependency patterns such as **X** *direct-object-of* **Y**. Unlike lexical or POS patterns that are proximity-based, the dependency patterns are extracted from the entire sentence.

We create three types of relational graphs using $127,402$ lexical (**LEX**) patterns , $57,494$ POS (**POS**) patterns, and 6835 dependency (**DEP**) patterns as edge labels. Moreover, we consider several popular methods for weighting the co-occurrences between a word-pair and a pattern as follows.

RAW: The total number of sentences in which a pattern l co-occurs with a word-pair (u, v) is considered as the co-occurrence strength w.

PPMI: The positive pointwise mutual information between a pattern l and a word-pair (u, v) computed as,

$$\max\left(0, \log\left(\frac{h(u, v, l) \times h(*, *, *)}{h(u, v, *) \times h(*, *, l)}\right)\right).$$

Here, $h(u, v, l)$ denotes the total number of sentences in which u, v, and l co-occurs. The operator $*$ denotes the summation of h over the corresponding variables.

LMI: The local mutual information between a pattern l and a word pair (u, v) is computed as,

$$\frac{h(u, v, l)}{h(*, *, *)} \log\left(\frac{h(u, v, l) \times h(*, *, *)}{h(u, v, *) \times h(*, *, l)}\right).$$

LOG: This method considers the logarithm of the raw co-occurrence frequency as the co-occurrence strength. It has been shown to produce vector spaces that demonstrate vector substraction-based analogy representations (Pennington *et al.* 2014).

ENT: Patterns that co-occur with many word-pairs tend to be generic ones that does not express a specific semantic relation. Turney (2006) proposed the entropy of a pattern over word-pairs as a measure to down-weight the effect of such patterns. We use this entropy-based co-occurrence weighting method to weigh the edges in relational graphs.

Evaluation

We use the semantic word analogy dataset first proposed by Mikolov et al. (2013a) and has been used in much previous work for evaluating word representation methods. Unlike syntactic word analogies such as the past tense or plural forms of verbs which can be accurately captured using rule-based methods (Lepage 2000), semantic analogies are more difficult to detect using surface-level transformations. Therefore, we consider it is appropriate to evaluate word representation methods using semantic word analogies. The dataset contains 8869 word-pairs that represent word analogies covering various semantic relations such as the capital of a country (e.g. *Tokyo, Japan* vs. *Paris, France*), and family (gender) relationships (e.g. *boy, girl* vs. *king, queen*).

A word representation method is evaluated by its ability to correctly answer word analogy questions using the

Table 1: Semantic word analogy detection accuracy using word representations learnt by the proposed method from relational graphs with different pattern types and weighting measures.

Measure	LEX	POS	DEP
RAW	26.61	25.35	24.68
PPMI	25.00	25.51	24.48
LMI	27.15	25.19	24.41
LOG	23.01	25.55	24.47
ENT	22.39	25.55	24.52

word representations created by that method. For example, the semantic analogy dataset contains word pairs such as (*man*, woman) and (*king*, *queen*), where the semantic relations between the two words in the first pair is similar to that in the second. Denoting the representation of a word w by a vector $\boldsymbol{v}(w)$, we rank all words w in the vocabulary according to their cosine similarities with the vector, $\boldsymbol{v}(king) - \boldsymbol{v}(man) + \boldsymbol{v}(woman)$. The prediction is considered correct in this example only if the top most similar vector is $\boldsymbol{v}(queen)$. During evaluations, we limit the evaluation to analogy questions where word representations have been learnt for all the four words. Moreover, we remove three words that appear in the question from the set of candidate answers. The set of candidates for a question is therefore the set consisting of fourth words in all semantic analogy questions considered as valid (after the removal step described above), minus the three words in the question under consideration. The percentage of correctly answered semantic analogy questions out of the total number of questions in the dataset (ie. micro-averaged accuracy) is used as the evaluation measure.

Results

To evaluate the performance of the proposed method on relational graphs created using different pattern types and co-occurrence measures, we train 200 dimensional word representations ($d = 200$) using Algorithm 1. 100 iterations ($T = 100$) was sufficient to obtain convergence in all our experiments. We then used the learnt word representations to obtain the accuracy values shown in Table 1. We see that the proposed method obtains similar results with all pattern types and co-occurrence measures. This result shows the robustness of our method against a wide-range of typical methods for constructing relational graphs from unstructured texts. For the remainder of the experiments described in the paper, we use the RAW co-occurrence frequencies as the co-occurrence strength due to its simplicity.

To study the effect of the dimensionality d of the representation that we learn on the accuracy of the semantic analogy task, we plot the accuracy obtained using LEX, POS, and DEP relational graphs against the dimensionality of the representation as shown in Figure 2. We see that the accuracy steadily increases with the dimensionality of the representation up to 200 dimensions and then becomes relatively stable. This result suggests that it is sufficient to use 200 dimensions for all three types of relational graphs that we constructed. Interestingly, among the three pattern types,

Table 2: Comparison of the proposed method (denoted by **Prop**) against prior work on word representation learning.

| Method	capital-common	capital-world	city-in-state	family (gender)	currency	Overall Accuracy
SVD+LEX	11.43	5.43	0	9.52	0	3.84
SVD+POS	4.57	9.06	0	29.05	0	6.57
SVD+DEP	5.88	3.02	0	0	0	1.11
CBOW	8.49	5.26	4.95	47.82	2.37	10.58
skip-gram	9.15	9.34	5.97	**67.98**	5.29	14.86
GloVe	4.24	4.93	4.35	65.41	0	11.89
Prop+LEX	**22.87**	**31.42**	**15.83**	61.19	25.0	**26.61**
Prop+POS	22.55	30.82	14.98	60.48	20.0	25.35
Prop+DEP	20.92	31.40	15.27	56.19	20.0	24.68
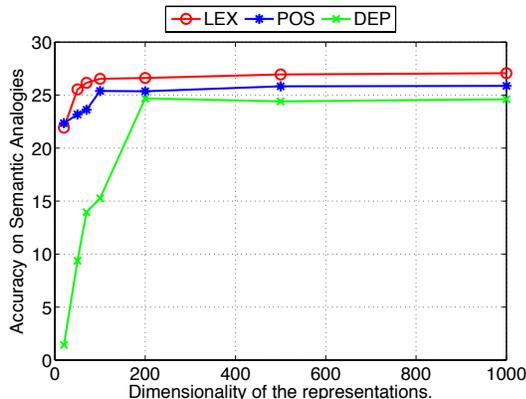
Figure 2: The effect of the dimensionality of the word representation on the accuracy of the semantic analogy task.

LEX stabilises with the least number of dimensions followed by POS and DEP. Note that LEX patterns have the greatest level of specificity compared to POS and DEP patterns, which abstract the surface-level lexical properties of the semantic relations. Moreover, relational graphs created with LEX patterns have the largest number of labels (patterns) followed by that with POS and DEP patterns. The ability of the proposed method to obtain better performances even with a highly specified, sparse and high-dimensional feature representations such as the LEX patterns is important when applying the proposed method to large relational graphs.

We compare the proposed method against several word representation methods in Table 2. All methods in Table 2 use 200 dimensional vectors to represent a word. A baseline method is created that shows the level of performance we can reach if we represent each word u as a vector of patterns l in which u occurs. First, we create a co-occurrence matrix between words u and patterns l, and use Singular Value Decomposition (SVD) to create 200 dimensional projections for the words. Because patterns represent contexts in which words appear in the corpus, this baseline can be seen as a version of the Latent Semantic Analysis (LSA), that has been widely used to represent words and documents in information retrieval. Moreover, SVD reduces the data sparseness in raw co-occurrences. We create three versions of this baseline denoted by **SVD+LEX**, **SVD+POS**, and **SVD+DEP** corresponding to relational graphs created using respectively LEX, POS, and DEP patterns. **CBOW** (Mikolov *et al.* 2013b), **skip-gram** (Mikolov *et al.* 2013c), and **GloVe** (Pen-

nington *et al.* 2014) are previously proposed word representation learning methods. In particular, **skip-gram** and **GloVe** are considered the current state-of-the-art methods. We learn 200 dimensional word representations using their original implementations with the default settings. We used the same set of sentences as used by the proposed method to train these methods. Proposed method is trained using 200 dimensions and with three relational graphs (denoted by **Prop+LEX**, **Prop+POS**, and **Prop+DEP**), weighted by RAW co-occurrences.

From Table 2, we see that **Prop+LEX** obtains the best overall results among all the methods compared. We note that the previously published results for skip-gram and CBOW methods are obtained using a 100B token news corpus, which is significantly large than the 2B token ukWaC corpus used in our experiments. However, the differences among the **Prop+LEX**, **Prop+POS**, and **Prop+DEP** methods are not significantly different according to the Binomial exact test. SVD-based baseline methods perform poorly indicating that de-coupling the 3-way co-occurrences between (u, v) and l into 2-way co-occurrences between u and l is inadequate to capture the semantic relations between words. **Prop+LEX** reports the best results for all semantic relations except for the *family* relation. Comparatively, higher accuracies are reported for the family relation by all methods, whereas relations that involve named-entities such as locations are difficult to process. Multiple relations can exist between two locations, which makes the analogy detection task hard.

Conclusion

We proposed a method that considers not only the co-occurrences of two words but also the semantic relations in which they co-occur to learn word representations. It can be applied to manually created relational graphs such as ontologies, as well as automatically extracted relational graphs from text corpora. We used the proposed method to learn word representations from three types of relational graphs. We used the learnt word representations to answer semantic word analogy questions using a previously proposed dataset. Our experimental results show that lexical patterns are particularly useful for learning good word representations, outperforming several baseline methods. We hope our work will inspire future research in word representation learning to exploit the rich semantic relations that exist between words, extending beyond simple co-occurrences.

References

Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673 – 721, 2010.

Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *EMNLP*, pages 1183 – 1193, 2010.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*, pages 238–247, 2014.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137 – 1155, 2003.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798 – 1828, March 2013.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. Websim: A web-based semantic similarity measure. In *Proc. of 21st Annual Conference of the Japanese Society of Artitificial Intelligence*, pages 757 – 766, 2007.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Exkstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1 – 122, 2010.

Ronald Brachman and Hector J. Levesque. *Knowledge Representation and Reasoning*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann Publishers Inc., June 2004.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuska. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493 – 2537, 2011.

Nguyen Tuan Duc, Danushka Bollegala, and Mitsuru Ishizuka. Using relational similarity between word pairs for latent relational search on the web. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 196 – 199, 2010.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121 – 2159, July 2011.

John R. Firth. A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis*, pages 1 – 32, 1957.

Edward Grefenstette. Towards a formal distributional semantics: Simulating logical calculi with tensors. In **SEM*, pages 1 – 10, 2013.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Improving word representations via global context and multiple word prototypes. In *ACL*, pages 873 – 882, 2012.

Yves Lepage. Languages of analogical strings. In *ACL*, pages 488 – 494, 2000.

Tomas Mikolov, Kai Chen, and Jeffrey Dean. Efficient estimation of word representation in vector space. *CoRR*, abs/1301.3781, 2013.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111 – 3119, 2013.

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continous space word representations. In *NAACL*, pages 746 – 751, 2013.

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In *NIPS*, 2013.

Denis Paperno, Nghia The Pham, and Marco Baroni. A practical and linguistically-motivated approach to compositional distributional semantics. In *ACL*, pages 90–99, 2014.

Jeffery Pennington, Richard Socher, and Christopher D. Manning. Glove: global vectors for word representation. In *EMNLP*, 2014.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Ng, and Chris Manning. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP*, pages 151–161, 2011.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP*, pages 1201–1211, 2012.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. Parsing with compositional vector grammars. In *ACL*, pages 455–465, 2013.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, 2013.

Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Aritificial Intelligence Research*, 37:141 – 188, 2010.

P.D. Turney. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416, 2006.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. Deep learning for Chinese word segmentation and POS tagging. In *EMNLP*, pages 647–657, 2013.