Updated Version – October 16, 2013

Multi-Relational Latent Semantic Analysis

Kai-Wei Chang*
University of Illinois
Urbana, IL 61801, USA
kchang10@illinois.edu

Wen-tau Yih **Christopher Meek**
Microsoft Research
Redmond, WA 98052, USA
{scottyih,meek}@microsoft.com

Abstract

We present Multi-Relational Latent Semantic Analysis (MRLSA) which generalizes Latent Semantic Analysis (LSA). MRLSA provides an elegant approach to combining multiple relations between words by constructing a 3-way tensor. Similar to LSA, a low-rank approximation of the tensor is derived using a tensor decomposition. Each word in the vocabulary is thus represented by a vector in the latent semantic space and each relation is captured by a latent square matrix. The degree of two words having a specific relation can then be measured through simple linear algebraic operations. We demonstrate that by integrating multiple relations from both homogeneous and heterogeneous information sources, MRLSA achieves state-of-the-art performance on existing benchmark datasets for two relations, *antonymy* and *is-a*.

1 Introduction

Continuous semantic space representations have proven successful in a wide variety of NLP and IR applications, such as document clustering (Xu et al., 2003) and cross-lingual document retrieval (Dumais et al., 1997; Platt et al., 2010) at the document level and sentential semantics (Guo and Diab, 2012; Guo and Diab, 2013) and syntactic parsing (Socher et al., 2013) at the sentence level. Such representations also play an important role in applications for lexical semantics, such as word sense disambiguation (Boyd-Graber et al., 2007), measuring word similarity (Deerwester et al., 1990) and relational similarity (Turney, 2006; Zhila et al., 2013; Mikolov et al., 2013). In many of these applications, Latent Semantic Analysis (LSA) (Deerwester et al., 1990) has been widely used, serving as a fundamental component or as a strong baseline.

LSA operates by mapping text objects, typically documents and words, to a latent semantic space. The proximity of the vectors in this space implies that the original text objects are semantically related. However, one well-known limitation of LSA is that it is unable to differentiate fine-grained relations. For instance, when applied to lexical semantics, synonyms and antonyms may both be assigned high similarity scores (Landauer and Laham, 1998; Landauer, 2002). Asymmetric relations like hyponyms and hypernyms also cannot be differentiated. Although there exists some recent work, such as PILSA which tries to overcome this weakness of LSA by introducing the notion of polarity (Yih et al., 2012). This extension, however, can only handle two opposing relations (e.g., synonyms and antonyms), leaving open the challenge of encoding multiple relations.

In this paper, we propose Multi-Relational Latent Semantic Analysis (MRLSA), which strictly generalizes LSA to incorporate information of multiple relations concurrently. Similar to LSA or PILSA when applied to lexical semantics, each word is still mapped to a vector in the latent space. However, when measuring whether two words have a specific relation (e.g., antonymy or *is-a*), the word vectors will be mapped to a new space according to the relation where the degree of having this relation will be

*Work conducted while interning at Microsoft Research.

judged by cosine similarity. The raw data construction in MRLSA is straightforward and similar to the document-term matrix in LSA. However, instead of using one matrix to capture all relations, we extend the representation to a 3-way tensor. Each slice corresponds to the document-term matrix in the original LSA design but for a specific relation. Analogous to LSA, the whole linear transformation mapping is derived through tensor decomposition, which provides a low-rank approximation of the original tensor. As a result, previously unseen relations between two words can be discovered, and the information encoded in other relations can influence the construction of the latent representations, and thus potentially improves the overall quality. In addition, the information in different slices can come from heterogeneous sources (conceptually similar to (Riedel et al., 2013)), which not only improves the model, but also extends the word coverage in a reliable way.

We provide empirical evidence that MRLSA is effective using two different word relations: *antonymy* and *is-a*. We use the benchmark GRE test of closest-opposites (Mohammad et al., 2008) to show that MRLSA performs comparably to PILSA, which was the pervious state-of-the-art approach on this problem, when given the same amount of information. In addition, when other words and relations are available, potentially from additional resources, MRLSA is able to outperform previous methods significantly. We use the *is-a* relation to demonstrate that MRLSA is capable of handling asymmetric relations. We take the list of word pairs from the *Class-Inclusion* (i.e., *is-a*) relations in SemEval-2012 Task 2 (Jurgens et al., 2012), and use our model to measure the degree of two words have this relation. The measures derived from our model correlate with human judgement better than the best system that participated in the task.

The rest of this paper is organized as follows. We first survey some related work in Section 2, followed by a more detailed description of LSA and PILSA in Section 3. Our proposed model, MRLSA, is presented in Section 4. Section 5 presents our experimental results. Finally, Section 6 concludes the paper.

2 Related Work

MRLSA can be viewed as a model that derives general continuous space *representations* for capturing *lexical semantics*, with the help of *tensor decomposition* techniques. We highlight some recent work related to our approach.

The most commonly used continuous space representation of text is arguably the vector space model (VSM) (Turney and Pantel, 2010). In this representation, each text object can be represented by a high-dimensional sparse vector, such as a term-vector or a document-vector that denotes the statistics of term occurrences (Salton et al., 1975) in a large corpus. The text can also be represented by a low-dimensional dense vector derived by linear projection models like latent semantic analysis (LSA) (Deerwester et al., 1990), by discriminative learning methods like Siamese neural networks (Yih et al., 2011), recurrent neural networks (Mikolov et al., 2013) and recursive neural networks (Socher et al., 2011), or by graphical models such as probabilistic latent semantic analysis (PLSA) (Hofmann, 1999) and latent Dirichlet allocation (LDA) (Blei et al., 2003). As a generalization of LSA, MRLSA is also a linear projection model. However, while the words are represented by vectors as well, multiple relations between words are captured separately by matrices.

In the context of lexical semantics, VSMs provide a natural way of measuring semantic word relatedness by computing the distance between the corresponding vectors, which has been a standard approach (Agirre et al., 2009; Reisinger and Mooney, 2010; Yih and Qazvinian, 2012). These approaches do not apply directly to the problem of modeling other types of relations. Existing methods that do handle multiple relations often use a model combination scheme to integrate signals from various types of information sources. For instance, morphological variations discovered from the Google n-gram corpus have been combined with information from thesauri and vector-based word relatedness models for detecting antonyms (Mohammad et al., 2008). An alternative approach proposed by Turney (2008) that handles synonyms, antonyms and associations is to use a uniform approach by first reducing the problem to determining whether two

pairs of words can be analogous, and then predicting it using a supervised model with features based on the frequencies of patterns in the corpus. Similarly, to measure whether two word pairs have the same relation, Zhila et al. (2013) proposed to combine heterogeneous models, which achieved state-of-the-art performance. In comparison, MRLSA models multiple lexical relations holistically. The degree that two words having a particular relation is estimated using the same linear function of the corresponding vectors and matrix.

Tensor decomposition generalizes matrix factorization and has been applied to several NLP applications recently. For example, Cohen et al. (2013) proposed an approximation algorithm for PCFG parsing that relies on Kruskal decomposition. Van de Cruys et al. (2013) modeled the composition of subject-verb-object triples using Tucker decomposition, which results in a better similarity measure for transitive phrases. Similar to this construction but used in the community-based question answering (CQA) scenario, Qiu et al. (2013) represented triples of question title, question content and answer as a tensor and applied 3-mode SVD to derive latent semantic representations for question matching. The construction of MRLSA bears some resemblance to the work that use tensors to capture triples. However, our goal of modeling different relations for lexical semantics is very different from the intended usage of tensor decomposition in the existing work.

3 Latent Semantic Analysis

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is a widely used continuous vector space model that maps words and documents into a low dimensional space. LSA consists of two main steps. First, taking a collection of d documents that contains words from a vocabulary list of size n, it first constructs a $d \times n$ document-term matrix $\mathbf{W}$ to encode the occurrence information of a word in a document. For instance, in its simplest form, the element $\mathbf{W}_{i,j}$ can be the term frequency of the j-th word in the i-th document. In practice, a weighting scheme that better captures the importance of a word in the document, such as TF×IDF (Salton et al., 1975), is often used instead. Notice that "document" here simply means a group of words and has been applied to various texts including news articles, sentences and bags of words. Once the matrix is constructed, the second step is to apply singular value decomposition (SVD) to $\mathbf{W}$ in order to derive a low-rank approximation. To have a rank-k approximation, $\mathbf{X}$ is the reconstruction matrix of $\mathbf{W}$, defined as

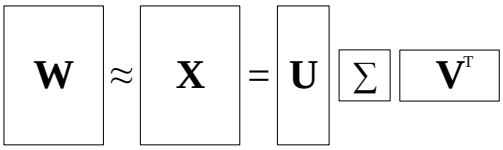


Figure 1: SVD applied to a $d \times n$ document-term matrix $\mathbf{W}$. The rank-k approximation, $\mathbf{X}$, is the multiplication of $\mathbf{U}$, $\mathbf{\Sigma}$ and $\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are $d \times k$ and $n \times k$ orthonormal matrices and $\mathbf{\Sigma}$ is a $k \times k$ diagonal matrix. The column vectors of $\mathbf{V}^T$ multiplied by the singular values $\mathbf{\Sigma}$ represent words in the latent semantic space.

$$\mathbf{W} \approx \mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (1)$$

where the dimensions of $\mathbf{U}$ and $\mathbf{V}$ are $d \times k$ and $n \times k$, respectively, and $\mathbf{\Sigma}$ is a $k \times k$ diagonal matrix. In addition, the columns in $\mathbf{U}$ and $\mathbf{V}$ are orthonormal and the elements in $\mathbf{\Sigma}$ are the singular values and are conventionally reverse-ordered. Figure 1 illustrates this decomposition.

LSA can be used to compute the similarity between two documents or two words in the latent space. For instance, to compare the u-th and v-th words in the vocabulary, one can compute the cosine similarity of the u-th and v-th column vectors of $\mathbf{X}$, the reconstruction matrix of $\mathbf{W}$. In contrast to a direct lexical matching via the columns of $\mathbf{W}$, the similarity measure computed as a result of the SVD may have a nonzero similarity score even if these two words do not co-occur in any documents. This is due to the fact that those words can share some latent components.

An alternative view of using LSA is to treat the column vectors of $\mathbf{\Sigma}\mathbf{V}^T$ as a representation of the words in a new k-dimensional latent space. This comes from the observation that the inner product of every two column vectors in $\mathbf{X}$ is the inner product of the corresponding column vectors of $\mathbf{\Sigma}\mathbf{V}^T$,

Figure 2: The matrix construction of PILSA. The vocabulary is {joy, gladden, sorrow, sadden, anger, emotion, feeling} and target words are {joyfulness, gladden, sad, anger}. For ease of presentation, we show the numbers with 0-1 values instead of TF×IDF scores. The polarity (i.e., sign) indicates whether the term in the vocabulary is a synonym or antonym of the target word.

which can be derived from the equations below.

$$\begin{aligned}\mathbf{X}^T\mathbf{X} &= (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T) \\ &= \mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \quad (\mathbf{\Sigma} \text{ is diagonal}) \\ &= \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^T \quad (\text{Columns of } \mathbf{U} \text{ are orthonormal}) \\ &= (\mathbf{\Sigma}\mathbf{V}^T)^T(\mathbf{\Sigma}\mathbf{V}^T) \end{aligned} \tag{2}$$

Thus, the semantic relatedness between the i-th and j-th words can be computed by cosine similarity[1]:

$$\cos(\mathbf{X}_{:,i}, \mathbf{X}_{:,j}) \tag{3}$$

When used to compare words, one well-known limitation of LSA is that the score captures the general notion of semantic similarity, and is unable to distinguish fine-grained word relations, such as antonyms (Landauer and Laham, 1998; Landauer, 2002). This is due to the fact that the raw matrix representation only records the occurrences of words in documents without knowing the specific relation between the word and document. To address this issue, Yih et al. (2012) proposed a polarity inducing latent semantic analysis model recently, which we introduce next.

[1]Cosine similarity is equivalent to the inner product of the normalized vectors.

3.1 Polarity Inducing Latent Semantic Analysis

In order to distinguish antonyms from synonyms, the polarity inducing LSA (PILSA) model (Yih et al., 2012) takes a thesaurus as input. Synonyms and antonyms of the same target word are grouped together as a "document" and a document-term matrix is constructed accordingly as done in LSA. Because each word in a group belongs to either one of the two opposite relations, *synonymy* and *antonymy*, the *polarity* information is induced by flipping the signs of antonyms. While the absolute value of each element in the matrix is still the same TF×IDF score, the elements that correspond to the antonyms become negative.

This design has an intriguing effect. When comparing two words using the cosine similarity (or simply inner product) of their corresponding column vectors in the matrix, the score of a synonym pair remains positive, but the score of an antonym pair becomes negative. Figure 2 illustrates this design using a simplified matrix as example.

Once the matrix is constructed, PILSA applies SVD as done in LSA, which generalizes the model to go beyond lexical matching. The sign of the cosine score of the column vectors of any two words indicates whether they are close to synonyms or to antonyms and the absolute value reflects the degree of the relation. When all the column vectors are normalized to unit vectors, it can also be viewed as synonyms are clustered together and antonyms lie on the opposite sides of a unit sphere. Although PILSA successfully extends LSA to handle not just one single *occurrence* relation, the extension is limited to encoding two opposing relations

4 Multi-Relational Latent Semantic Analysis

The fundamental reason why it is difficult to handle multiple relations is due to the 2-dimensional matrix representation. In order to overcome this, we encode the raw data in a 3-way tensor. Each slice captures a particular relation and is in the format of the document-term matrix in LSA. Just as in LSA, where the low-rank approximation by SVD helps generalize the representation and discover unseen relations, we apply a tensor decomposition method,

(a) Synonym layer (b) Antonym layer (c) Hypernym layer

Figure 3: The three slices of MRLSA raw tensor $\mathcal{W}$ for an example with vocabulary {joy, gladden, sorrow, sadden, anger, emotion, feeling} and target words {joyfulness, gladden, sad, anger}. Figures 3(a), 3(b), 3(c) show the matrices $\mathcal{W}_{:,:,\text{syn}}$, $\mathcal{W}_{:,:,\text{ant}}$, $\mathcal{W}_{:,:,\text{hyper}}$, respectively. Rows represent documents (see definition in text), and columns represent words. For ease of presentation, we show numbers with 0-1 values instead of TF×IDF scores.

the Tucker decomposition, to the tensor.

4.1 Representing Multi-Relational Data in Tensors

A tensor is simply a multi-dimensional array. In this work, we use a 3-way tensor $\mathcal{W}$ to encode multiple word relations. An element of $\mathcal{W}$ is denoted by $\mathcal{W}_{i,j,k}$ using its indices, and $\mathcal{W}_{:,:,k}$ represents the k-th *slice* of $\mathcal{W}$ (a slice of a 3-way tensor is a matrix, obtained by fixing the third index). Following (Kolda and Bader, 2009), a *fiber* of a tensor $\mathcal{W}_{:,j,k}$ is a vector, which is a high order analog of a matrix row or column.

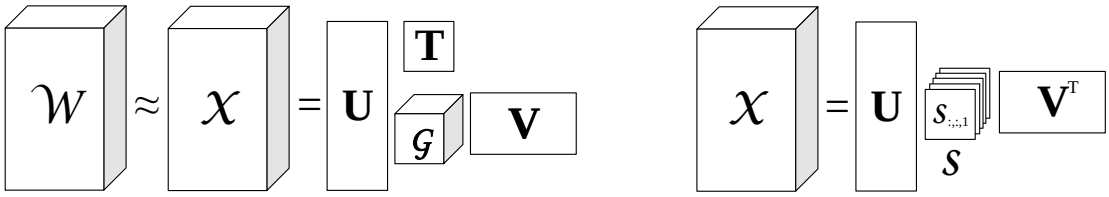
When constructing the raw tensor $\mathcal{W}$ in MRLSA, each slice is analogous to the document-term matrix in LSA, but created based on the data of a particular relation, such as synonyms. With a slight abuse of notation, we sometimes use the value rather than index when there is no confusion. For instance, $\mathcal{W}_{:,\text{“word”},k}$ represents the fiber corresponding to the “word” in slice k, and $\mathcal{W}_{:,:,\text{syn}}$ refers to the slice that encodes the synonymy relation. Below we use an example to compare this construction to the raw matrix in PILSA, and discuss how it extends LSA.

Suppose we are interested in representing two relations, synonymy and antonymy. The raw tensor in MRLSA would then consist of two slices, $\mathcal{W}_{:,:,\text{syn}}$ and $\mathcal{W}_{:,:,\text{ant}}$, to encode synonyms and antonyms of target words from a knowledge source (e.g., a thesaurus). Each row in $\mathcal{W}_{:,:,\text{syn}}$ represents the synonyms of a target word, and the corresponding row in $\mathcal{W}_{:,:,\text{ant}}$ encodes its antonyms. Figures 3(a) and 3(b) illustrate an example, where “joy”, ”gladden” are synonyms of the target word “joyfulness” and “sorrow” is its antonym. Therefore, the values of the corresponding entries are 1. Notice that the matrix $\mathbf{W}' = \mathcal{W}_{:,:,\text{syn}} - \mathcal{W}_{:,:,\text{ant}}$ is identical to the PILSA raw matrix. We can extend the construction above to enable MRLSA to utilize other semantic relations (e.g., *hypernymy*) by adding a slice corresponding to each relation of interest. Fig. 3(c) demonstrates how to add another slice $\mathcal{W}_{:,:,\text{hyper}}$ to the tensor for encoding hypernyms.

4.2 Tensor Decomposition

The MRLSA raw tensor encodes relations in one or more data resources, such as thesauri. However, the knowledge from a thesaurus is usually noisy and incomplete. In this section, we derive a low-rank approximation of the tensor to generalize the knowledge. This step is analogous to the rank-k approximation in LSA.

Various tensor decomposition methods have been proposed in literature. Among them, Tucker decomposition (Tucker, 1966) is recognized as a multi-dimensional extension of SVD and has been widely used in many applications. An illustration of this method is in Fig. 4(a). In Tucker decomposition, a $d \times n \times m$ tensor $\mathcal{W}$ is decomposed into four components $\mathcal{G}, \mathbf{U}, \mathbf{V}, \mathbf{T}$. A low-rank approximation



(a) Tucker Tensor Decomposition (b) Our Reformulation

Figure 4: Fig. 4(a) illustrates the Tucker tensor decomposition method which factors a 3-way tensor $\mathcal{W}$ to three orthogonal matrices, $\mathbf{U}, \mathbf{V}, \mathbf{T}$, and a core tensor $\mathcal{G}$. We further apply a n-mode matrix product on the core tensor $\mathcal{G}$ with $\mathbf{T}$. Consequently, each slice of the resulted core tensor $\mathcal{S}$ (a square matrix) captures a semantic relation type, and each column of $\mathbf{V}^T$ is a vector representing a word.

$\mathcal{X}$ of $\mathcal{W}$ is defined by

$$\mathcal{W}_{i,j,k} \approx \mathcal{X}_{i,j,k} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \mathcal{G}_{r_1,r_2,r_3} \mathbf{U}_{i,r_1} \mathbf{V}_{j,r_2} \mathbf{T}_{k,r_3},$$

where $\mathcal{G}$ is a core tensor with dimensions $R_1 \times R_2 \times R_3$ and $\mathbf{U}, \mathbf{V}, \mathbf{T}$ are orthogonal matrices with dimensions $d \times R_1, n \times R_2, m \times R_3$, respectively. The rank parameters $R_1 \leq d, R_2 \leq n, R_3 \leq m$ are given as input to the algorithm. In MRLSA, m (the number of relations) is usually small, while d and n are typically large (often in the scale of hundreds of thousands). Therefore, we choose $R_1 = R_2 = \tau$, $\tau \ll d, n$ and $R_3 = m$, where τ is typically less than 1000.

To make the analogy to SVD clear, we rewrite the results of Tucker decomposition by performing a n-mode matrix product over the core tensor $\mathcal{G}$ with the matrix $\mathbf{T}$. This produces a tensor $\mathcal{S}$ where each slice is a linear combination of the slices of $\mathcal{G}$ with coefficients given by $\mathbf{T}$ (see (Kolda and Bader, 2009) for detail). That is, we have

$$\mathcal{S}_{:,:,k} = \sum_{t=1}^{m} \mathbf{T}_{k,t} \mathcal{G}_{:,:,t}, \quad \forall k.$$

An illustration is shown in Fig. 4(b), Then, a straightforward calculation shows that k-th slice of tensor $\mathcal{W}$ is approximated by

$$\mathcal{W}_{:,:,k} \approx \mathcal{X}_{:,:,k} = \mathbf{U} \mathcal{S}_{:,:,k} \mathbf{V}^T. \tag{4}$$

Comparing Eq. (4) to Eq. (1), one can observe that matrices $\mathbf{U}$ and $\mathbf{V}$ play similar roles here, and each slice of the core tensor $\mathcal{S}$ is analogous to Σ. However, the square matrix $\mathcal{G}_{:,:,k}$ is not necessary to be diagonal. As in SVD, the column vectors of $\mathcal{G}_{:,:,k}\mathbf{V}^T$ (capture both word and relation information) behave similarly to the column vectors of the original tensor slice $\mathcal{W}_{:,:,k}$.

4.3 Measuring the Degrees of Word Relations

In principle, the raw information in the input tensor $\mathcal{W}$ can be used for computing lexical similarity using the cosine score between the column vectors for two words from the same slice of the tensor. To measure the degree of other relations, however, our approach requires one to specify a *pivot* slice. The key role of the pivot slice is to expand the lexical coverage of the relation of interest to additional lexical entries and, for this reason, the pivot slice should be chosen to capture the equivalence of the lexical entries. In this paper, we use the synonymy relation as our pivot slice. First we consider measuring the degree of a relation *rel* holding between the i-th and j-th words using the raw tensor $\mathcal{W}$, which can be computed as

$$\cos\left(\mathcal{W}_{:,i,\text{syn}}, \mathcal{W}_{:,j,rel}\right). \tag{5}$$

This measurement can be motivated from the logical rule: $\text{syn}(\text{word}_i, \text{target}) \wedge rel(\text{target}, \text{word}_j) \rightarrow rel(\text{word}_i, \text{word}_j)$, where the pivot relation *syn* expands the coverage of the relation of interest *rel*.

Turning to the use of the tensor decomposition, we use a similar derivation to Eq. (3), and measure the degree of relation *rel* between two words by

$$\cos\left(\mathcal{S}_{:,:,\text{syn}} \mathbf{V}^T_{i,:}, \mathcal{S}_{:,:,rel} \mathbf{V}^T_{j,:}\right). \tag{6}$$

For instance, the degree of antonymy between "joy" and "sorrow" is measured by the cosine similarity between the respective fibers $\cos(\mathcal{X}_{:,\text{"joy"},\text{syn}}, \mathcal{X}_{:,\text{"sorrow"},\text{ant}})$. We can encode both symmetric relations (e.g., *antonymy* and *synonymy*) and asymmetric relations (e.g., *hypernymy* and *hyponymy*) in the same tensor representation. For a symmetric relation, we use both $\cos(\mathcal{X}_{:,i,\text{syn}}, \mathcal{X}_{:,j,rel})$ and $\cos(\mathcal{X}_{:,j,\text{syn}}, \mathcal{X}_{:,i,rel})$ and measure the degree of a symmetric relation by the average of these two cosine similarity scores. However, for asymmetric relations, we use only $\cos(\mathcal{X}_{:,i,\text{syn}}, \mathcal{X}_{:,j,rel})$.

5 Experiments

We evaluate MRLSA on two tasks: answering the closest-opposite GRE questions and measuring degrees of various class-inclusion (i.e., *is-a*) relations. In both tasks, we design the experiments to empirically validate the following claims. When encoding two opposite relations from the same source, MRLSA performs comparably to PILSA. However, MRLSA generalizes LSA to model multiple relations, which could be obtained from both homogeneous and heterogeneous data sources. As a result, the performance of a target task can be further improved.

5.1 Experimental Setup

We construct the raw tensors to encode a particular relation in each slice based on two data sources.

Encarta The Encarta thesaurus is developed by Bloomsbury Publishing Plc[2]. For each target word, it provides a list of synonyms and antonyms. We use the same version of the thesaurus as in (Yih et al., 2012), which contains about 47k words and a vocabulary list of approximately 50k words.

WordNet We use four types of relations from WordNet: synonymy, antonymy, hypernymy and hyponymy. The number of target words and the size of the vocabulary in our version are 117,791 and 149,400, respectively. WordNet has better vocabulary coverage, but fewer antonym pairs. For instance, the WordNet antonym slice contains only 46,945 nonzero entries, while the Encarta antonym slice has 129,733.

[2]http://www.bloomsbury.com

We apply a memory-efficient Tucker decomposition algorithm (Kolda and Sun, 2008) implemented in *tensor_toolbox v2.5* (Bader et al., 2012)[3] to factor the tensor. The largest tensor considered in this paper can be decomposed in about 3 hours using less than 4GB of memory with a commodity PC.

5.2 Answering GRE Antonym Questions

The first task is to answer the closest-opposite questions from the GRE test provided by Mohammad et al. (2008)[4]. Each question in this test consists of a target word and five candidate words, where the goal is to pick the candidate word that has the most opposite meaning to the target word. In order to have a fair comparison, we use the same data split as in (Mohammad et al., 2008), with 162 questions used for the development set and 950 for test. Following (Mohammad et al., 2008; Yih et al., 2012), we report the results in precision (accuracy of the questions that the system attempts to answer), recall (percentage of the questions answered correctly over all questions) and F_1 (the harmonic mean of precision and recall).

We tune two sets of parameters using the development set: (1) the rank parameter τ in the tensor decomposition and (2) the scaling factors of different slices of the tensor. The rank parameter specifies the number of dimensions of the latent space. In the experiments, We pick the best value of τ from $\{100, 200, 300, 500, 750, 1000\}$. The scaling factors adjust the values of each slice of the tensor. The elements of each slice are multiplied by the scaling factor before factorization. This is important because Tucker decomposition minimizes the reconstruction error (the Frobenius norm of the residual tensor). As a result, the slice with a larger range of values becomes more influential to $\mathbf{U}$ and $\mathbf{V}$. In this work, we fix $\mathcal{W}_{:,:,\text{ant}}$, and search for the scaling factor of $\mathcal{W}_{:,:,\text{syn}}$ in $\{0.25, 0.5, 1, 2, 4\}$ and the factors of $\mathcal{W}_{:,:,\text{hyper}}$ and $\mathcal{W}_{:,:,\text{hypo}}$ in $\{0.0625, 0.125, 0.25\}$.

Table 1 summarizes the results of training

[3]`http://www.sandia.gov/~tgkolda/TensorToolbox`. The Tucker decomposition involves performing SVD on a large matrix. We modify the MATLAB code of *tensor_toolbox* to use the built-in `svd` function instead of `svds`. This modification reduces both the running time and memory usage.

[4]`http://www.saifmohammad.com`

	Dev. Set			Test Set		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1
WordNet Lookup	0.40	0.40	0.40	0.42	0.41	0.42
WordNet RawTensor	0.42	0.41	0.42	0.42	0.41	0.42
WordNet PILSA	0.63	0.62	0.62	0.60	0.60	0.60
WordNet MRLSA:Syn+Ant	0.63	0.62	0.62	0.59	0.58	0.59
WordNet MRLSA:4-layers	0.66	0.65	0.65	0.61	0.59	0.60
Encarta Lookup	0.65	0.61	0.63	0.61	0.56	0.59
Encarta RawTensor	0.67	0.64	0.65	0.62	0.57	0.59
Encarta PILSA	0.86	0.81	0.84	0.81	0.74	0.77
Encarta MRLSA:Syn+Ant	0.87	0.82	0.84	**0.82**	0.74	0.78
MRLSA:WordNet+Encarta	**0.88**	**0.85**	**0.87**	0.81	**0.77**	**0.79**

Table 1: GRE antonym test results of models based on Encarta and WordNet data in precision, recall and F_1. RawTensor evaluates the performance of the tensor with 2 slices encoding synonyms and antonyms before decomposition (see Eq. (5)), which is comparable to checking the original data directly (Lookup). MRLSA:Syn+Ant applies Tucker decomposition to the raw tensor and measures the degree of antonymy using Eq. (6). The result is similar to that of PILSA (see Sec. 3.1). MRLSA:4-layers adds hypernyms and hyponyms from WordNet; MRLSA:WordNet+Encarta consists of synonyms/antonyms from Encarta and hypernyms/hyponyms from WordNet, where the target words are aligned using the synonymy relations. Both models demonstrate the advantage of encoding more relations, from either the same or different resources.

MRLSA using two different corpora, Encarta and WordNet. The performance of the MRLSA raw tensor is close to that of looking up the thesaurus. This indicates the tensor representation is able to capture the word relations explicitly described in the thesaurus. After conducting tensor decomposition, MRLSA:Syn+Ant achieves similar results to PILSA. This confirms our claim that when giving the same among of information, MRLSA performs at least comparably to PILSA. However, the true power of MRLSA is its ability to incorporate other semantic relations to boost the performance of the target task. For example, when we add the hypernymy and hyponymy relations to the tensor, these class-inclusion relations provide a weak signal to help resolve antonymy. We suspect that this is due to the fact that antonyms typically share the same properties but only have the opposite meaning on one particular semantic dimension. For instance, the antonyms "sadness" and "happiness" are different forms of emotion. When two words are hyponyms of a target word, the likelihood that they are antonyms should thus be increased. We show that the target relations and these auxiliary semantic relations can be collected from the same data source (e.g., WordNet MRLSA:4-layers) or from multiple, heterogeneous sources (e.g., MRLSA:WordNET+Encarta). In both cases, the performance of the model improves as more relations are incorporated. Moreover, our experiments show that adding the hypernym and hyponym layers from WordNet improves modeling antonym relations based on the Encarta thesaurus. This suggests that the weak signal from a resource with a large vocabulary (e.g., WordNet) can help predict relations between out-of-vocabulary words and thus improve the recall.

To better understand the model, we examine the top antonyms for three question words from the GRE test. The lists below show antonyms and their MRLSA scores for each of the GRE question words as determined by the MRLSA:WordNET+Encarta model. Antonyms that can be found directly in the Encarta thesaurus are in italics.

inanimate *alive* (0.91), living (0.90), *bodily* (0.90), in-the-flesh (0.89), *incarnate* (0.89)

alleviate exacerbate (0.68), make-worse (0.67), inflame (0.66), amplify (0.65), stir-up (0.64)

relish detest (0.33), abhor (0.33), abominate (0.33), despise (0.33), loathe (0.31)

We can see that from these examples, MRLSA not

	Dev.	Test			
	1a (Taxonomic)	1b (Functional)	1c (Singular)	1d (Plural)	Avg.
WordNet Lookup	52.9	34.5	41.4	34.3	36.7
WordNet RawTensor	51.0	38.3	50.0	42.1	43.5
WordNet MRLSA:Syn+Hypony	55.8	41.7 (43.2)	51.0 (51.4)	37.5 (44.4)	43.4 (46.3)
WordNet MRLSA:4-layers	52.9	51.5 (53.9)	51.9 (60.0)	43.5 (50.5)	49.0 (54.8)
MRLSA:WordNet+Encarta	**62.1**	**55.3** (**58.7**)	**57.1** (**65.7**)	**48.6** (**53.7**)	**55.8** (**60.1**)
UTD_{NB} (Rink and Harabagiu, 2012)	-	38.3	36.7	28.2	34.4

Table 2: Results of measuring the class-inclusion (*is-a*) relations in MaxDiff accuracy (see text for detail). RawTensor has synonym and hyponym slices and measures the degree of *is-a* relation using Eq. (5). MRLSA:Syn+Hypo factors the raw tensor and judges the relation by Eq. (6). The constructions of MRLSA:4-layers and MRLSA:WordNet+Encarta are the same as in Sec. 5.2 (see the caption of Table 1 for detail). For MRLSA models, numbers shown in the parentheses are the results when parameters are tuned using the test sets. UTD_{NB} is the results of the best performing system in SemEval-2012 Task 2.

only preserves the antonyms in the thesaurus, but also discovers additional ones, such as *exacerbate* and *inflame* for "alleviate". Another interesting finding is that while the scores are useful in ranking the candidate words, they might not be comparable across different question words. This could be an issue for some applications, which need to make a binary decision on whether two words are antonyms.

5.3 Measuring degrees of *Is-A* relations

We evaluate MRLSA using the class-inclusion portion of SemEval-2012 Task 2 data (Jurgens et al., 2012). Here the goal is to measure the degree of two words having the *is-a* relation. Five annotated datasets are provided for different subcategories of this relation: 1a-taxonomic, 1b-functional, 1c-singular, 1d-plural, 1e-class individual. We omit 1e because it focuses on real world entities (e.g., queen:Elizabeth, river:Nile), which are not included in WordNet.

Each dataset contains about 100 questions based on approximately 40 word pairs. The question consists of 4 randomly chosen word pairs and asks the best and worst pairs that exemplify the specific *is-a* relation. The performance is measured by the average prediction accuracy, also called the MaxDiff accuracy (Louviere and Woodworth, 1991).

Because the questions are generated from the same set of word pairs, these questions are not mutually independent. Therefore, it is not proper to split the data of each subcategory into the development and test sets. Alternatively, we follow the setting of SemEval-2012 Task 2 and use the first subcategory (1a-taxonomy) to tune the model and evaluate its performance based on the results on other datasets. Since the models are tuned and tested on different types of subcategories, they might not be the optimal ones when evaluated on the test sets. Therefore, we show results using the best parameters tuned on the development set and those tuned on the test set, where the latter suggests a performance upper-bound. Besides the rank parameter, we tune the scaling factors of the synonym, hypernym and hyponym slices from $\{4, 16, 64\}$. The scaling factor of the antonym slice is fixed to 1.

Table 2 shows the performance in MaxDiff accuracy. Results show that even the raw tensor representation (RawTensor) performs better than WordNet lookup. We suspect that this is because the tensor representation can capture the fact that the hyponyms of a word are usually synonymous to each other. By performing Tucker decomposition on the raw Tensor, MRLSA achieves better performance. MRLSA:4-layers further leverages the information from antonyms and hypernyms and thus improves the model. As we notice in the GRE antonym test, models based on the Encarta thesaurus perform better in predicting antonyms. Therefore, it is interesting to check if combining synonyms and antonyms from Encarta helps. As a result, MRLSA:WordNet+Encarta improves over MRLSA:4-layers significantly. This demonstrates again that MRLSA can leverage knowledge stored in heterogeneous resources. Notably, MRLSA outper-

forms the best system participated in the SemEval-2012 task with a large margin, with a difference of 21.4 in MaxDiff accuracy.

Next we examine the top words that have the *is-a* relation relative to three question words from the task. The lists below show the hyponyms and their respective MRLSA scores for each of the question words as determined by MRLSA:4-layers.

bird ostrich (0.75), gamecock (0.75), nighthawk (0.75), amazon (0.74), parrot (0.74)

automobile minivan (0.48), wagon (0.48), taxi (0.46), minicab (0.45), gypsy cab (0.45)

vegetable buttercrunch (0.61), yellow turnip (0.61), romaine (0.61), chipotle (0.61), chilli (0.61)

Although the model in general does a good job finding hyponyms, we observe that some suggested words, such as *buttercrunch* (a mild lettuce) vs. "vegetable", do not seem intuitive (e.g., compared to *carrot*). Having one additional slice to capture the general term co-occurrence relation may help improve the model in this respect.

6 Conclusions

In this paper, we propose Multi-Relational Latent Semantic Analysis (MRLSA) which generalizes Latent Semantic Analysis (LSA) for lexical semantics. MRLSA models multiple word relations by leveraging a 3-way tensor, where each slice captures one particular relation. A low-rank approximation of the tensor is then derived using a tensor decomposition. Consequently, words in the vocabulary are represented by vectors in the latent semantic space, and each relation is captured by a latent square matrix. Given two words, MRLSA not only can measure their degree of having a specific relation, but also can discover unknown relations between them. These advantages have been demonstrated in our experiments. By encoding relations from both homogeneous or heterogeneous data sources, MRLSA achieves state-of-the-art performance on existing benchmark datasets for two relations, *antonymy* and *is-a*.

For future work, we plan to explore directions that aim for improving both the quality and word coverage of the model. For instance, the knowledge encoded by MRLSA can be enriched by adding more relations from a variety of linguistic resources, including the co-occurrence relations from large corpora. On model refinement, we notice that MRLSA can be viewed as a 3-layer neural network without applying the sigmoid function. Following the strategy of using Siamese neural networks to enhance PILSA (Yih et al., 2012), training MRLSA with a multi-task discriminative learning setting can be a promising approach as well.

Acknowledgments

We thank Geoff Zweig for valuable discussions and the anonymous reviewers for their comments.

References

E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca and A. Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *NAACL '09*, pages 19–27.

Brett W. Bader, Tamara G. Kolda, et al. 2012. Matlab tensor toolbox version 2.5. Available online, January.

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

Jordan L Boyd-Graber, David M Blei, and Xiaojin Zhu. 2007. A topic model for word sense disambiguation. In *EMNLP-CoNLL*, pages 1024–1033.

Shay B. Cohen, Giorgio Satta, and Michael Collins. 2013. Approximate PCFG parsing using tensor decomposition. In *NAACL-HLT 2013*, pages 487–496.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(96).

S. Dumais, T. Letsche, M. Littman, and T. Landauer. 1997. Automatic cross-language retrieval using latent semantic indexing. In *AAAI-97 Spring Symposium Series: Cross-Language Text and Speech Retrieval*.

Weiwei Guo and Mona Diab. 2012. Modeling sentences in the latent space. In *ACL 2012*, pages 864–872.

Weiwei Guo and Mona Diab. 2013. Improving lexical semantics for sentential semantics: Modeling selectional preference and similar words in a latent variable model. In *NAACL-HLT 2013*, pages 739–745.

Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence*, pages 289–296.

D. Jurgens, S. Mohammad, P. Turney, and K. Holyoak. 2012. SemEval-2012 Task 2: Measuring degrees of relational similarity. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 356–364.

Tamara G. Kolda and Brett W. Bader. 2009. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September.

Tamara G. Kolda and Jimeng Sun. 2008. Scalable tensor decompositions for multi-aspect data mining. In *ICDM 2008*, pages 363–372.

T. Landauer and D. Laham. 1998. Learning human-like knowledge by singular value decomposition: A progress report. In *NIPS 1998*.

T. Landauer. 2002. On the computational basis of learning and cognition: Arguments from lsa. *Psychology of Learning and Motivation*, 41:43–84.

Jordan J. Louviere and G. G. Woodworth. 1991. Best-worst scaling: A model for the largest difference judgments. Technical report, University of Alberta.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *NAACL-HLT 2013*.

Saif Mohammad, Bonnie Dorr, and Graeme Hirst. 2008. Computing word pair antonymy. In *Empirical Methods in Natural Language Processing (EMNLP)*.

John Platt, Kristina Toutanova, and Wen-tau Yih. 2010. Translingual document representations from discriminative projections. In *Proceedings of EMNLP*, pages 251–261.

Xipeng Qiu, Le Tian, and Xuanjing Huang. 2013. Latent semantic tensor indexing for community-based question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 434–439, Sofia, Bulgaria, August. Association for Computational Linguistics.

Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Proceedings of HLT-NAACL*, pages 109–117.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL-HLT 2013*, pages 74–84.

Bryan Rink and Sanda Harabagiu. 2012. UTD: Determining relational similarity using lexical patterns. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 413–418, Montréal, Canada, 7-8 June. Association for Computational Linguistics.

G. Salton, A. Wong, and C. S. Yang. 1975. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11).

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 *ICML '11*.

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

Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

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

Peter Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *International Conference on Computational Linguistics (COLING)*.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1142–1151, Atlanta, Georgia, June. Association for Computational Linguistics.

Wei Xu, Xin Liu, and Yihong Gong. 2003. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273, New York, NY, USA. ACM.

Wen-tau Yih and Vahed Qazvinian. 2012. Measuring word relatedness using heterogeneous vector space models. In *Proceedings of NAACL-HLT*, pages 616–620, Montréal, Canada, June.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 247–256, Portland, Oregon, USA, June. Association for Computational Linguistics.

Wen-tau Yih, Geoffrey Zweig, and John Platt. 2012. Polarity inducing latent semantic analysis. In *Proceedings of EMNLP-CoNLL*, pages 1212–1222, Jeju Island, Korea, July.

Alisa Zhila, Wen-tau Yih, Christopher Meek, Geoffrey Zweig, and Tomas Mikolov. 2013. Combining heterogeneous models for measuring relational similarity. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1000–1009, Atlanta, Georgia, June. Association for Computational Linguistics.