Contextual Text Understanding in Distributional Semantic Space *

Jianpeng Cheng [†#,1] Zhongyuan Wang [‡†,2] Ji-Rong Wen [‡,3] Jun Yan [†,4] Zheng Chen [†,5]

[†]Microsoft Research, Beijing, China [#]University of Oxford [‡]Renmin University of China, Beijing, China

[1]jianpeng.ch,[3]jirong.wen@gmail.com [2]zhy.wang,[4]junyan,[5]zhengc@microsoft.com

ABSTRACT

Representing discrete words in a continuous vector space turns out to be useful for natural language applications related to text understanding. Meanwhile, it poses extensive challenges, one of which is due to the polysemous nature of human language. A common solution (a.k.a word sense induction) is to separate each word into multiple senses and create a representation for each sense respectively. However, this approach is usually computationally expensive and prone to data sparsity, since each sense needs to be managed discriminatively. In this work, we propose a new framework for generating context-aware text representations without diving into the sense space. We model the concept space shared among senses, resulting in a framework that is efficient in both computation and storage. Specifically, the framework we propose is one that: i) projects both words and concepts into the same vector space; ii) obtains unambiguous word representations that not only preserve the uniqueness among words, but also reflect their context-appropriate meanings. We demonstrate the effectiveness of the framework in a number of tasks on text understanding, including word/phrase similarity measurements, paraphrase identification and question-answer relatedness classification.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms

Algorithm, Application

Keywords

Information retrieval; Knowledge representation; Machine learning; Search engine

*This work was done at Microsoft Research Asia.

CIKM'15 October 19 - 23, 2015, Melbourne, VIC, Australia

ACM 978-1-4503-3794-6/15/10 ...$15.00.
DOI:http://dx.doi.org/10.1145/2806416.2806517.

1. INTRODUCTION

Native speakers of a language are able to make sense of sentences they have never seen or heard before. They understand words with more than one meaning in a given context, and how these combine to form meaningful utterances. While humans perform language understanding effortlessly, the task poses enormous challenges for machines. Discovering the meaning representations for texts is thus crucial for many natural language applications, such as web document classification and search query understanding [44, 43]. A common first approach is to translate words into a machine-understandable form, such as with vectors, which has led to the creation of distributional models.

The underlying idea of representing words as vectors dates back to the distributional hypothesis that "a word is characterized by the company it keeps" [15]. Under this assumption, the meaning of a word can be obtained empirically by examining the context in which the word appears. Ultimately, words are represented as vectors and their meanings are distributed across the dimensions of a semantic space. The points in space, therefore, represent semantic concepts in such a way that similar concepts are close to each other. The distributional models constitute a powerful representation framework in which the similarity of two meanings can be easily measured.

Traditional approaches to constructing a vector space rely on the explicit collection of co-occurrence statistics. Until recently, the idea has been reformed with the development of neural language models [2, 10, 29], in which the word vectors are treated as parameters to be optimized via the training of neural networks on unstructured text data. An advantage of the training based approach is that it produces salient word encodings in a dense and low-dimensional form (a.k.a word embeddings), proven to capture the semantic relationship among words [30].

Despite their effectiveness, generic distributional models, as suggested by their mathematical nature, suffer from low precision. One reason is due to the polysemous nature of human language. As a word reveals different senses as the context changes, it should be assigned more than one representation. For example, the word *apple* may either refer to "the name of an IT *company*" or "a type of *fruit*". Merging the two senses into a single vector may lead to inaccurate representation of either sense. Further consider two queries [*python zoo*] and [*python string*]. Humans can easily identify by context that the word *python* in the former query stands for an animal species while in the latter it means a type of *programming language*. Machines, however, are not generally equipped with this ability.

Previously solutions to address language polysemy in distributional models mainly focus on the creation of sense-specific embeddings. The idea was first proposed in the work of Schütze [38],

Table 1: Words, Senses and Concepts

Word	Sense	Concept
apple	the usually round, red or yellow, edible fruit of a small tree in the rose family	fruit
	an American multinational corporation that designs, develops, and sells consumer electronics	company
orange	the fruit of the citrus species in the family Rutaceae	fruit
	the colour of saffron, carrots, pumpkins and apricots, which is between red and yellow on the spectrum of light	colour

who obtain sense representations for a target word by collecting and clustering its context vectors. In the end, a sense is simply represented by the centroid of the corresponding cluster. Reisinger and Mooney [37] formalize the cluster-based model as a *multi-prototype vector space*, which has later been used in a series of tasks on word sense disambiguation [20, 21] and neural language modeling [18, 34]. The major limitation of the above works lies in the excessive computational cost induced by the clustering step, which has to be performed repeatedly for every word in the vocabulary.

Meanwhile, another line of research directly extends neural language models with the non-parametric Bayesian approach to learn sense embeddings [35, 1, 24]. Despite improved efficiency, there is no clear mapping between the senses learned from the data and the actual senses of a word. This becomes problematic for rare words or rare senses, for which there does not exist enough training context.

Following the above analysis and extending it a bit further, traditional approaches for modeling word senses in a vector space face the following challenges:

- Linguistic diversity: Each word in a vocabulary reveals different amounts of senses that vary with context.
- Data sparsity: Purely data-driven word sense induction methods require sufficient training contexts. In the case of rare words or rare senses, the representations will be underfit.
- Efficiency: Word sense induction aims to create as output a representation for each sense. This renders high cost in both computation and storage.

On the other hand, there exist many knowledge sources which contain rich linguistic knowledge, such as the relations between words. These knowledge sources can potentially help in the learning of word representations. Against this background, an interesting question we ask in the paper is as follows:

- Can we make use of additional knowledge sources to help learn contextual word representations in a unified manner, rather than partitioning each word into isolated senses and learning sense-specific representations given only a finite number of training contexts?

The solution we provide is based on the observation that different senses can share common concepts. Consider the two words *apple* and *orange*. While each has two distinct senses (as shown in Table 1), they share the same concept of *fruit*. To a certain extent, modeling the shared concepts among senses suffices to disambiguate a word meaning in a specific context. Motivated by this, we propose a novel framework for addressing language polysemy based on the concept space shared among senses. With the word-concept relations induced from an appropriate knowledge source, we train joint word-concept embeddings from a neural language model. As a result, each word is associated with an intrinsic vector that maintains the unique features of the word. Each concept is analogously assigned a vector that delivers an unambiguous meaning. We obtain the contextual representation of a word by combining its intrinsic vector and the most context-appropriate concept vector.

The main advantage of the proposed framework are in scalability, coverage and efficiency. The concept space has a much smaller and manageable size than the sense space while it emphasizes the commonality of senses. The proposed framework is completely general. Any meaningful concept space can be integrated with our neural language model, not limited to the one obtained from an explicit knowledge base.

An alternative to our approach is to utilize the word-concept relations outside embeddings. This involves a preprocessing step that replaces each word in the training text with a word-concept pair, after which a regular embedding training is applied. While this method violates our intention to avoid the memory cost and data sparsity issue in training sense embeddings, we refer to the method as *preprocessed* and use it as a baseline.

We perform extensive experiments to validate the effectiveness and efficiency of the proposed framework. In all experiments related to text understanding, including contextual word/phrase similarity measurements, paraphrase identification and question-answer relatedness classification. we achieve impressive results comparing to the state-of-art. The contextual embedding features computed from our framework have been used as important features to shift a commercial question-answering system.

The rest of this paper is organized as follows. Section 2 introduces neural skip-gram models, to which we extend train word-concept embeddings. Section 3 provides insights on the relationship between words and concepts. Section 4 details the topology and training aspects of the extended neural language model. Section 5 describes how context-aware representations can be created. The experiments and findings are described in Section 6. We present related work in Section 7 and conclude the paper in Section 8.

2. NEURAL SKIP-GRAM MODELS

Skip-gram [29] is an efficient neural language model for learning distributional word representations from unstructured text data. The training goal of Skip-gram is to find word representations that best predicate the existence of surrounding words (i.e, contextual words). Mathematically, the objective is to maximize the following average log probability:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d}\log p(w_c|w_t) \tag{1}$$

where $w_1...w_T$ denotes the training text as a sequence of words, w_t and w_c represent the target word and the contextual word respectively, and d is the size of the context window centered at the target word. The standard way to compute the conditional probability $p(w_c|w_t)$ is to use a softmax function normalized over the entire vocabulary:

$$p(w_c|w_t) = \frac{\exp(s(\mathbf{w_c}, \mathbf{w_t}))}{\sum_{w_c\in W}\exp(s(\mathbf{w_t}, \mathbf{w_c}))} \tag{2}$$

where $\mathbf{w_t}$ and $\mathbf{w_c}$ are the respective vectors of the target word and the contextual word. W denotes the vocabulary. s is a scoring function for the pair $(\mathbf{w_t}, \mathbf{w_c})$. In a log-bilinear assumption, s is simply the dot product of the two word vectors.

Factorized Models. In the training of the above neural Skip-gram model, the time complexity induced in the softmax step is $O(|W|)$. One trick to reduce this excessive number is to factorize the vocabulary into *K* classes, such that every word belongs to exactly one class. As such, the problem of selecting a word from the whole vocabulary boils down to a two-step task: identifying the target class and selecting the word within the class. In other words, the conditional probability $p(w_c|w_t)$ is factorized as follows:

$$p(w_c|w_t) = p(s_c|w_t)p(w_c|w_t, s_c) \tag{3}$$

where s_c is the class that the word w_c belongs to. Since both terms $p(s_c|w_t)$ and $p(w_c|w_i, s_c)$ are normalized separately, the optimal time complexity of the softmax step can be reduced to $O(\sqrt{|W|})$.

In practice, a further reduction of the complexity can be achieved by repeatedly partitioning the vocabulary of a class into subclasses. The resulting tree structure is often called a *hierarchical softmax*. In general, factorized neural language models do not have any linguistic motivations behind them but are rather constructed for efficiency.

Noise Contrastive Normalization. Noise Contrastive Normalization (NCE) has previously been used as an *alternative* to factorization for speeding up the training of a neural language model. It is a technique that avoids evaluating the explicit normalization constant when computing gradients. The idea behind NCE is to convert the probability estimation problem into a binary classification task. Specifically, a binary classifier is defined to discriminate between samples drawn from the empirical distribution and those generated by a known noise distribution q, for example, a unigram distribution [33]. The objective replaces Equation (1) as:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d}(\log p(D=1|w_t,w_c)+\sum_{n=1}^{N}\log p(D=0|w_t,\bar{w}_c)) \tag{4}$$

where N is the number of negative samples drawn from the noise distributions. In the above equation, the probability that a contextual word is generated from the empirical distribution given a target word is computed as follows.

$$\log p(D=1|w_t,w_c) = \frac{p(w_c|w_t)}{p(w_c|w_t)+Nq(w_c)} \tag{5}$$

Note the equation still involves the normalization constant in each term of $p(w_c|w_t)$, but here it is treated as a parameter to be optimized during training. In the work of Mnih and Teh [33], the authors discovered that fixing the normalization constants to 1 would not significant affect the model performance.

3. RELATIONS OF WORDS AND CONCEPTS

Before we extend the neural Skip-gram to account for both words and concepts, we provide some insights on the word-concept relations and how such relations can be obtained. The term *concept* is inherited from the context of the knowledge base, where it stands for the common hypernym of a group of hyponyms. For example, both *apple* and *pear* have the concept of *fruit*. Both *Microsoft* and *Google* hold the concept of *company*. In a broader sense, the term *concept* refers to the shared meaning of a group of words. The key idea behind our model is to obtain the word-concept relations and derive the respective embeddings for each word and each concept.

Table 2: Explicit and Implicit Word-Concept Relations

	Concept	Words
explicit	fruit	apple,pear,orange,peach,banana...
	company	microsoft,google,apple,amazon,sun...
	color	red, orange, green, yellow, grey...
implicit	class1	drugs,drug,medicine,body,abuse...
	class2	mind,thought,remember,memory,thinking...
	class3	school, students, schools, education, teacher...

Here, intrinsic word embeddings are local but ambiguous: every word has a unique representation, in which all senses get merged; concept embeddings are global but specific: a concept is shared by many words but conveys an unambiguous meaning. The complementary nature of word and concept embeddings suggests that better contextual representations of words can be derived by combining the two in the right manner. Consider for instance a target word w with a set of related concepts $c_1 \cdots c_n$. The contextual word representation for w is obtained with a compositional function f as follows:

$$\mathbf{w_x} = f(\mathbf{w}, \mathbf{c_i}) \tag{6}$$

where c_i is the most appropriate concept of the word in the given context.

A first step required by our model is to identify the *candidate* concepts for each word, after which we apply an algorithm to update the relevant concept vectors together with the word vectors during the training of a neural language model. In general, the term *concept* can be defined either explicitly or implicitly (see Table 2). On the one hand, explicit concepts of a word can be acquired from a knowledge base, such as Freebase [6] and Probase [46], where the word-concept relations are explicitly mentioned. On the other hand, implicit word-concept relations can be obtained with either supervised or unsupervised machine learning algorithms. Examples include topic models [5] and Brown clustering [12]. As we mentioned, the framework we propose is broadly applicable to any meaningful concept space. Next, we show how the word and concept embeddings can be jointly trained.

4. JOINTLY EMBEDDING WORDS AND CONCEPTS

We propose two classes of neural language models for co-training word-concept embeddings, based on the Skip-gram.

4.1 Parallel Word-Concept Skip-gram

The essence of a neural language model is to discover salient word representations that best interpret the co-occurrence relations among words. This is instantiated in the Skip-gram as updating the word representations so that the model can predict the likely contexts of a target word. An assumption made here is that words that appear often in similar contexts tend to have similar meanings, and hence should be assigned similar representations. Based on this assumption, we extend the Skip-gram by introducing concepts into the prediction. In the first model variant, we emphasize the co-occurrence relations between the target concept and the contextual words. The augmented objective function is to maximize the following average log probability:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d}\log p(w_c|w_t)p(w_c|e_t) \tag{7}$$

Same as in Equation (1), $w_1, ...w_T$ denotes the entire training se-

quence. w_t and w_c represent the target word and contextual word respectively. e_t denotes the concept of w_t in the given context. The way e_t is selected from the candidate list will be discussed in a later section.

In the above model, the embeddings of a target word and its concept are updated in parallel; the two predictions $p(w_c|w_t)$ and $p(w_c|e_t)$ are decoupled but we want their values to be maximized at the same time. For the reason we call the model a variant of the Parallel Word-Concept Skip-gram (PWCS-1).

In the second variant of PWCS (PWCS-2), we use the target word to predict the concepts of the contextual words, with the following objective function:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d}\log p(w_c|w_t)p(e_c|w_t) \tag{8}$$

Combining the above two model variants, we arrive at a more complete objective function embracing all possible predicative relationships:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d}\log p(w_c|w_t)p(w_c|e_t)p(e_c|w_t)p(e_c|e_t) \tag{9}$$

The above model variant, which we name PWCS-3, depicts a complete bipartite graph between the target vertexes and the contextual vertexes, with a vertex representing a word or a concept. A pictorial interpretation of the three variants of PWCS is shown in Figure 1.

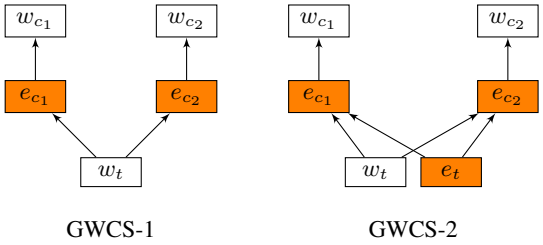
4.2 Generative Word-Concept Skip-gram

PWCS trains word-concept embeddings in a parallel fashion, where a word and its corresponding concept are assumed to be conditionally independent. A further question we pose is whether we can better emphasize the connections between a word and *its* concept within a single prediction process. As a *concept* refers to a collection of similar meanings, the task of choosing a word to fit in the context can be reduced to two steps: locating the right concept and then searching for a word underneath the chosen concept. This decomposed task can be integrated into a factorized neural language model, replacing the class s in Equation (3) with the concept e. The resulting objective function is

$$\begin{gathered}\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d} logp(w_c|w_t) \\ p(w_c|w_t) = p(e_c|w_t)p(w_c|w_t,e_c)\end{gathered} \tag{10}$$

We call the above model a variant of the Generative Word-Concept Skip-gram (GWCS-1) based on the generative nature. Note that the formulation of GWCS is in contrast to the conventional factorized neural language model, where factorization is only for efficiency concerns. Instead, we intend to extract the representation of a class as a summary of the word meanings underneath it. This in return injects linguistic motivations to the factorized neural language model. Also note that a word is allowed to be affiliated with multiple classes in GWCS.

GWCS-1 emphasizes the intrinsic relationship of a contextual word and its concept, but the concept of the target word is not included. Although it is hard to link the target word and its concept in a similar generative process, we can start from Equation (7) and extend each probabilistic term respectively:



GWCS-1 GWCS-2

Figure 2: Generative Word-Concept Skip-grams

$$\begin{gathered}\frac{1}{T}\sum_{i=1}^{T}\sum_{c=i-d,c\neq t}^{i+d} logp(w_c|w_t)p(w_c|e_t) \\ p(w_c|w_t)p(w_c|e_t) = p(e_c|w_t)p(w_c|w_t,e_c)p(e_c|e_t)p(w_c|e_t,e_c)\end{gathered} \tag{11}$$

We denote this model variant as GWCS-2. The topologies of GWCS-1 and GWCS-2 are depicted in Figure 2.

4.3 Speeding up the Training

In all variants of the extended neural Skip-gram models, computing the normalization constant for each probabilistic term can be expensive. Even for the factorized GWCSs, training can be slow if the number of concepts or the size of the class vocabulary is large. We therefore investigate methods to accelerate the training via term-wise noise contrastive estimation (NCE) [33]. Take the term $p(w_c|w_t,e_c)$ as an example, we show in the following how NCE can be applied.

With the notion of NCE described in Section 2, we define a binary classifier discriminating between samples drawn from the data distribution and those drawn from a uniform noise distribution of w_c, namely the distribution of words *within* a given class. The objective of maximizing the (log) likelihood of $p(w_c|w_t,e_c)$ is replaced as:

$$\log p(D=1|w_t,w_c,e_c) + \sum_{n=1,\bar{w}_c\sim q}^{N}\log p(D=0|w_t,\bar{w}_c,e_c) \tag{12}$$

By explicitly fixing the normalization constant to 1, we can estimate the probability as

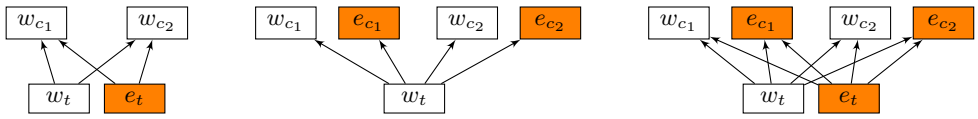
$$p(D=1|w_t,w_c,e_c) = \frac{s(w_t,w_c,e_c)}{s(w_t,w_c,e_c)+Nq(w_c)} \tag{13}$$

where s is the scoring function as in Equation (2). By applying NCE to every term in the objective function by drawing samples from the noise distribution within the domain of interest, the training can be conducted much more efficiently.

4.4 Selecting the Right Concept to Update

Another question yet to be answered is how the context-appropriate concept of a word can be identified for updating. For example, when *apple* appears in the context of "*an apple engineer in america*", we hope that the majority of the concept updates are on the concept of *company*, since the other concept (*fruit*) is irrelevant. Formally, assuming a word w in a training sentence is associated with a set of candidate concepts C, the task can be recasted to that of finding the most suitable concept $c_i \in C$ for w_t to fit the current context.

The task of linking a word to its context-appropriate concept refers to *conceptualization* in the field of knowledge representa-



PWCS-1 PWCS-2 PWCS-3

Figure 1: Parallel Word-Concept Skip-grams

tion and reasoning, which is considered an off-line task of unsupervised inference [42, 17, 45]. To save the excessive amount of time required in this preprocessing step, we propose using a softmax classifier and conceptualize each word "on the fly" of the training.

Specifically, for a word w occurring in a sentence $w_1, \cdots, w_m$ as context, we calculate the context representation as an average of the word vectors in the context:

$$\mathbf{x} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{w_i} \tag{14}$$

where w_i is the representation of the ith contextual word. We are now able to estimate the probabilistic distribution of concepts of w_i based on the softmax function acting on the context representation:

$$p(e|x) = \frac{\exp(\mathbf{e}\cdot\mathbf{x})}{\sum_{e_i\in E(w)}\exp(\mathbf{e_i}\cdot\mathbf{x})} \tag{15}$$

where $E(w)$ is the set of candidate concepts for w.

Given the likelihood of each concept in the current context, the updates can be carried out in two ways. In a hard assignment, the concept that receives the highest score will be selected for updating:

$$e = \arg\max_{e_i\in E(w)} p(e_i|x) \tag{16}$$

Under a soft criterion, a weighted update of all candidate concepts is performed. Here, we redefine the error induced by the concept as a weighted sum error induced by all candidate concepts. Take Equation (7) as an example and assume we want to update parameters based on the error $J(e)$ induced when maximizing $p(w_c|e_t)$. We redefine $J(e)$ as

$$J_o(e) = \sum_{e_i\in E(w)} p(e_i|x)J(e_i) \tag{17}$$

Based on the above equation, we perform a weighted update of all candidate concepts. The soft update has been empirically found to ensure optimum performance in a "cold start" condition.

5. CONTEXTUAL REPRESENTATIONS

In this section, we discuss how the word-concept embeddings can be used to derive context-appropriate representations for words, phrases and sentences.

5.1 Contextual Representations for Words

In the proposed model, the intrinsic word embeddings preserve the unique features of each word, but theses representations are ambiguous. On the other hand, a concept is shared by a group of words, but it conveys a precise meaning. To obtain a better representation framework that is context aware, we combine the two types of embeddings in a compositional manner. Let $\mathbf{w_x}$ denote the contextual representation for word w, whose intrinsic embedding is $\mathbf{w}$. $\mathbf{c_x}$ denotes the current concept representation of w obtained either in a hard (selecting only one concept embedding) or a soft (weighing all concept embeddings) way based on Equation (15). Then

$$\mathbf{w_x} = f(\mathbf{w}, \mathbf{c_x}) \tag{18}$$

In the above equation, f stands for an arbitrary compositional function and we use weighted sum in this work. That is,

$$\mathbf{w_x} = \mathbf{w} + \lambda\mathbf{c_i} \tag{19}$$

where λ controls the relative importance between the two types of embeddings. The optimal value of λ can be decided via grid search in a small validation set.

5.2 Contextual Representations for Larger Text Units: A Brief Discussion

Distributional representations have proven successful at the word level. But in practice, the need for representations of larger text units is more evident. For example, in web searching, we are concerned with the representations of queries and documents. While this is not the main theme of the current paper, we provide a brief discussion on how the compositional representations can be obtained so as to be injected with the contextual component carried by the word representations.

Much work in research literature has been dedicated to exploring the compositional operations of words, ranging from simple algebraic operations [31] to more sophisticated neural networks [41, 19]. Among these approaches, the simplest one is vector addition, in which the representation of a piece of text s is obtained as the sum of all word representations w_i in the text:

$$\mathbf{s}(w_1w_2\cdots w_l) = \sum_{i=1}^{l}\mathbf{w_i} \tag{20}$$

The bag-of-words approach often constitutes a simple but effective baseline.

On the other hand, neural network based compositional models are evolving to become more important in compositionality [41, 19]. In these models, a neural network takes as input a pair of word vectors $\mathbf{w_1}$, $\mathbf{w_2}$ and returns a composite vector $\mathbf{y}$ as follows:

$$\mathbf{y} = f(\mathbf{t}\cdot\mathbf{w_{1:2}} + \mathbf{b}) \tag{21}$$

Here, $\mathbf{t}$ and $\mathbf{b}$ are model parameters and f represents the nonlinearity. During experiments, we find that a simple neural bigram model perform surprisingly well. The model refers to a convolutional neural network with one convolution layer (with a window of size 2) and one pooling layer that combines the compositional bigram features. This architecture is shown in Figure 3.

6. EXPERIMENTS

Three sets of experiments are conducted to evaluate the contextual word embeddings created in our framework. This section covers all experimental details, results and findings. We start by specifying aspects of the experimental set-up.

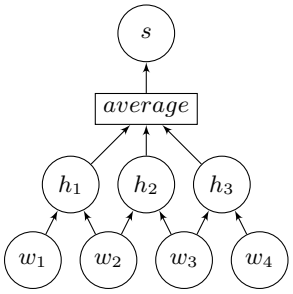


Figure 3: The architecture of a neural bigram sentence model.

6.1 Acquiring Concepts of Words

We explore two sets of word-concept relations, one obtained explicitly and the other implicitly.

Explicit Concepts. The explicit word-concept relations are retrieved from a large-scale probabilistic knowledge base, known as the Probase[1] [46]. As an advantage, Probase has a much richer concept space (with 2.7 million concepts) than other existing knowledge bases, such as Freebase (with 2,000 concepts) [6] and Cyc (with 120,000 concepts) [23]. The word-concept relationships in Probase are harvested from 1.68 billion web pages and two years of Microsoft Bing search logs [46, 22].

Overall, Probase contains a number of closely related concepts. For example, *apple* is associated with both concepts of *fruit* and *seasonal fruit*. To reduce the amount of overlapped concepts, we perform k-Medoids clustering [25] on the raw concepts of Probase. Eventually we get 4,819 concept clusters, with each centered at the most salient concept. We use the 4,819 concept clusters as the explicit concept space.

In Probase, every word-concept relationship has a typicality score indicating the importance of the concept to the word. After the K-Medoids clustering, we compute a similar score for each word-cluster by aggregating the scores between the word and every concept within the cluster. We then prune for each word the clusters with low scores. As a result, a polysemous word normally belongs to 3 to 10 clusters.

Implicit Concepts. The explicit word-concept relations usually have high accuracy but low coverage since many verbs and adjectives do not possess concepts in the knowledge base. We additionally explore implicit concepts obtained in a data-driven manner. Specifically, we perform Latent Dirichlet Allocation (LDA) [5], a well established topic model, with 3 million Wikipedia documents.

LDA is a hierarchical Bayesian model for text generation. In LDA, each document d is modeled as a distribution over K topics, each of which is then characterized by a distribution over words. The words in a document are generated by repeatedly sampling a topic based on the topic distribution and then sampling a single word from the selected topic.

Formally, given a corpus consisting of M documents, the generative process for a document d is defined as follows. First, the mixing proportion over topics θ_d is drawn from a Dirichlet prior with parameters α. Each document is then generated according to the topic distributions $z_{1:K}$ and word probabilities over topics β. The probability of a document d in a corpus is defined as:

$$p(d|\alpha,\beta) = \int_\theta p(\theta_d|\alpha)(\prod_{n=1}^{N}\sum_{z_k} p(z_k|\theta_d)p(w_n|z_k,\beta))d\theta \quad (22)$$

[1]Probase data is publicly available at http://probase.msra.cn/dataset.aspx

The central challenge in LDA is to compute the posterior distribution $p(\theta, z|d, \alpha, \beta)$ of the hidden variable z given a document d. Although this distribution is generally intractable, a variety of approximate inference algorithms have been proposed in the literature. In this work, we use a well-implemented variational inference for posterior estimation [36]. The number of topics is set to 500. As a result, we obtain 500 latent topics as implicit concepts. We prune for each word the topics that receive low probabilities.

6.2 Training

We train our neural language models on the "One Billion Word Language Modeling Benchmark" dataset [9] released by Google. All model variants are implemented in two respective concept spaces and the resulting embeddings are evaluated. In each model variant, a word is associated with two sets of vectors, assuming the same word as target and as context come from two distinct vocabularies [28]. For concepts we do not adopt this assumption since they are essentially shared among words. The dimension of all embeddings is set to 300. The code is written in C language as an extension of the efficient WORD2VEC toolkit [29]. All trainings are performed on a Windows Server 2008 with a 2.13 GHz Intel Xeon CPU (4 processors).

6.3 Similarities of Words and Phrases

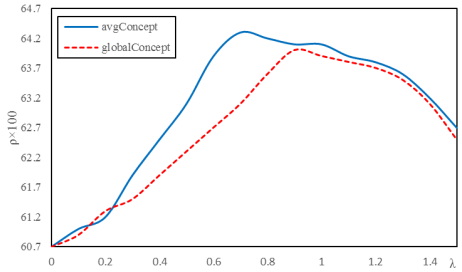
The first set of experiments concerns the similarity of words and phrases as a generic evaluation of the vector space. We provide comparisons between the aforementioned model variants, as well as comparisons between our models and a few well-established baselines. To avoid overfitting, we adopt two datasets here, one at the word level and the other for phrases.

The word-level dataset refers to the Stanford Contextual Word Similarity Dataset (SCWS) of Huang et al. [18], which contains 2,003 pairs of words and their sentence contexts. For each pair of words, there are 10 human similarity scores provided. Our task is to evaluate by Spearman's correlation how the similarities computed from various models match those of human judgments. Note that the sentence contexts provided in the dataset enable us to create contextual word representations.

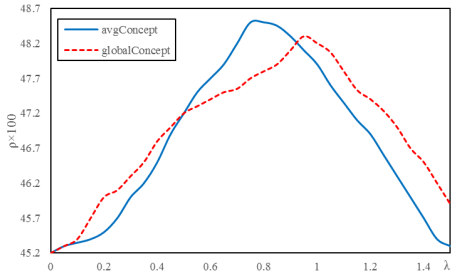
The phrase similarity dataset we use is the one of Mitchell and Lapata (M& L) [32]. This dataset consists of similarity judgments for 1,944 pairs of adjective-noun (A-N), noun-noun (N-N) and verb-object (V-O) phrases, respectively. (In total, there are 5,832 phrase pairs.) Similar to SCWS, each pair of phrases in M&L is annotated with a similarity score. Although there is no additional context provided for each phrase, the contextual representation of a word can be derived with its neighboring word in the phrase as context.

Hyper-parameters. At first, we consider the two forms of concept representations of a word in a given context. The *globalConcept* metric selects a single concept for the word based on the score computed from Equation (15). The *avgConcept* metric, on the other hand, derives the contextual concept representation as a sum of all concept embeddings weighed by their scores. We compare the two types of concept representations obtained by PWCS-1 with the explicit concept space. Meanwhile, we perform a grid search to decide the optimum value of λ in Equation (19).

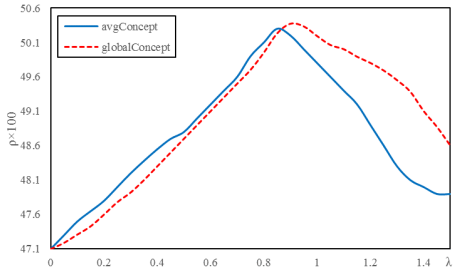
From Figure 4, we observe that *avgConcept* generally outperforms *globalConcept*. In both metrics, the Spearman's correlation increases as the conceptual features start to be injected into the word embeddings. However, a further increase of λ will lead to some performance degradation in either case. One reason is that words need to maintain their unique features in order to discriminate one another; whereas the representations of two conceptually same words will be almost the same in the case of a large λ.



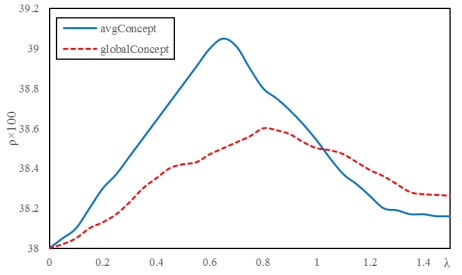
(a) SCWS



(b) A-N



(c) N-N



(d) V-O

Figure 4: Plot for Spearman's Correlation against λ.

For *avgConcept*, the best result is obtained when λ is between 0.5 to 0.8, while for *globalConcept* the optimum λ is between 0.8 to 1. The rest of the experiments will be based on *avgConcept* with lambda set to 0.75.

Comparison of Model Variants. In the next part, we evaluate the effectiveness of conceptual word representations against raw word embeddings. The comparison is respectively performed in each of the five model variants with the explicit concept space. The results are summarized in Table 3. As we can see, for all variants of PWCS and GWCS, representing words contextually brings significant performance gains. We further notice that GWCSs in general outperform PWCSs in terms of both types of embeddings. The finding indicates that taking into account the semantic relations within a neural architecture is helpful for the representation learning task.

		PWCS-1	PWCS-2	PWCS-3	GWCS-1	GWCS-2
SCWS	Raw	60.7	60.3	60.7	61.9	61.6
	Contextual	**63.5**	**63.4**	**64.1**	**65.7**	**64.5**
A-N	Raw	45.2	44.7	45.0	45.8	45.5
	Contextual	**48.5**	**48.1**	**49.3**	**50.4**	**49.2**
N-N	Raw	47.1	46.3	47.0	47.6	47.3
	Contextual	**50.3**	**50.5**	**50.6**	**53.1**	**51.3**
V-O	Raw	38.0	37.9	37.8	38.2	38.5
	Contextual	**38.9**	**38.2**	**38.9**	**39.7**	**38.6**

Table 3: Spearman's correlation $\rho \times 100$ of the variants of P-WCS and GWCS.

Comparison of Concept Spaces. Next, we compare the contextual word representations trained with the two concept spaces: explicit concepts based on the Probase and implicit concepts obtained via LDA. We use PWCS-3 and GWCS-1 in this evaluation. The results are shown in Table 4.

Comparing the numbers given by the two spaces, the explicit space shows superior performance on the similarity measure of noun-noun phrases, while the implicit space does better on the SCWS and verb-object tasks. One interpretation is that explicit concept mapping has higher accuracy but low "recalls": many words (such as verbs and adjectives) do not have any concept in the knowledge base. In this case, inducing implicit concepts provides a complement.

	Space	SCWS	A-N	N-N	V-O
PWCS-3	Explicit	64.1	49.3	**50.6**	38.9
	Implicit	**64.3**	**49.9**	49.2	**39.0**
GWCS-1	Explicit	65.7	**50.4**	**53.1**	39.7
	Implicit	**65.9**	50.0	52.5	**39.8**

Table 4: Spearman's correlation $\rho \times 100$ for contextual word representations in different concept spaces.

Comparison with Baselines and Published Results. Next, we compare the results of our models with a few well-established baselines in the SCWS task. These include the neural language model of Collobert and Weston (C&W) [10], the ordinary Skip-gram model trained with hierarchical softmax, and three sets of sense embedding models of Huang et al. [18] (*nClass*=3), Neelakantan et al. [34] and Bartunov et al. [1], respectively. These models were all trained with the code released by the authors. An additional baseline is *preprocessed*, in which the word-concept relations are used outside embeddings. In this method, the training words are preprocessed into *word-concept* pairs, after which the ordinary Skip-gram is applied. In terms of our models, we report the *best* number achieved with PWCS and GWCS respectively.

Overall, we see from Table 5 that the contextual word representations significantly outperform the word embedding baselines. The numbers given by sense embeddings are quite competitive without any consideration of efficiency in mind, but still the highest number is achieved by GWCS. Table 6 further compares the training time of various models *relative to* the Skip-gram. In terms of efficiency, our models have a clear advantage over Huang et al. [18], while they are slightly faster than the non-parametric methods of Neelakantan et al. [34] and Bartunov et al.[1], But note that we only need to store 4,819 explicit concept vectors or 500 implicit concept vectors, whereas the size of sense vectors is usually several times more than the size of the vocabulary (55,000).

	Model	$\rho \times 100$
Word Emb.	C&W	58.6
	Skip-gram	58.3
Sense Emb.	Huang et al. (2012)	62.9
	Neelakantan et al. (2014)	64.7
	Bartunov et al. (2015)	65.5
	preprocessed	63.5
Contextual Emb.	PWCS	64.3
	GWCS	**65.9**

Table 5: Spearman's correlation $\rho \times 100$ of various models in comparison.

Model	Time
Skip-gram	1
Huang et al. (2012)	117
Neelakantan et al. (2014)	3.9
Bartunov et al. (2015)	3.5
preprocessed	5
PWCS	3.4
GWCS	2.2

Table 6: Training time of various models relative to the Skip-gram.

Concept Assignments for Rare Words. In the end, we qualitatively evaluate the selected concepts by GWCS-1 for a few rare words in SCWS. In the concept-based disambiguation method, we can visualize directly the *exact* concept that a word is assigned to. However, in the sense-based models, we are unable to find out the exact meaning behind a sense.

Word	Context	Concept
Cambridge	Whittington , with the help of research students Derek Briggs and Simon Conway Morris of the University of **Cambridge** , began a thorough reassessment of the Burgess Shale , and revealed that the fauna represented were much more diverse and unusual than Walcott had recognized.	school:0.62 institution:0.27 spot:0.07 town:0.02 country:0.01 guide:0.01
placental	As a result, native Australian **placental** mammals, such as hopping mice, are more recent immigrants.	pregnancy outcome: 0.53 condition: 0.38 organ: 0.09
assimilation	Ethnic cohesiveness is a resistance strategy to **assimilation** and the accompanying cultural dissolution.	adaptation: 0.49 physiological process: 0.18 impact: 0.17 disturbance: 0.13 lifestyle change: 0.02 symptom:0.01
bootleg	Often , **bootleg** DVD copies of movies are available before the prints are officially released in cinemas.	motif: 0.77 work: 0.12 rarity: 0.07 brewery: 0.04

Table 7: Concept assignments for rare words in SCWS.

As the results in Table 7 suggest, rare words can be accurately mapped to the most context-appropriate concepts. We argue that the concept-based disambiguation method is superior in the sense that a concept representation is learnt with all words sharing that concept. This not only emphasizes the semantic interactions among words, but also provides an elegant solution to rare words whose senses are hard to induce the other way around.

6.4 Paraphrase Detection

Measuring the similarity of two words or phrases provides preliminary evidence on the quality of the contextual word representations. In the second experiment, we augment the evaluation by measuring the similarity of sentences. We use the Microsoft Research Paraphrase Corpus (MSRPC) introduced by Dolan et al. [11]. The MSRPC contains 5,801 sentence pairs (4,076 for training and 1,725 for test) and labels (0 or 1) indicating whether a pair of sentences has a paraphrase relationship. We explore the use of contextual features obtained with our models in the classification task and compare the results with various baselines and published numbers.

	Acc.	F1
Baselines		
Skip-gram + $\oplus$	71.9	81.3
Our Models		
PWCS + $\oplus$	74.5	82.7
PWCS + nn	75.0	**83.0**
GWCS + $\oplus$	74.3	82.5
GWCS + nn	**75.1**	82.9
Published Models		
LSA (Hassan,2011)	68.8	79.9
MSC (Mihalcea et al., 2006)	70.3	81.3
SSA (Hassan,2011)	72.5	81.4
SDS (Blacoe and Lapata, 2012)	73.0	82.3
RAE+DP (Socher et al., 2011)	76.8	83.6
MTM (Madnani et al., 2012)	**77.4**	**84.1**

Table 8: Results of the paraphrase identification.

Specifically, we use three sets of features here: 1) the cosine similarity of the pair of sentence vectors; 2) the number of shared concepts of the two sentences identified from our models; 3) the lengths and the co-occurring word counts of the two sentences. In the baselines, the second feature is not included. Note that the third set of features do not stem from our models but are used as a complement, since raw embeddings cannot capture subtle aspects of a sentence such as proper nouns. Similar features are also used in the studies of Blacoe and Lapata [4] and Socher et al. [40]. We follow Blacoe and Lapata [4] to use the liblinear classifier [13] in this task. Besides, we explore two ways to compose words into sentences: by vector addition ($\oplus$) and neural bigram model (nn) respectively.

As the results in Table 8 reveal, our models outperform most baselines and the published numbers which are based on distributional features [16, 4, 27]. Among the published models, the closest one to ours is the work of Blacoe and Lapata [4], who use a similar set of features except that their embeddings are based on co-occurrence statistics. Results demonstrate the superiority of our contextual embedding space. Still, there exist more sophisticated models that are successful in performing this task. For example, Socher et al. [40] apply dynamic pooling on the output of the recursive auto-encoder to create more faithful similarity features; and Madnani et al. [26] use a combination of eight machine translation metrics to achieve state-of-art results. Compared with these works, our features are much shallower yet we obtain close numbers.

6.5 Question-Answer Relatedness Classification

In the last experiment, we evaluate the effectiveness contextual word representations in a practical task. As an aid to open domain question-answering, the task is to decide whether a pair of question and answer are semantically related. The corresponding dataset contains 85,512 pairs of question-type queries and the corresponding answers retrieved from the search log of a commercial search engine. The semantic relatedness of each question and answer is scored by a group of professional annotators, after which a label (0 or 1) is assigned to each pair based on the average score. Our task here is to evaluate the quality of the embedding features by evaluating how the semantic relatedness computed in the vector space matches that of human judgment.We divide the dataset into

80% for training and 20% for testing. We train a logistic regression classifier that takes as input the cosine similarity of each question-answer and outputs the label. The representation of a question or an answer is obtained with vector addition excluding stop-words. The classification results of various models are shown in Table 9.

	Acc.
Baselines	
All positive	69.1
Skip-gram	76.8
Our models	
PWCS, Raw	77.8
PWCS, Contextual	**82.5**
GWCS, Raw	78.2
GWCS, Contextual	**83.1**

Table 9: Results of the Q-A relatedness classification.

We see a significant increase in the performance of contextual word representations with respect to raw word embeddings and baselines. In the presence of a large amount of training data, the experimental results provide stronger evidence on the effectiveness of our contextual word representations.

7. RELATED WORK

Word representations learnt from traditional distributional practices and neural language models are limited by their inability to handle polysemy. To solve the problem, several approaches have been proposed to create multi-sense word representations. The initial proposal was to obtain the sense vectors of a word by clustering the context vectors obtained in a normal distributional practice. This methodology has later been popularized by Reisinger and Mooney [37] and named as Multi-Prototype Vector Space (MPVS). Huang et al. [18] apply MPVS to pre-trained word embeddings to get neural sense representations. More recently, Neelakantan et al. [34] integrate the context clustering step into a Skip-gram, resulting in a unified model where all parameters are jointly optimized during training.

A common limitation of all the above work lies in the excessive amount of computational cost induced in the clustering step, which has to be applied to every word in the vocabulary.

There also exist several approaches to training sense embeddings directly from a neural language model, without considering any clustering step in mind. Pina and Johansson [35] use a naive Bayesian step for updating the most proper senses in a neural language model. Bartunov et al. [1] propose a non-parametric Bayesian extension of the Skip-gram. Their model is capable of learning not only sense embeddings but also the number of senses a word has. Meanwhile, Li and Jurafsky [24] use the non-parametric Chinese Restaurant Process for sense induction in a Skip-gram. Despite the improved efficiency of these models, they all require a significant amount of storage in the case of a large vocabulary, since every sense is associated with a vector which will be stored. Moreover, there is no clear mapping between the senses learned by these models and the actual senses of a word. This becomes problematic for rare words or rare senses, for which there does not exist enough training contexts for inference.

Another line of related research is on knowledge-powered word embeddings. Following the work of Bordes et al. [8], several methods have been proposed to represent entities of a knowledge base in the vectorial form [8, 7, 39]. The majority deal with the $< head\,entity, relation, tail\,entity >$ triplets and scores their relational plausibility with word embeddings. The scoring function ranges from simple spatial distance between entity embeddings transformed through the relation [8, 7] to more sophisticated Neural Tensor Networks [39]. Additionally, some works use relational information extracted from semantic lexicons as constrains in the training of neural language models [3], or in a post-processing step to refine pre-trained word vectors [14]. To the best of our knowledge, no previous work on embeddings deals with word-concept relations in a knowledge base.

8. CONCLUSION

Representing text well is crucial for machines to understand human language. In this paper, we propose a novel framework for representing text contextually in the distributional semantic space. The key idea is to utilize the conceptual information of words to disambiguate their meanings. In the proposed framework, words are assigned with dynamic representations which change with contexts. This framework differs from traditional "vector-space based word sense induction" in the sense that it does not create any sense-specific representations. Instead, it looks into the shared concepts among senses, resulting in a class of neural language models efficient in both computation and storage, and with improved applicability to rare words and rare senses. An interesting future direction would be incorporating the concept induction step within the training of a neural language model referencing Brown clustering [12].

9. ACKNOWLEDGMENTS

This work was partially supported by the National Key Basic Research Program (973 Program) of China under grant No. 2014CB340403 and the Fundamental Research Funds for the Central Universities & the Research Funds of Renmin University of China. The authors would like to acknowledge Haibo Shi for the initial discussions on this topic at Microsoft Research Asia.

10. REFERENCES

[1] S. Bartunov, D. Kondrashkin, A. Osokin, and D. Vetrov. Breaking sticks and ambiguities with adaptive skip-gram. *arXiv preprint arXiv:1502.07257*, 2015.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[3] J. Bian, B. Gao, and T.-Y. Liu. Knowledge-powered deep learning for word embedding. In *Machine Learning and Knowledge Discovery in Databases*, pages 132–148. Springer, 2014.

[4] W. Blacoe and M. Lapata. A comparison of vector-based representations for semantic composition. In *EMNLP 2012*, pages 546–556. Association for Computational Linguistics, 2012.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD 2008*, pages 1247–1250. ACM, 2008.

[7] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.

[8] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *Conference on Artificial Intelligence*, number EPFL-CONF-192344, 2011.

[9] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

[10] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[11] B. Dolan, C. Quirk, and C. Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, page 350. Association for Computational Linguistics, 2004.

[12] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

[13] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[14] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*, 2014.

[15] Z. S. Harris. Distributional structure. *Word*, 1954.

[16] S. Hassan. *Measuring semantic relatedness using salient encyclopedic concepts*. University of North Texas, 2011.

[17] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou. Short text understanding through lexical-semantic analysis. In *International Conference on Data Engineering (ICDE)*, 2015.

[18] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th ACL*. Association for Computational Linguistics, 2012.

[19] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[20] D. Kartsaklis, M. Sadrzadeh, et al. Prior disambiguation of word tensors for constructing sentence vectors. In *EMNLP*, pages 1590–1601, 2013.

[21] D. Kartsaklis, M. Sadrzadeh, and S. Pulman. Separating disambiguation from composition in distributional semantics. In *Proceedings of CoNLL*, pages 114–123, 2013.

[22] T. Lee, Z. Wang, H. Wang, and S.-w. Hwang. Attribute extraction and scoring: A probabilistic approach. 2013.

[23] D. B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[24] J. Li and D. Jurafsky. Do multi-sense embeddings improve natural language understanding? *arXiv preprint arXiv:1506.01070*, 2015.

[25] P. Li, H. Wang, K. Q. Zhu, Z. Wang, and X. Wu. Computing term similarity by large probabilistic isa knowledge. In *CIKM*, 2013.

[26] N. Madnani, J. Tetreault, and M. Chodorow. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of NAACL*, pages 182–190. Association for Computational Linguistics, 2012.

[27] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780, 2006.

[28] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[29] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[30] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. 2013.

[31] J. Mitchell and M. Lapata. Vector-based models of semantic composition. In *ACL*, pages 236–244, 2008.

[32] J. Mitchell and M. Lapata. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[33] A. Mnih and Y. W. Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.

[34] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum. Efficient nonparametric estimation of multiple embeddings per word in vector space. In *Proceedings of EMNLP*, 2014.

[35] L. N. Pina and R. Johansson. A simple and efficient method to generate word sense representations. *arXiv preprint arXiv:1412.6045*, 2014.

[36] D. Ramage, E. Rosen, J. Chuang, C. D. Manning, and D. A. McFarland. Topic modeling for the social sciences. In *Workshop on Applications for Topic Models, NIPS*, 2009.

[37] J. Reisinger and R. J. Mooney. Multi-prototype vector-space models of word meaning. In *NAACL 2010*, pages 109–117. Association for Computational Linguistics, 2010.

[38] H. Schütze. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123, 1998.

[39] R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934, 2013.

[40] R. Socher, E. H. Huang, J. Pennin, C. D. Manning, and A. Y. Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809, 2011.

[41] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of (ICML-11)*, pages 129–136, 2011.

[42] Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2330–2336. AAAI Press, 2011.

[43] F. Wang, Z. Wang, Z. Li, and J.-R. Wen. Concept-based short text classification and ranking. In *ACM International Conference on Information and Knowledge Management (CIKM)*, October 2014.

[44] Z. Wang, H. Wang, and Z. Hu. Head, modifier, and constraint detection in short texts. In *IEEE 30th International Conference on Data Engineering (ICDE)*. IEEE, 2014.

[45] Z. Wang, K. Zhao, H. Wang, X. Meng, and J.-R. Wen. Query understanding through knowledge-based conceptualization. In *IJCAI*, 2015.

[46] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD 2012*, pages 481–492. ACM, 2012.