Accepted in Information Retrieval Journal.
Publisher link: http://link.springer.com/article/10.1007/s10791-018-9340-3

Beyond Word Embeddings: Learning Entity and Concept Representations from Large Scale Knowledge Bases

Walid Shalaby · Wlodek Zadrozny · Hongxia Jin.

Received: 30 January 2018 / Accepted: 2 August 2018

Abstract Text representations using neural word embeddings have proven effective in many NLP applications. Recent researches adapt the traditional word embedding models to learn vectors of multiword expressions (concepts/entities). However, these methods are limited to textual knowledge bases (e.g., Wikipedia). In this paper, we propose a novel and simple technique for integrating the knowledge about concepts from two large scale knowledge bases of different structure (Wikipedia, and Probase) in order to learn concept representations. We adapt the efficient skip-gram model to seamlessly learn from the knowledge in Wikipedia text and Probase concept graph. We evaluate our concept embedding models on two tasks: 1) analogical reasoning, where we achieve a state-of-the-art performance of 91% on semantic analogies, 2) concept categorization, where we achieve a state-of-the-art performance on two benchmark datasets achieving categorization accuracy of 100% on one and 98% on the other. Additionally, we present a case study to evaluate our model on unsupervised argument type identification for neural semantic parsing. We demonstrate the competitive accuracy of our unsupervised method and its ability to better generalize to out of vocabulary entity mentions compared to the tedious and error prone methods which depend on gazetteers and regular expressions.

In this paper, we use the terms "concept" and "entity" interchangeably.

Walid Shalaby
Department of Computer Science
University of North Carolina at Charlotte
9201 University City Blvd, Charlotte, NC 28223, USA
E-mail: wshalaby@uncc.edu

Wlodek Zadrozny
Department of Computer Science
University of North Carolina at Charlotte
9201 University City Blvd, Charlotte, NC 28223, USA
E-mail: wzadrozn@uncc.edu

Hongxia Jin
Samsung Research America
665 Clyde Avenue, Mountain View, CA 94043, USA
E-mail: hongxia.jin@samsung.com

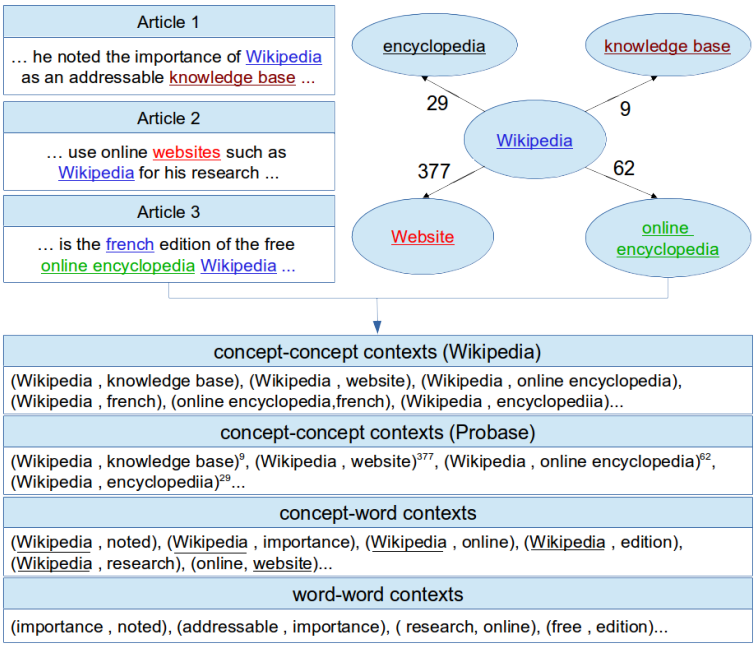

Fig. 1 Integrating knowledge from Wikipedia text (left) and Probase concept graph (right). Local concept-concept, concept-word, and word-word contexts are generated from both KBs and used for training the skip-gram model.

Keywords Entity & Concept Embeddings · Entity Identification · Concept Categorization · Skip-gram · Probase · Knowledge Graph Representations

1 Introduction

Vector-based semantic representation models are used to represent textual structures (words, phrases and documents) as multidimensional vectors. Typically, these models utilize textual corpora and/or Knowledge Bases (KBs) in order to extract and model real-world knowledge. Once acquired, any given text structure is represented as a real-valued vector in the semantic space. The goal is thus to accurately place semantically similar structures close to each other in that semantic space, while placing dissimilar structures far apart.

Recent neural-based methods for learning word vectors (embeddings) have even succeeded in capturing both syntactic and semantic regularities using simple vector arithmetic (Mikolov et al (2013a,b); Pennington et al (2014)). For example, inferring analogical relationships between words: *vec(king)-vec(man)+vec(woman)=vec(queen)*. This indicates that the learned vector dimensions encode meaningful multi-clustering for each word.

Word vectors suffer significant limitations. First, each word is assumed to have a single meaning regardless of its context and thus is represented by a single vector in the semantic space (e.g., *charlotte (city)* vs. *charlotte (given name)*). Second, the space contains vectors of single words only. Vectors of multiword expressions (MWEs) are typically obtained by averaging the vectors of individual words. However, this would often produce inaccurate representations especially if the

meaning of the MWE is different from the composition of meanings of its individual words (e.g., *vec(north carolina)* vs. *vec(north)+vec(carolina)*. Additionally, mentions that are used to refer to the same concept would have different embeddings (e.g., *u.s., america, usa*), and the model might not be able to place those individual vectors in the same sub-cluster, especially the rare surface forms.

To address these limitations, a lot of research interest has been focusing on learning distributed representations of concepts and entities which are lexical expressions (single or multiword) that denote an idea, event, or an object and have a set of properties. Typically each concept has an entry in a KB (e.g., an article in Wikipedia or a node in knowledge graph). Such entity embeddings models utilize text KBs (e.g., Wikipedia) or a triple-based KBs (e.g., DBpedia and Freebase) in order to learn entity vectors. Broadly speaking, existing methods can be divided into two categories. First, methods that learn embeddings of KB concepts only (Hu et al (2015); Zwicklbauer et al (2016); Li et al (2016); Ristoski and Paulheim (2016)). Second, methods that jointly learn embeddings of words and concepts in the same semantic space (Wang et al (2014); Fang et al (2016); Yamada et al (2016); Camacho-Collados et al (2016); Fang et al (2016); Cao et al (2017); Shalaby and Zadrozny (2017); Phan et al (2017)).

In this paper, we introduce an effective approach for jointly learning word and concept vectors from two large scale KBs of different modalities: a text KB (Wikipedia) and a graph-based concept KB (Microsoft concept graph[1] (aka Probase)). We adapt skip-gram, the popular local context window method Mikolov et al (2013b), to integrate the knowledge from both KBs. As shown in Figure 1, three key properties differentiate our approach from existing methods. First, we generate word and concept contexts from their raw mentions in the Wikipedia text. This makes our model extensible to other text corpora with annotated concept mentions. Second, we model Probase as a weighted undirected KB graph, exploiting the co-occurrence counts between pairs of concepts. This allows us to generate more concept-concept contexts during training, and subsequently learn better concept vectors for rare and infrequent concepts in Wikipedia. Third, to our knowledge, this work is the first to combine knowledge from two KBs of different modalities (Wikipedia and Probase) into a unified representation.

We evaluate the generated concept vectors intrinsically on two tasks: 1) analogical reasoning where we achieve a state-of-the-art accuracy of 91% on semantic analogies, 2) concept categorization on two datasets, where we achieve 100% accuracy on one dataset and 98% accuracy on the other. We also present a case study to analyze the impact of using our concept vectors for unsupervised argument type identification with semantic parsing as an end-to-end task. The results show competitive performance of our unsupervised method compared to the tedious and error prone argument type identification methods which depend on gazetteers and regular expressions. The analysis also shows superior generalization performance on utterances containing out of vocabulary (OOV) mentions.

We make our concept vectors and source code publicly available[2] for the research community for further experimentation and replication.

[1] https://concept.research.microsoft.com

[2] https://sites.google.com/site/conceptembeddings/

2 Learning Concept Embeddings

2.1 Skip-gram

We learn continuous vectors of words and entities by building upon the skip-gram model of Mikolov et al (2013b). In the conventional skip-gram model, a set of contexts are generated by sliding a context window of predefined size over sentences of a given text corpus. The vector representation of a target word is learned with the objective to maximize the ability of predicting surrounding words of that target word.

Formally, given a training corpus of V words $w_1, w_2, ..., w_V$. The skip-gram model aims to maximize the average log likelihood probability:

$$\frac{1}{V}\sum_{i=1}^{V}\sum_{-s\leq j\leq s, j\neq 0} \log\ p(w_{i+j}|w_i) \tag{1}$$

where s is the context window size, w_i is the target word, and w_{i+j} is a surrounding context word. The softmax function is used to estimate the probability $p(w_O|w_I)$ as follows:

$$p(w_O|w_I) = \frac{\exp(\mathbf{v}_{w_O}^{\intercal}\mathbf{u}_{w_I})}{\sum_{w=1}^{V}\exp(\mathbf{v}_{w}^{\intercal}\mathbf{u}_{w_I})} \tag{2}$$

where $\mathbf{u}_w$ and $\mathbf{v}_w$ are the input and output vectors respectively, and V is the vocabulary size. Mikolov et al (2013b) proposed hierarchical softmax and negative sampling as efficient alternatives to approximate the softmax function (which becomes computationally intractable when V becomes huge).

2.2 Learning from Text

Our approach genuinely learns distributed concept representations by generating concept contexts from mentions of those concepts in large encyclopedic text KBs such as Wikipedia. Utilizing such annotated KBs eliminates the need to manually annotate concept mentions and thus comes at no cost.

Here we propose learning the embeddings of both words and concepts jointly. First, all concept mentions are identified in the given corpus. Second, contexts are generated for both words and concepts from other surrounding words and other surrounding concepts as well. After generating all the contexts, we use the skip-gram model to jointly learn embeddings of words and concepts. Formally, given a training corpus of V words $w_1, w_2, ..., w_V$. We iterate over the corpus identifying words and concept mentions and thus generating a sequence of T tokens $t_1, t_2, ...t_T$ where $T < V$ (as multiword concepts will be counted as one token). Afterwards we train the a skip-gram model aiming to maximize:

$$\mathscr{L}_t = \frac{1}{T}\sum_{i=1}^{T}\sum_{-s\leq j\leq s, j\neq 0} \log\ p(t_{i+j}|t_i) \tag{3}$$

where as in the conventional skip-gram model, s is the context window size. Here, t_i is the target token which would be either a word or a concept mention, and t_{i+j} is a surrounding context word or concept mention.

2.3 Learning from Concept Graph

We employ Microsoft concept graph (Probase), a large scale probabilistic KB of millions of concepts and their relationships (basically is-a hierarchy). Probase was created by mining billions of Web pages and search logs of Microsoft's Bing[3] repository using syntactic patterns. The concept KB was then leveraged for text conceptualization to support text understanding tasks such as clustering of Twitter messages and News titles (Song et al (2011, 2015)), search query understanding (Wang et al (2015b)), short text segmentation (Hua et al (2015)), and term similarity (Kim et al (2013)).

Probase has a different structure (or modality) than Wikipedia because the knowledge is organized as a graph whose nodes are concepts and edges represent a weighted is-a relationship between pairs of concepts. Formally, we model Probase as a 4-tuple graph $G = (C, E, \mathscr{T}_C, \mathscr{T}_E)$ such that:

- C is a set of vertices representing concepts.
- E is a set of edges (arcs) connecting pairs of concepts.
- $\mathscr{T}_C$ is a finite set of tuples representing global statistics of each concept (i.e. its total occurrences).
- $\mathscr{T}_E$ is a finite set of tuples representing co-statistics of each edge connecting pairs of concepts (i.e. their co-occurrence count).

Under this representation, location information is lost. Therefore the context of each concept can be defined by the set of its neighbors in the graph. Formally, the skip-gram optimization function would be maximizing:

$$\mathscr{L}_p = \frac{1}{|C|} \sum_{i=1}^{|C|} \sum_{(c_i, c_j) \in E} \log\ p(c_j | c_i) \tag{4}$$

Note that, while maximizing $\mathscr{L}_p$, the number of training examples generated from $(c_i, c_j) \in E$, is equal to their co-occurrence count n_{c_i, c_j}. The incorporation of the concept-concept co-occurrence counts in Probase will result in a dynamic adjustment to the overall likelihood $\mathscr{L}_p$ depending on the counts between pairs of concepts. For example, for highly related concepts the co-occurrence count will be high, and so will be their contribution to $\mathscr{L}_p$ and vice versa. Thus Probase provides another source of conceptual knowledge to generate more concept-concept contexts, and subsequently learn better concept representations.

2.4 Data and Model Training

2.4.1 Wikipedia

We utilized the Wikipedia dump of August 2016[4], which had ∼7 million articles. We extracted articles plain text discarding images and tables. We also discarded *References* and *External links* sections (if any). We pruned articles not under the main namespace[5]. Eventually, our corpus contained ∼5 million articles in total. We preprocessed each article replacing all its references to other Wikipedia articles with the their corresponding article IDs. In case any of the references is a title of a redirect page, we used the page ID of the original page to ensure that all concept mentions were normalized to their article IDs.

[3] https://www.bing.com/

[4] http://dumps.wikimedia.org/enwiki/

[5] Articles which are prefixed with a string then colon before the title name

2.4.2 Microsoft Concept Graph (Probase)

We used Probase data repository[6] which contained ∼5 million unique concepts, ∼12 million unique instances, and ∼85 million is-a relationships. We followed a simple exact string matching between Wikipedia article titles and Probase concept names in order to align the concepts in both KBs and generate the final concepts set.

2.4.3 Training

We call our model Concept Multimodal Embedding (CME). During training, we jointly train our model to maximize $\mathscr{L} = \mathscr{L}_t + \mathscr{L}_p$, which as mentioned before is estimated using the softmax function. Although it is possible to use weighted sum of $\mathscr{L}_t$ and $\mathscr{L}_p$, we opted using unweighted sum as it is simpler to train, and will not to introduce an extra hyperparameter to the learning model. Thus, we let the model learn the best combination between $\mathscr{L}_t$ and $\mathscr{L}_p$ based on the global words/concepts counts and local co-occurrences between pairs of them.

Following Mikolov et al (2013b), we utilize negative sampling to efficiently approximate the softmax function by replacing every $\log\ p(w_O|w_I)$ term in the softmax function (equation 2) with:

$$\log \sigma(\mathbf{v}_{w_O}^{\intercal}\mathbf{u}_{w_I}) + \sum_{g=1}^{k} \mathbb{E}_{w_s \sim P_n(w)}[\log \sigma(-\mathbf{v}_{w_g}^{\intercal}\mathbf{u}_{w_I})] \tag{5}$$

where k is the number of negative samples drawn for each term, and $\sigma(x)$ is the sigmoid function ($\frac{1}{1+e^{-x}}$).

We consider global word and concept statistics when generating the negative samples for training. As in Mikolov et al (2013b), we implement the downsampling trick where words with normalized frequency ($>10^{-3}$) are downsampled. For each training sample, we sample 5 noisy words/concepts as negatives from the uniform distribution raised to 3/4rd power.

For text learning, we use a context window of size 9. We set the vector size to 500 dimensions and train the model for 10 iterations using 12 cores machine with 64GB of RAM. Our model takes ∼15 hours to train. The total vocabulary size is ∼12.7 million including words and concepts.

3 Evaluation

3.1 Analogical Reasoning

Mikolov et al (2013c) introduced this intrinsic evaluation scheme to assess the capacity of the embedding model to learn a vector space with meaningful substructure. Typically, analogies take the form "*a to b* is same as *c to* __*?*" where *a, b*, and *c* are elements of the vocabulary V. Using vector arithmetic, this can be answered by identifying d such that: $d = arg\ \max_d\ Sim(vec(d), vec(b) - vec(a) + vec(c))$, $\forall d \in V - \{a, b, c\}$, where Sim is a similarity function[7]. A good performance on this task indicates the model's ability to learn semantic and syntactic patterns as linear relationships between vectors in the embedding space (Pennington et al (2014)).

[6] https://concept.research.microsoft.com/Home/Download

[7] Cosine similarity or dot product if vectors are normalized.

Dataset/Questions	Semantic	Syntactic	All
Method	(8,869)	(10,675)	(19,544)
Word2Vec$_{sg}$	58	61	59.5
Word2Vec$_{sg_b}$	78.1	**62.8**	69.8
Glove	80.8	61.5	70.3
Glove$_b$	69.5	32.1	49.1
MPME	71.6	54.6	63.1
CME	**91.4**	61.7	**75.2**

Table 1 Results of analogical reasoning, given as percent accuracy (bold indicates best obtained accuracy). Our CME model gives the best result on semantic analogies and higher overall accuracy than all other models.

3.1.1 Dataset

We use the word analogies dataset of Mikolov et al (2013a). The dataset contains 19,544 questions divided into semantic analogies (8,869), and syntactic analogies (10,675). The semantic analogies are questions about country capitals, state cities, country currencies...etc. For example, "*cairo* to *egypt* is same as *paris* to *france*". The syntactic analogies are questions about verb tenses, opposites, and adjective forms. For example, "*big* to *biggest* is same as *great* to *greatest*". In order to leverage the concept vectors, we first identify the corresponding entity of each analogy word and use its vector. If the word has no corresponding entity or corresponds to a disambiguation page under Wikipedia we use its word vector instead.

3.1.2 Compared Systems

We compare our model to various word and entity embedding methods including:

1. **Word embeddings:** a) Word2Vec$_{sg}$, word embedding model trained on Wikipedia using skip-gram Mikolov et al (2013a), b) Word2Vec$_{sg_b}$, a baseline model we created by training the skip-gram model on the same Wikipedia dump we used for our CME model, c) GloVe, word embedding model proposed by Pennington et al (2014), and d) GloVe$_b$, same model by Pennington et al (2014), but trained on the same Wikipedia version used by CME without preprocessing, for fair comparison. We use recommended hyperparameter values in Pennington et al (2014).
2. **Entity mention embeddings:** MPME, a recent model proposed by Cao et al (2017). The model jointly learn embeddings of words and entity mentions by training the skip-gram on Wikipedia, and utilizing anchor texts to generate multi-prototype entity mention embeddings.

3.1.3 Results

We report the accuracy scores of analogical reasoning in Table 1. As we see, our CME model outperforms all other models by significant percentages on the semantic analogies. The closest performing model (Glove) is ∼10% less accurate. Performance on syntactic analogies is still very competitive to Word2Vec$_{sg_b}$ and GloVe. Overall, our model is ∼ 5% better than the closest performing model.

3.1.4 Error Analysis

Local context window models like ours generally perform better on semantic analogies than syntactic ones. This indicates that syntactic regularities in most textual corpora are more difficult

to capture, using embeddings, than semantic regularities. A possible reason could be the more morphological variations of verbs and adjectives than nouns. Our model training is even more biased toward capturing semantic relationships between concepts by incorporating knowledge from Probase concept graph. This bias caused our model to produce some semantic predictions on the syntactic analogies compared to the Word2Vec$_{sg_b}$ baseline, returning a semantically related word to the answer. For instance, our model predicted *"fast"* rather than *"slows"* 9 times compared to 2 times by Word2Vec$_{sg_b}$. And *"large"* rather than *"smaller"* 14 times compared to 1 time by Word2Vec$_{sg_b}$, Another set of errors were predicting the correct word but with wrong ending especially *"ing"*. For instance, *"implementing"* rather than *"implements"* 27 times compared to 19 time by Word2Vec$_{sg_b}$. We argue that, despite this bias, our CME model still produces very competitive performance compared to other models on syntactic analogies. And more importantly, emphasizing the semantic relatedness between concepts during training contributes to the significant accuracy gains on the semantic analogies.

Algorithm 1: Classification + Bootstrapping

Input: $\mathbf{U}=\{(l_1,\mathbf{u}_{l_1}),...,(l_n,\mathbf{u}_{l_n})\}$: labels + embeddings
$\mathbf{D}=\{(d_1,\mathbf{v}_{d_1}),...,(d_m,\mathbf{v}_{d_m})\}$: instances + embeddings
N: number of bootstrap instances
Result: $\mathbf{L}=\{...,(d_i,l_j),...\}$: label assignment for each instance

repeat
  candidates $\leftarrow \{l_1:\phi,...,l_n:\phi\}$
  foreach $(d,\mathbf{v}_d)\in\mathbf{D}$ **do**
    $d_{max\_sim}=0$
    $d_{max\_label}=null$
    foreach $(l,\mathbf{u}_l)\in\mathbf{U}$ **do**
      $sim_l=Sim(\mathbf{v}_d,\mathbf{u}_l)$
      if $sim_l > d_{max\_sim}$ **then**
        $d_{max\_sim}=sim_l$
        $d_{max\_lebel}=l$
      end
    end
    add (d,d_{max_sim}) to **candidates**$[l]$
  end
  foreach $(l,\mathbf{candidates}_l)\in\mathbf{candidates}.items$ **do**
    repeat
      $score_{max}=0$
      $d_{max}=null$
      foreach $(d,score_d)\in\mathbf{candidates}_l$ **do**
        if $score_d > score_{max}$ **then**
          $score_{max}=score_d$
          $d_{max}=d$ ▷ most similar instance so far
        end
      end
      add (d_{max},l) to $\mathbf{L}$ ▷ assign class label
      $\mathbf{u}_l\leftarrow\mathbf{u}_l+\mathbf{v}_d$ ▷ bootstrap label embedding
      remove d from **candidates**$_l$
      remove d from $\mathbf{D}$
    until *N highest scored instances added*
  end
until $\mathbf{D}=\phi$ ▷ *no more instances to classify*

3.2 Concept Learning

Concept learning is a cognitive process which involves classifying a given concept/entity to one or more candidate categories (e.g., *"milk"* as *beverage, dairy product, liquid*...etc). This process is also known as *concept categorization*[8] Li et al (2016).

Automated concept categorization can be viewed through both intrinsic and extrinsic evaluation. Intrinsic because a "good" embedding model would generate clusters of concepts belonging to the same category, and optimally place the category vector at the center of its instances cluster. And extrinsic as the embedding model could be leveraged in many knowledge modeling tasks such as *KB construction* (creating new concepts), *KB completion* (inferring new relationships between concepts), and *KB curation* (removing noisy or assessing weak relationships).

Similar to Li et al (2016), we assign a given concept to a target category using Rocchio classification (Rocchio (1971)), where the centroid of each category is set to the category's corresponding embedding vector. Formally, given a set of n candidate concept categories $G = \{g_1, ..., g_n\}$, an instance concept c, an embedding function f, and a similarity function Sim, then c is assigned to the ith category g_i such that $g_i = arg\ \max_i\ Sim(f(g_i), f(c))$. Under our CME model, the embedding function f would always map the given concept to its vector.

3.2.1 Bootstrapping

We leverage bootstrapping in order to improve the categorization accuracy without the need for labeled data. In the context of concept learning, we start with the vectors of target category concepts as a prototype view upon which categorization assignments are made (e.g., *vec(bird), vec(mammal)...etc*). We leverage bootstrapping by iteratively updating this prototype view with the vectors of concept instances we are most confident. For example, if *"deer"* is closest to *"mammal"* than any other instance in the dataset, then we update the definition of *"mammal"* by performing *vec(mammal)+=vec(deer)*, normalize it, and repeat the same operation for other categories as well. This way, we adapt the initial prototype view to better match the specifics of the given data. Algorithm 1 presents the pseudocode for performing concept categorization with bootstrapping. In our implementation, we bootstrap the category vector with vectors of the most similar **N** instances at a time. Another implementation option might be defining a threshold and bootstrapping using vectors of **N** instances if their similarity scores exceed that threshold.

3.2.2 Datasets

As in Li et al (2016), we utilize two benchmark datasets: 1) Battig test (Baroni and Lenci (2010)), which contains 83 single word concepts (e.g., *cat, tuna, spoon..etc*) belonging to 10 categories (e.g., *mammal, fish, kitchenware..etc*), and 2) DOTA, which was created by Li et al (2016) from Wikipedia article titles (entities) and category names (categories). DOTA contains 300 single-word concepts (DOTA-single) (e.g., *coffee, football, semantics..etc*), and (150) multiword concepts (DOTA-mult) (e.g., *masala chai, table tennis, noun phrase..etc*). Both belong to 15 categories (e.g., *beverage, sport, linguistics...etc*). Performance is measured in terms of the ability of the system to assign concept instances to their correct categories.

[8] In this paper, we use concept learning and concept categorization interchangeably

Dataset/Instances Method	Battig (83)	DOTA-single (300)	DOTA-mult (150)	DOTA-all (450)
WE_{Senna}	44	52	32	45
$WE_{Mikolov}$	74	72	67	72
$TransE_1$	66	72	69	71
$TransE_2$	75	80	77	79
$TransE_3$	46	55	52	54
CE	79	89	85	88
HCE	87	93	91	92
WE_b	77	93	86	91
+bootstrap	88	97	86	90
Wiki-cc_b	72	90	80	87
+bootstrap	81	91	86	87
Probase-cc_b	73	65	70	67
+bootstrap	95	78	81	83
CME	94	91	88	90
+bootstrap	**100**	**99**	**95**	**98**

Table 2 Results of the concept categorization task, given as percent accuracy (bold indicates best obtained accuracy). Our CME model with bootstrapping gives the best results outperforming all other models and baselines.

3.2.3 Compared Systems

We compare our model to various word, entity and category embedding methods including:

1. **Word embeddings:** Collobert et al (2011) model (WE_{Senna}) trained on Wikipedia. Here vectors of multiword concepts are obtained by averaging their individual word vectors.
2. **MWEs embeddings:** Mikolov et al (2013b) model ($WE_{Mikolov}$) trained on Wikipedia. This model jointly learns single and multiword embeddings where MWEs are identified using corpus statistics.
3. **Entity-category embeddings:** which include Bordes et al (2013) embedding model (TransE). This model utilizes relational data between entities in a KB as triplets in the form (entity, relation, entity) to generate representations of both entities and relationships. Li et al (2016) implemented three variants of this model ($TransE_1$, $TransE_2$, $TransE_3$) to generate representations for entities and categories jointly. Two other models introduced by Li et al (2016) are CE and HCE. CE generates embeddings for concepts and categories using category information of Wikipedia articles. HCE extends CE by incorporating Wikipedia's category hierarchy while training the model to generate concept and category vectors.
4. **Other baselines:** we created three baselines: a) WE_b, has word embeddings only and was obtained by training the skip-gram model on the same Wikipedia dump we used for our CME model (cf. equation 1), b) Wiki-cc_b, has concept embeddings only and was obtained by first preprocessing Wikipedia to remove all non-concept tokens, and then training the skip-gram model on concept-concept contexts (cf. equation 3 where each token t is a concept mention), and c) Probase-cc_b, has concept embeddings only and was obtained by training the adapted skip-gram model on Probase concept graph (cf. equation 4).
   These baselines are meant to quantify and analyze the contribution of each type of information individually. Specifically, entity-entity in Wikipedia conceptual contexts, entity-entity in Probase knowledge graph, and word-word in Wikipedia raw contexts.

No	Utterance	Logical form
1	where is **new orleans** where is **ci0**	(lambda $0 e (loc:t **new_orleans:ci** $0)) (lambda $0 e (loc:t **ci0** $0))
2	what states border the **mississippi river** how many states border **ri0**	(lambda $0 e (and (state:t $0) (next_to:t $0 **mississippi_river:r**))) (count $0 (and (state:t $0) (next_to:t $0 **ri0**)))
3	list flights from **philadelphia** to **san francisco** via **dallas** list flight from **ci0** to **ci1** via **ci2**	(lambda $0 e (and (flight $0) (from $0 **philadelphia:ci**) (to $0 **san_francisco:ci**) (stop $0 **dallas:ci**))) (lambda $0 e (and (flight $0) (from $0 **ci0**) (to $0 **ci1**) (stop $0 **ci2**)))
4	flights from **jfk** or **la guardia** to **cleveland** flight from **ap0** or **ap1** to **ci0**	(lambda $0 e (and (flight $0) (or (from $0 **jfk:ap**) (from $0 **lga:ap**)) (to $0 **cleveland:ci**))) (lambda $0 e (and (flight $0) (or (from $0 **ap0**) (from $0 **ap1**)) (to $0 **ci0**)))

Table 3 Example utterances and their corresponding logical forms from the geography and flights domains. Left, utterances before and after argument type identification. Right, logical forms before and after argument type identification. City is mapped to *ci*, Airport to *ap*, and River to *ri*.

3.2.4 Results

We report the accuracy scores of concept categorization[9] in Table 2. Accuracy is calculated by dividing the number of correctly classified concepts by the total number of concepts in the given dataset. Scores of all non-baseline methods are obtained from Li et al (2016). As we can see in Table 2, our CME+bootstrap model outperforms all other models and baselines by significant percentages. It even achieves 100% accuracy on the Battig dataset. With single word concepts, CME achieves the best performance on Battig and competitive performance to WE_b on DOTA-single. When it comes to multiword concepts, our CME model comes second after HCE. In general, baselines which depend only on pure concept-concept contexts (Wiki-cc_b and Probase-cc_b) perform worse than the word-word contexts baseline (WE_b). This indicates the significance of the full concept contextual information obtained when including both other nearby words and other nearby concepts while learning target concept representation.

3.2.5 Analysis

Is bootstrapping a magic bullet? A first look at the results of CME+bootstrap vs. CME might indicate that if bootstrapping is applied to HCE or WE_b which perform better than CME on some datasets, their performance would still be superior. However, the results of WE_b+bootstrap show that the margin of performance gains of bootstrapping is not necessarily proportional to the performance of the model without it. For example, WE_b+bootstrap performs worse than CME_b+bootstrap on DOTA-single, though WE_b was initially better than CME. This means that bootstrapping other better performing models such as HCE might not be as beneficial as it is to CME. The bottom line here is: the model should learn a semantic space with optimal substructures which cluster instances of the same category together, and keep them far from instances of other categories. This is clearly the case with our CME model which ends up having (near-)optimal category vectors with bootstrapping.

[9] From a multi-class classification perspective, the accuracy scores would be equivalent to the clustering purity score as reported in Li et al (2016).

3.3 Argument Type Identification: A Case Study

In this section, we present a case study to analyze the impact of using our concept vectors for unsupervised argument type identification with semantic parsing as an end-to-end task. In a nutshell, semantic parsing is concerned with mapping natural language utterances into executable logical forms Wang et al (2015a). The logical form is subsequently executed on a knowledge base to answer the user question. Table 3 shows some example utterances and their corresponding logical forms from the geography and flights domains.

3.3.1 Argument Identification

As we can notice from the examples in Table 3, user utterances usually contain mentions of entities of various types (e.g., *city*, *state*, and *airport* names). These mentions are typically parsed as arguments in the resulting logical form. Some of these mentions could be rare or even missing in the training data. As noted by Dong and Lapata (2016), this problem reduces the model's capacity to learn reliable parameters for such mentions.

One possible solution is to preprocess the training data, replacing all entity mentions with their type names (e.g., *san francisco* to *city*, *california* to *state*...etc). This step allows the model to see more identical input/output patterns during training, and thus better learn the parameters of such patterns. The model would also generalize better to out of vocabulary mentions because the same preprocessing could be done at test time.

Dong and Lapata (2016) proposed using gazetteers and regular expressions for argument identification. The authors also demonstrated increased accuracy when employing such approach. However, using regular expressions is error prone as the same utterance could be paraphrased in many different ways. In addition, gazetteers usually have low recall, and will not cover many surface forms of the same entity mention.

In this paper, we embrace argument type identification in a totally unsupervised fashion. The idea is to build upon the promising performance we achieved in concept categorization and apply the same scheme to map entity mentions to their corresponding type names. Our unsupervised argument type identification is a four step process: 1) we predefine target entity types and retrieve their corresponding vectors from our CME model, 2) we identify entity mentions in user utterances (e.g., *mississippi river*), 3) we lookup the mention vector in our CME model, and 4) we compute the similarity between the mention vector and each of the predefined target entity types and choose the most similar type if it exceeds a predefined threshold. This scheme is efficient and doesn't require any manually crafted rules or heuristics. The only needed parameter is the similarity threshold which we fix to 0.5 during experiments.

Note that standard off-the-shelf entity recognition systems could help in identifying the entity mentions but not their type names. In domains like flights, we are interested in non standard types such as *airports* and *airlines*. It is also important to distinguish between *city*, *state*, and *country* mentions in the geography domain and not classifying all instances of these categories as the standard *location* type.

Dataset	GEO	ATIS
w/o Identification	68.6	73.2
w/ Identification	77.1	83.7

Table 4 Results of semantic parsing before and after argument type identification, given as percent accuracy. Using CME to identify argument types resulted in improved accuracy on both datasets.

3.3.2 Datasets

We analyze our unsupervised scheme on two datasets[10] : 1) GEO which contains a total of 880 utterances about U.S. geography Zettlemoyer and Collins (2012). The dataset is split into 680 training instances and 200 test instances. Here we target identifying five entity types: *city*, *state*, *river*, *mountain*, and *country*, and 2) ATIS which contains 5,410 utterances about flight bookings split into 4,480 training instances, 480 development instances, and 450 test instances. Here we target identifying six entity types: *city*, *state*, *airline*, *airport*, *day name*, and *month*.

3.3.3 Model & Training

We assess the performance of argument type identification by training Dong and Lapata (2016) neural semantic parsing model[11]. The model utilizes sequence-to-sequence learning with neural attention (see Dong and Lapata (2016) for more details). We use the Seq2Seq variant of the model and do not perform any parameter tuning as our purpose is to analyze the performance before and after argument type identification, and not to get a state-of-the-art performance on these datasets.

3.3.4 Results

We report the parsing accuracy in Table 4. Accuracy is defined as the proportion of the input utterances whose logical form is identical to the gold standard. As we can see, our argument type identification scheme resulted in significant accuracy improvements of $\sim$10% on both datasets.

We present this experiment as a case study for the utility of our embedding model in an end-to-end task. We don't claim superiority over other embedding techniques here, rather we show that the application of our embedding space to infer is-a relationships can be extended successfully to other application areas including but not limited to: 1) unsupervised argument type identification, and 2) inferring is-a relationship of other categories (*city, state, airline, airport, day name...etc*) than those categories in the concept learning datasets (DOTA and Battig).

3.3.5 Error Analysis

Training the Seq2Seq semantic parsing model on preprocessed data is clearly beneficial as the results in Table 4 show. Without argument identification, the model is prone to the out of vocabulary problem. For example, on GEO we spotted 24 test instances with entities not mentioned in the training data (e.g., *new jersey*, *chattahoochee river*). The same on ATIS with 23 instances. Another source of errors was due to rare mentions. For example, *"portland"* appeared once in GEO training data.

[10] We obtained the raw dataset files by contacting the authors of Dong and Lapata (2016)

[11] https://github.com/donglixp/lang2logic

Our scheme demonstrated good ability to capture most entity mentions and map them to their correct type names. However, there was some subtle failure cases. For example, in *"what length is the mississippi"*, our scheme mapped *"mississippi"* to the *state*, while it was mapped to the *river* in the gold standard logical form. Another example was mapping *"new york"* to the *city* in *"what is the density of the new york"*, while it was mapped to the *state* in the gold standard.

Overall, the results show competitive performance of our unsupervised method compared to the tedious and error prone argument type identification methods. The analysis also shows superior generalization performance when using unsupervised argument identification with utterances containing out of vocabulary and rare mentions.

4 Related Work

Neural embedding models have been proposed to learn distributed representations of concepts and entities. Song and Roth (2015) proposed using the popular Word2Vec model of Mikolov et al (2013a) to obtain the embeddings of each concept by averaging the vectors of the concept's individual words. For example, the embeddings of "*Microsoft Office*" would be obtained by averaging the embeddings of "*Microsoft*" and "*Office*" obtained from the Word2Vec model. Clearly, this scheme fails when the semantics of multiword concepts is different from the compositional meaning of their individual words.

More robust entity embeddings can be learned from the entity's corresponding article and/or from the structure of the employed KB (e.g., its link graph) as in Hu et al (2015); Li et al (2016); Yamada et al (2016); and Shalaby and Zadrozny (2017) who all utilize the skip-gram model, but differ in how they define the context of the target concept. However, all these methods utilize one KB only (Wikipedia) to learn entity representations. Our approach, on the other hand, learns better entity representations by exploiting the conceptual knowledge in a weighted KB graph (Probase) and not only from Wikipedia.

Unlike Hu et al (2015) and Li et al (2016) who learn entity embeddings only, our proposed CME model maps both words and concepts into the same semantic space. In addition, compared to Yamada et al (2016) model which also learns words and entity embeddings jointly, we better model the local contextual information of entities and words in Wikipedia viewed as a textual KB. During training, we generate word-word, word-concept, concept-word, and concept-concept contexts (cf. equation 3). In Yamada et al (2016) model, concept-concept contexts are generated from Wikipedia link graph, and not from their raw mentions in Wikipedia text.

Exploiting all concept tokens surrounding a target concept allows us, given another corpus with annotated concept mentions, to easily harness concept-concept contexts even if the corpus has no link structure (e.g., news stories, scientific publications, medical guidelines...etc).

Our model is computationally less costly than those of Hu et al (2015) and Yamada et al (2016) as it requires a few hours rather than days to train using similar computing resources.

Although the learning of the embeddings might seem straightforward, as it uses the standard skip-gram model, we see this as an advantage. On one hand, it allows our training to scale efficiently to huge vocabulary of words and concepts without the need for a lot of preprocessing (e.g., removing low frequent words and phrases as in Wang et al (2014); Fang et al (2016)). On the other hand, to learn from the knowledge graph contexts, we propose simple adaption to the skip-gram model (cf. equation 4), which allows us to use the same dot product scoring function when optimizing for both $\mathscr{L}_t$ and $\mathscr{L}_p$. This is a simpler and more computationally efficient function than the scoring

function proposed by previous approaches which learn from knowledge graphs (cf. Fang et al (2016)'s equation 1).

5 Conclusion & Discussion

Concepts are lexical expressions (single or multiword) that denote an idea, event or an object and typically have a set of properties associated with it. In this paper, we introduced a neural-based approach for learning embeddings of explicit concepts using the skip-gram model. Our approach learns concept representations from mentions in free text corpora with annotated concept mentions. These mentions even if not available could be obtained through state-of-the-art entity linking systems. We also proposed an effective and seamless addition to the skip-gram learning scheme to learn concept vectors from two large scale knowledge bases of different modalities (Wikipedia, and Probase).

We evaluated of the learned concept embeddings intrinsically and extrinsically. Our performance on the analogical reasoning task produced a new state-of-the-art performance of 91% on semantic analogies.

Empirical results on two datasets for performing concept categorization show superior performance of our approach over other word and entity embedding models.

We also presented a case study to analyze the feasibility of using the learned vectors for argument identification with neural semantic parsing. The analysis shows significant performance gains using our unsupervised argument type identification scheme and better handling of out of vocabulary entity mentions.

To our knowledge, this work is the first to combine knowledge from both Wikipedia and Probase into a unified representation. Our concept space contains all Wikipedia article titles (∼5 million). We use Probase as another source of conceptual knowledge to generate more concept-concept contexts, and subsequently learn better concept vectors. In this spirit, we first filter Probase graph keeping only edges whose both vertices are Wikipedia concepts. Using string matching, ∼1 million unique Probase concepts were mapped to Wikipedia articles. Note that we still use the contexts generated from the 5 million Wikipedia concepts, and add to them contexts obtained from the filtered Probase graph. Out of the ∼12.7 million vectors in our model, we have ∼5 million concept vectors and ∼7.7 million word vectors.

One important future improvement is to better match entities from both Wikipedia and Probase. For example, using string edits to increase recall or graph matching techniques to increase precision. Despite using a simple string matching, the performance of our method is superior to other methods utilizing Wikipedia only. It is expected that string matching might produce incorrect mappings. However, it is important to mention that our string matching exploits the redirect pages titles as well as the canonical titles of Wikipedia articles. This increases the recall. For example, in Probase, *nyc*, *city of new york*, *new york city* are all matched with same Wikipedia article *New York City*.

Our initial qualitative analysis shows that it is common to match single-sense Wikipedia concepts (*ss-Wiki*) with multi-sense Probase concepts (*ms-Pro*). However, in many of these cases, the *ms-Pro* is dominated by the *ss-Wiki*. For example, the Wikipedia page for *Tiger* describes the animal. In Probase, *Tiger* is-a *Animal* and *Tiger* is-a *Big cat* has more co-occurrences (917 & 315 respectively) compared to *Tiger* is-a *Dance* (1 co-occurrence). Same for *Rose* which is described in Wikipedia as flowering plant. In Probase, *Rose* is-a *Flower* has (906) and *Rose* is-a *Plant* has (487) co-occurrences compared to *Rose* is-a *Garden* (10) and *Rose* is-a *Odor* (5) co-occurrences. We believe this would

help generating more consistent contexts from Wikipedia and Probase. On the other hand, such multiple sense concepts in Probase could be leveraged for tasks like sense disambiguation and multi-prototype embeddings, along the lines of Camacho-Collados et al (2016), Iacobacci et al (2015), and Mancini et al (2016).

One important aspect of our CME model is its ability to better represent the long tail entities with few mentions. Existing approaches that utilize Wikipedia's link graph treat Wikipedia as unweighted directed KB graph. During training, a context is generated for entities e_1 and e_2 if e_1 has incoming/outgoing link from/to e_2. This mechanism poorly represents rare/infrequent Wikipedia concepts which have few incoming links (i.e. few mentions). We, alternatively, exploit Probase link structure modeling it as a weighted undirected KB graph. We also utilize the co-occurrence counts between pairs of concepts (cf. Figure 1). Therefore, we generate more concept-concept contexts, resulting in better representations of the long-tail concepts. Consider for example *Nightstand* which has in Wikipedia 17 incoming links. In Probase, *Nightstand* is-a *Furniture*, is-a *Casegoods*, and is-a *Bedroom furniture* with co-occurrences 47, 47, and 32 respectively. This is a 100+ more contexts than we can generate from Wikipedia. Even for frequent Wikipedia concepts, by exploiting the co-occurrence counts, our model will reinforce concept-concept relatedness from the many contexts obtained from Probase.

Our aim in this work was to combine the knowledge from both Wikipedia and Probase in a seamless and simple way which is scalable (computationally cheap) and effective. The integration learning scheme and the results show that we can achieve these two goals with high degree of success. It principle, it is possible to perform such integration between Wikipedia and Probase contexts in other ways, which may for example distinguish between syntactic and semantic information in these contexts. However, such approaches will require extra preprocessing in order to prepare such contexts. For instance, Levy and Goldberg (2014) explored learning word embeddings from contexts generated from a dependency parser. We still claim an advantage over such approaches, because they require costly preprocessing in terms of scalability and effectiveness. As demonstrated by the results, our CME model advances the state-of-the-art on both the analogical reasoning and the concept learning tasks, without the need to do expensive preprocessing or training to learn concept representations.

Acknowledgements This work was partially supported by the National Science Foundation under grant number 1624035. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors would like to thank Avik Ray and Yilin Shen from Samsung Research America for their constructive feedback and discussions while developing the case study on the argument type identification task. The authors also appreciate the reviewers valuable and profound comments.

References

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

Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems, pp 2787–2795

Camacho-Collados J, Pilehvar MT, Navigli R (2016) Nasari: Integrating explicit knowledge and corpus statistics for a multilingual representation of concepts and entities. Artificial Intelligence 240:36–64

Cao Y, Huang L, Ji H, Chen X, Li J (2017) Bridge text and knowledge by learning multi-prototype entity mention embedding. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol 1, pp 1623–1633

Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. Journal of Machine Learning Research 12(Aug):2493–2537

Dong L, Lapata M (2016) Language to logical form with neural attention. arXiv preprint arXiv:160101280

Fang W, Zhang J, Wang D, Chen Z, Li M (2016) Entity disambiguation by knowledge and text jointly embedding. In: Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, pp 260–269

Hu Z, Huang P, Deng Y, Gao Y, Xing EP (2015) Entity hierarchy embedding. In: Proceedings of The 53rd Annual Meeting of the Association for Computational Linguistics

Hua W, Wang Z, Wang H, Zheng K, Zhou X (2015) Short text understanding through lexical-semantic analysis. In: Data Engineering (ICDE), 2015 IEEE 31st International Conference on, IEEE, pp 495–506

Iacobacci I, Pilehvar MT, Navigli R (2015) Sensembed: Learning sense embeddings for word and relational similarity. In: ACL (1), pp 95–105

Kim D, Wang H, Oh AH (2013) Context-dependent conceptualization. In: IJCAI, pp 2330–2336

Levy O, Goldberg Y (2014) Dependency-based word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), vol 2, pp 302–308

Li Y, Zheng R, Tian T, Hu Z, Iyer R, Sycara K (2016) Joint embedding of hierarchical categories and entities for concept categorization and dataless classification. arXiv preprint arXiv:160707956

Mancini M, Camacho-Collados J, Iacobacci I, Navigli R (2016) Embedding words and senses together via joint knowledge-enhanced training. arXiv preprint arXiv:161202703

Mikolov T, Chen K, Corrado G, Dean J (2013a) Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013b) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp 3111–3119

Mikolov T, Yih Wt, Zweig G (2013c) Linguistic regularities in continuous space word representations. In: hlt-Naacl, vol 13, pp 746–751

Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014) 12:1532–1543

Phan MC, Sun A, Tay Y, Han J, Li C (2017) Neupl: Attention-based semantic matching and pair-linking for entity disambiguation. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, pp 1667–1676

Ristoski P, Paulheim H (2016) Rdf2vec: Rdf graph embeddings for data mining. In: International Semantic Web Conference, Springer, pp 498–514

Rocchio JJ (1971) Relevance feedback in information retrieval

Shalaby W, Zadrozny W (2017) Learning concept embeddings for efficient bag-of-concepts densification. arXiv preprint arXiv:170203342

Song Y, Roth D (2015) Unsupervised sparse vector densification for short text similarity. In: Proceedings of NAACL

Song Y, Wang H, Wang Z, Li H, Chen W (2011) Short text conceptualization using a probabilistic knowledgebase. In: Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three, AAAI Press, pp 2330–2336

Song Y, Wang S, Wang H (2015) Open domain short text conceptualization: A generative+ descriptive modeling approach. In: IJCAI, pp 3820–3826

Wang Y, Berant J, Liang P, et al (2015a) Building a semantic parser overnight. In: ACL (1), pp 1332–1342

Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph and text jointly embedding. In: EMNLP, vol 14, pp 1591–1601

Wang Z, Zhao K, Wang H, Meng X, Wen JR (2015b) Query understanding through knowledge-based conceptualization

Yamada I, Shindo H, Takeda H, Takefuji Y (2016) Joint learning of the embedding of words and entities for named entity disambiguation. arXiv preprint arXiv:160101343

Zettlemoyer LS, Collins M (2012) Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. arXiv preprint arXiv:12071420

Zwicklbauer S, Seifert C, Granitzer M (2016) Robust and collective entity disambiguation through semantic embeddings. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, ACM, pp 425–434