Representation Learning of Entities and Documents from Knowledge Base Descriptions

Ikuya Yamada[1,3] **Hiroyuki Shindo**[2,3] **Yoshiyasu Takefuji**[4]
ikuya@ousia.jp shindo@is.naist.jp takefuji@sfc.keio.ac.jp

[1] Studio Ousia, [2] Nara Institute of Science and Technology,
[3] RIKEN AIP, [4] Keio University

Abstract

In this paper, we describe *TextEnt*, a neural network model that learns distributed representations of entities and documents directly from a knowledge base (KB). Given a document in a KB consisting of words and entity annotations, we train our model to predict the entity that the document describes and map the document and its target entity close to each other in a continuous vector space. Our model is trained using a large number of documents extracted from Wikipedia. The performance of the proposed model is evaluated using two tasks, namely fine-grained entity typing and multiclass text classification. The results demonstrate that our model achieves state-of-the-art performance on both tasks. The code and the trained representations are made available online for further academic research.

1 Introduction

The problem of learning distributed representations (or embeddings) from a knowledge base (KB) has recently attracted considerable attention. These representations enable us to use the large-scale, human-edited information of a KB in machine learning models, and can be applied in various natural language tasks such as entity linking (Hu et al., 2015; Yamada et al., 2016; Yamada et al., 2017), entity search (Hu et al., 2015), and link prediction (Bordes et al., 2013; Wang et al., 2014).

In this paper, we describe *TextEnt*, a simple neural network model that learns distributed representations of entities and documents from a KB. Specifically, given a document in a KB consisting of words and *contextual* entities (i.e., entities referred from entity annotations in the document), our model predicts the *target* entity explained by the document (see Figure 1), and maps the document and its target entity close to each other in a continuous vector space. Here, words, contextual entities, and target entities are mapped into continuous vectors that are updated throughout the training. In this study, we train the model using documents retrieved from Wikipedia.

One key characteristic of our model is that it enables us to combine the semantic signals obtained from both words and entities in a straightforward manner. The main motivation for using entities in addition to words is to address the problems of ambiguity (i.e., the same words or phrases may have different meanings) and variety (i.e., the same meaning may be expressed using different words or phrases) in natural language. For example, the word *Washington* is ambiguous because it can refer to a US state, or the capital city of the US, or the first US president *George Washington*, and so on. Further, *New York* is sometimes referred to as *NY* or by its nickname, the *Big Apple*. Obviously, entities do not have these problems, because they are uniquely identified in the KB.

To evaluate our model, we address two important tasks using the proposed representations. Firstly, we consider a fine-grained entity typing task (Yaghoobzadeh and Schutze, 2015) to evaluate the quality of the learned entity representations. In this task, the aim is to infer one or more types of each entity (e.g., *athlete*, *airport*, *sports_team*) from a predefined type set. We perform this task using the simple multilayer perceptron (MLP) classifier with the learned entity representations as features. The results show that our method outperforms the state-of-the-art methods by a wide margin.

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/

Saturn is the sixth planet from the Sun and the second-largest in the Solar System, after Jupiter. It is a gas giant with an average radius about nine times that of Earth.

Contextual entities: *Planet, Sun, Solar System, Jupiter, Gas giant, Earth*

Target entity: *Saturn*

Figure 1: Example of a KB document with entity annotations.

Secondly, we consider a multiclass text classification task, which aims to classify documents into a set of predefined classes. This task examines the ability of our model as a generic encoder of arbitrary documents. One important approach adopted here is that we automatically annotate entities appearing in the target documents using a publicly available entity linking system and encode the documents to the document representations in the same manner as the documents in the KB. For this task, the logistic regression classifier is applied to the document representations. Because of the quality of semantic signals obtained from the entities, our method outperforms strong state-of-the-art methods on two popular datasets (i.e., the 20 newsgroups dataset (Lang, 1995) and R8 dataset (Debole and Sebastiani, 2005)). To facilitate further research, our code and the trained representations are available online at `https://github.com/studio-ousia/textent/`.

Our contributions can be summarized as follows:

- We propose *TextEnt*, a simple neural network model that learns distributed representations of entities and documents from a KB. Given a document in a KB consisting of words and contextual entities, our model learns the representations by predicting the target entity explained by the document (see Figure 1). We train our model using large-scale documents extracted from Wikipedia.
- Our proposed model allows us to effectively combine the semantic signals retrieved from both words and entities in a straightforward manner. We demonstrate the effectiveness of this feature by addressing two important tasks: fine-grained entity typing and text classification. Despite the simplicity of our approach, we achieve state-of-the-art results in both tasks.
- We have published our code and the trained representations online to facilitate further academic research.

2 Our Method

In this section, we describe our approach of learning distributed representations of entities and documents from a KB.

2.1 Model

Given a document D in a KB consisting of a set of words $w_1, ..., w_N$ and a set of contextual entities $e_1, ..., e_K$, we train our model to predict the target entity that the document is explaining. We first derive two vector representations of document D: the word-based representation $\mathbf{v}_{D_w}$ and the contextual entity-based representation $\mathbf{v}_{D_e}$. For simplicity, we compute $\mathbf{v}_{D_w}$ and $\mathbf{v}_{D_e}$ by averaging the vector representations of words and those of contextual entities, respectively.

$$\mathbf{v}_{D_w} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{a}_{w_n},\ \mathbf{v}_{D_e} = \frac{1}{K}\sum_{n=1}^{K}\mathbf{b}_{e_n}, \tag{1}$$

where $\mathbf{a}_w \in \mathbb{R}^d$ and $\mathbf{b}_e \in \mathbb{R}^d$ are the vector representations of words and contextual entities, respectively.

We define a probability that represents the likelihood of entity e_t being the target entity of document D as the following softmax function:

$$P(e_t|D) = \frac{\exp({\mathbf{c}_{e_t}}^\top \mathbf{v}_D)}{\sum_{e' \in E_{KB}} \exp({\mathbf{c}_{e'}}^\top \mathbf{v}_D)}, \tag{2}$$

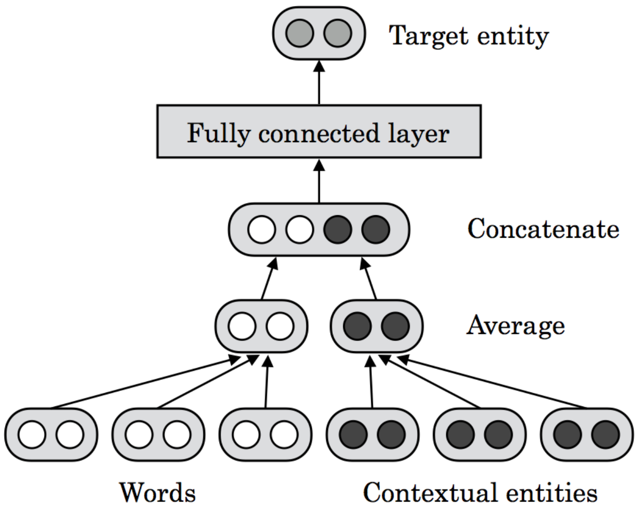

Figure 2: Model architecture of TextEnt.

where E_{KB} is a set of all entities in the KB, $\mathbf{c}_e \in \mathbb{R}^d$ denotes the vector representation of target entity e, and $\mathbf{v}_D \in \mathbb{R}^d$ is the vector representation of document D.

Here, $\mathbf{v}_D$ is computed using a fully connected hidden layer with $\mathbf{v}_{D_w}$ and $\mathbf{v}_{D_e}$ as inputs:

$$\mathbf{v}_D = \mathbf{W}[\mathbf{v}_{D_w}, \mathbf{v}_{D_e}] \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{d \times 2d}$ is a weight matrix, and $[\mathbf{v}_i, \mathbf{v}_j]$ is the concatenation of $\mathbf{v}_i$ and $\mathbf{v}_j$. This layer projects the input vector ($[\mathbf{v}_{D_w}, \mathbf{v}_{D_e}]$) down to d dimensions, and captures the interactions between $\mathbf{v}_{D_w}$ and $\mathbf{v}_{D_e}$.

We use the categorical cross-entropy loss to train the model:

$$\mathcal{L} = -\sum_{(D,e_t)\in\Gamma} \log P(e_t|D), \tag{4}$$

where Γ represents a set of pairs consisting of a document D and its target entity e_t in the KB.

When training our model, the denominator in Eq. (2) is computationally expensive because it involves summation over all KB entities. To address this, we use negative sampling (Mikolov et al., 2013b); specifically, we replace E_{KB} in Eq. (2) with a set consisting of the target entity e_t and k randomly chosen negative entities. Furthermore, to avoid overfitting, we use *word dropout* (Iyyer et al., 2015), which randomly excludes words and contextual entities with a probability p during the training.

We also test models trained using only words (denoted by *TextEnt-word*) and only contextual entities (denoted by *TextEnt-entity*) in our experiments. These variants are created by replacing $\mathbf{v}_D$ in Eq. (2) with $\mathbf{v}_{D_w}$ (*TextEnt-word*) and $\mathbf{v}_{D_e}$ (*TextEnt-entity*). Hereafter, our original model is referred to as *TextEnt-full*.

2.2 Dataset

We trained our model using documents obtained from the April 2016 version of the DBpedia NIF abstract dataset[1], which contains the texts and entity annotations in the first introductory sections of Wikipedia articles.

For computational efficiency, we limited the size of our dataset. In particular, we excluded documents with fewer than five incoming links from other documents if the corresponding entity of the document is not contained in the dataset used in our fine-grained entity typing experiments, presented in Section 3.1. As a result, the number of target documents was 702,388.

We also modified all words to lowercase, and excluded words that make fewer than five appearances and contextual entities that make fewer than three appearances in the documents. Thus, the final dataset contained 242,771 unique words and 327,263 unique contextual entities.

[1]`http://wiki.dbpedia.org`

2.3 Parameters

The parameters to be trained in our model are the weight matrix $\mathbf{W}$ in the fully connected layer and the vector representations of the words, contextual entities, and target entities. The weight matrix was initialized at random and the vector representations were initialized using pre-trained representations. The pre-trained representations of words and entities were learned jointly using the skip-gram model (Mikolov et al., 2013a; Mikolov et al., 2013b) with negative sampling[2]. The corpus was automatically generated by replacing the name of each entity annotation in the Wikipedia documents with a unique identifier of the entity corresponding to that annotation. Note that we used the same pre-trained entity representations to initialize the representations of the contextual entities and the target entities. Additionally, we used all Wikipedia documents obtained from the July 2016 version of Wikipedia dump[3] to build the corpus.

2.4 Implementation Details

The proposed model was implemented using PyTorch[4] and trained with mini-batch stochastic gradient descent (SGD). The mini-batch size was fixed at 100 and the learning rate was automatically controlled by Adadelta (Zeiler, 2012). We trained the model by iterating over the documents in the KB in random order for 50 epochs[5]. For computational efficiency, we used only the first 2,000 words and first 300 entities in the documents. The training took approximately 25 h on an NVIDIA GTX 1080 Ti GPU. Regarding the other hyper-parameters, the representations were set to have $d = 300$ dimensions, the size of the negative entities was $k = 100$, and the dropout probability was set to $p = 0.5$, as recommended in Srivastava et al. (2014)

3 Experiments

To evaluate the models described in the previous section, we conducted fine-grained entity typing and text classification tasks using the learned representations. A description of each task is given in the following subsections. Finally, we qualitatively analyze the learned representations.

3.1 Fine-grained Entity Typing

This section describes the task of fine-grained entity typing (Yaghoobzadeh and Schutze, 2015; Neelakantan and Chang, 2015; Yaghoobzadeh and Schütze, 2017) using the entity representations learned by our proposed models. The aim of this task is to assign each entity with one or more fine-grained types such as *musician* and *film*. Because an entity typing model is capable of predicting the entity types that are missing from the KB, this can be seen as a *knowledge base completion* problem. The task is important because entity type information is often missing from KBs, but is known to be beneficial for various downstream natural language tasks such as entity linking (Ling et al., 2015), coreference resolution (Hajishirzi et al., 2013), and semantic parsing (Liu et al., 2015).

Setup

Our experimental setup follows that of Yaghoobzadeh and Schutze (2015). In particular, we use their entity dataset of 201,933 Freebase[6] entities mapped to 102 entity types based on the FIGER type set (Ling and Weld, 2012). The dataset consists of a training set (50%), development set (20%), and test set (30%). Because the dataset is constructed based on Freebase, we preprocessed the data by mapping each entity to the corresponding entry in Wikipedia and excluded those entities that did not exist in Wikipedia.[7] As a result, we successfully mapped approximately 92% of the entities to Wikipedia, and obtained training,

[2]We used the skip-gram model implemented in the open-source Gensim library with $size = 300$, $window = 10$, $negative = 15$, $min\_count = 3$, and $iter = 5$. Default values were used for other parameters.

[3]We obtained the Wikipedia dump from Wikimedia Downloads: `https://dumps.wikimedia.org/`

[4]`http://pytorch.org`

[5]We experimented using 10, 20, 30, and 50 epochs. All numbers achieved similar performance in our experiments. We used the model trained for 50 epochs because it achieved the best P@1 performance in our fine-grained entity typing task.

[6]`https://developers.google.com/freebase/`

[7]We used the *wikipedia.en_title* property contained in the Freebase dump to create the mapping.

development, and test sets containing 93,350, 37,036, and 55,715 entities, respectively. We publicized the dataset and the code used to generate the dataset at `https://github.com/studio-ousia/textent/`.

Following Yaghoobzadeh and Schutze (2015), we evaluated the models using ranking and classification measures. The ranking measures test how well a model ranks entity types. In particular, we ranked the entity types based on the probabilities assigned by the model and evaluated the ranked list using the precision at 1 (P@1) and breakeven point (BEP)[8].

The classification measures evaluate the quality of the thresholded assignment decisions of a model. The assignment decisions are based on thresholding the probability assigned to each type. The threshold is selected per type by maximizing the F1 score of entities assigned to the type in the development set. We used the accuracy (an entity is correct if all its types and no incorrect types are assigned to it), micro-average F1 (F1 score of all type–entity assignment decisions), and macro-average F1 (F1 score of types assigned to an entity, averaged over entities). These ranking and classification measures are exactly the same as those used in Yaghoobzadeh and Schutze (2015).

Method

We used an MLP classifier with the entity representations as inputs to predict the probability of entity e being a member of type t in the set of possible types T. In particular, we used an MLP with a single hidden layer and the tanh activation function, and an output layer that contains, for each possible type $t \in T$, a logistic regression classifier that predicts the probability of t:

$$\big[P(t_1|e), ..., P(t_{|T|}|e)\big] = \sigma\big(\mathbf{W}_o\, tanh\,(\mathbf{W}_h \mathbf{c}_e)\big), \tag{5}$$

where $\mathbf{c}_e \in \mathbb{R}^d$ is the vector representation of entity e, σ is the sigmoid function, and $\mathbf{W}_h \in \mathbb{R}^{h \times d}$ and $\mathbf{W}_o \in \mathbb{R}^{|T| \times h}$ are the weight matrices corresponding to the hidden layer and the output layer, respectively. The model was trained to minimize the binary cross-entropy loss summed over all entities and types:

$$-\sum_e \sum_t \big(y_{e,t} \log p_{e,t} + (1 - y_{e,t}) \log(1 - p_{e,t})\big), \tag{6}$$

where $y_{e,t} \in \{0, 1\}$ and $p_{e,t}$ denote the ground-truth label and predicted probability, respectively, of entity e being type t. The parameters in $\mathbf{W}_h$ and $\mathbf{W}_o$ are updated in the training stage. Note that the model described here is equivalent to that proposed in Yaghoobzadeh and Schutze (2015).

The model was trained using mini-batch SGD, with the learning rate controlled by Adam (Kingma and Ba, 2014) and the mini-batch size set to 32. The model was trained using the training set and evaluated using the test set. Following Yaghoobzadeh and Schutze (2015), the number of hidden units was set to 200. We also measured P@1 on the development set to locate the best epoch for testing.

Baselines

The performance of our models is compared with that of the following three entity representation models.

- **Figment-GM** (Yaghoobzadeh and Schutze, 2015) is based on the skip-gram model (Mikolov et al., 2013a; Mikolov et al., 2013b) trained using a large corpus with automatically generated entity annotations (i.e., FACC1 (Gabrilovich et al., 2013)). In this experiment, we used the entity representations publicized by the authors[9].

- **Skip-Gram-Wiki** is equivalent to Figment-GM, except that Wikipedia is used as the entity-annotated corpus. This model is also the same as our pre-trained representations described in Section 2.3.

- **Wikipedia2Vec** (Yamada et al., 2016) extends the skip-gram model to learn entity representations based on the contextual words of link anchors in Wikipedia and the internal link structure

[8]BEP is the F1 score at the point in the ranked list at which the precision and recall have the same value.

[9]`https://github.com/yyaghoobzadeh/figment`

	P@1	BEP	Acc.	Mic.	Mac.
TextEnt-full	**.932**	**.948**	**.626**	**.857**	**.842**
TextEnt-word	.909	.933	.611	.838	.820
TextEnt-entity	.882	.912	.560	.702	.770
Figment-GM	.813	.858	.421	.719	.683
Wikipedia2Vec	.925	.943	.600	.844	.822
Wikipedia2Vec (all)	.897	.917	.554	.798	.787
Skip-Gram-Wiki	.900	.927	.576	.831	.804
Skip-Gram-Wiki (all)	.852	.881	.510	.764	.740

Table 1: Results of the entity typing task.

of Wikipedia entities. We used the entity representations trained using the code publicized by the authors[10] and the Wikipedia dump used to train the Skip-Gram-Wiki model.[11]

We used the entity typing method presented above with the entity representations of each baseline model as inputs. Note that, because the Wikipedia2Vec and Skip-Gram-Wiki models were trained using the link anchors in Wikipedia, they do not contain entities that do not appear or are very rare as the link anchor destinations in Wikipedia. To address this, we evaluated these models in the following two settings: (1) using only the entities that exist in the model, and (2) using all entities, including non-existent ones. In the latter setting, we used the zero vector as the representation of non-existent entities. Similar to the latter setting, the former is not a fair comparison because it is typically more difficult to learn good entity representations of rare entities than those of popular entities (Yaghoobzadeh and Schutze, 2015).

Results

Table 1 compares the results of our models with those of the baseline models. Our TextEnt-full model outperforms the baseline models in all measures. In particular, the TextEnt-full model achieves a strong P@1 score of 93.2%, which clearly shows the effectiveness of our entity typing model for many downstream NLP tasks. Moreover, the TextEnt-full model generally performs better than both the TextEnt-word and TextEnt-entity models. This demonstrates the effectiveness of combining the semantic signals obtained from words and entities.

3.2 Multiclass Text Classification

This section describes the multiclass text classification task, which tests the ability of our proposed representations to encode arbitrary documents. Our key assumption here is that, because our proposed representations are trained to predict the corresponding entity of a given document in the KB, they can also classify non-KB documents into classes that are much more *coarse-grained* than entities.

Setup

Following Jin et al. (2016), we used two standard text classification datasets: the 20 newsgroups dataset[12] (denoted by 20NG) (Lang, 1995) and the R8 dataset (Debole and Sebastiani, 2005). The 20NG dataset consists of 11,314 training documents and 7,532 test documents retrieved from 20 different newsgroups. The documents are partitioned nearly equally across the classes. The R8 dataset contains documents from the eight most frequent classes of the Reuters-21578 corpus (Lewis, 1992), which consists of labeled news articles from the 1987 Reuters newswire. The R8 dataset contains 5,485 documents for training and 2,189 documents for testing. Unlike the 20NG dataset, the R8 dataset is imbalanced; the largest class contains 3,923 documents and the smallest class contains 51 documents. For both datasets, we report the

[10]`https://github.com/wikipedia2vec/wikipedia2vec`

[11]We trained the representations with $dim\_size = 300$, $window = 10$, $negative = 15$, $min\_entity\_count = 3$, and $iteration = 5$. Default values were used for other parameters.

[12]We used the *by-date* version of the dataset obtained from `http://qwone.com/~jason/20Newsgroups/`.

accuracy and macro-average F1 score. Furthermore, the development set was formed by selecting 10% of the documents in the training set at random for both datasets.

As preprocessing, we lowercased all words and removed words and entities appearing fewer than five times. Furthermore, we automatically annotated entity mentions in the documents using an entity linking system. In particular, we used TAGME[13] (Ferragina and Scaiella, 2010), a state-of-the-art entity linking system that is freely available and has been frequently used in recent studies (Xiong et al., 2016; Hasibi et al., 2016). However, TAGME returned many irrelevant entity mentions that would act as noise (e.g., *I like* refers to an entity *I Like (Keri Hilson song)*). Thus, we excluded mentions having relevance scores[14] of less than 0.05[15].

Method

For this task, we simply stacked a logistic regression layer onto our TextEnt model to classify documents into the predefined classes. First, we encoded each document (words with entity annotations) and used the resulting document representation (i.e., $\mathbf{v}_D$ in the TextEnt-full model, $\mathbf{v}_{D_w}$ in the TextEnt-word model, and $\mathbf{v}_{D_e}$ in the TextEnt-entity model) as the feature of the logistic regression classifier.

We trained the classifier using the training set of each dataset, and evaluated the classification performance using the corresponding test set. The classifier was trained using mini-batch SGD, with the learning rate controlled by Adam (Kingma and Ba, 2014) and the mini-batch size set to 32. The accuracy on the development set of each dataset was used to locate the best epoch for testing.

Baselines

We adopted the following state-of-the-art models as our baselines.

- **BoW-SVM** is based on a linear support vector machine (SVM) classifier with bag-of-words (BoW) features as inputs. This model outperforms the conventional naive Bayes model (Jin et al., 2016).

- **BoE** (Jin et al., 2016) is an extension of the skip-gram model that learns different word representations per target class. A linear model based on learned word representations was used to classify documents. This model achieves state-of-the-art results on both the 20NG and R8 datasets.

We also used the **Wikipedia2Vec** and **Skip-Gram-Wiki** models described in Section 3.1 as baselines. For this experiment, we simply input the representations of words and entities in these models to our text classification model described in the previous section.

Results

Table 2 compares the results of our proposed models with those of the baseline models. We obtained the BoW-SVM and BoE results from Jin et al. (2016). Our TextEnt-full model outperforms the state-of-the-art models in terms of accuracy and macro F1 score on both the 20NG and R8 datasets. Furthermore, similar to the results of our previous experiment, the TextEnt-full model generally performs better than both the TextEnt-word and TextEnt-entity models. This shows that combining semantic signals obtained from words and entities is also beneficial for text classification tasks.

Furthermore, we conducted a detailed comparison of the BoW-SVM model, BoE model, and TextEnt-full model using the class-level F1 scores on the 20NG dataset (Table 3) and the R8 dataset (Table 4). On the 20NG dataset, our model achieves the best scores in more than half of the classes and provides comparable performance in the other classes. Moreover, our model achieves strong performance in classes with relatively few documents on the R8 dataset. This is because our model successfully captures the strong semantic signals that can only be obtained from entities.

	20NG		R8	
	Acc.	F1	Acc.	F1
TextEnt-full	**.845**	**.839**	**.967**	**.910**
TextEnt-word	.836	.828	.965	.860
TextEnt-entity	.831	.824	.957	.878
BoW-SVM	.790	.783	.947	.851
BoE	.831	.827	.965	.886
Wikipedia2Vec	.829	.823	.965	.881
Skip-Gram-Wiki	.822	.815	.963	.879

Table 2: Results of the text classification task.

Class	SVM	BoE	TextEnt
alt.atheism	.699	.712	**.783**
comp.graphics	.702	.724	**.773**
comp.os.ms-windows.misc	.714	.724	**.742**
comp.sys.ibm.pc.hardware	.673	.706	**.721**
comp.sys.mac.hardware	.778	.792	**.840**
comp.windows.x	.779	**.853**	.846
misc.forsale	.846	**.852**	.829
rec.autos	.817	**.910**	.909
rec.motorcycles	.900	.942	**.943**
rec.sport.baseball	.895	**.947**	.941
rec.sport.hockey	.935	**.967**	.960
sci.crypt	.890	.926	**.934**
sci.electronics	.721	.737	**.757**
sci.med	.803	.869	**.891**
sci.space	.892	.885	**.900**
soc.religion.christian	.823	.877	**.904**
talk.politics.guns	.781	**.833**	.810
talk.politics.mideast	.837	.920	**.944**
talk.politics.misc	**.699**	.687	.678
talk.religion.misc	.590	**.676**	.672

Table 3: Class-level F1 scores in each class on the 20NG dataset.

4 Qualitative Analysis

To investigate how our model encodes documents and entities into the same continuous vector space, we extracted five example sentences from the 20NG dataset and encoded each sentence into a vector using our model. The closest entities to this vector based on the cosine similarity are presented in Table 5. We automatically annotated the entity mentions using TAGME[16], and fed the words and detected entities into the TextEnt-full model. Table 5 presents the sentences, nearest entities, and their corresponding classes in the 20NG dataset. Our model successfully encodes the sentences into vectors that are close to their relevant entities. For example, all nearest entities of the first sentence "*At one time there was speculation that the first spacewalk (Alexei Leonov?) was a staged fake*" are strongly related to the historic Soviet space program. Similar results can be observed in the other four examples.

[13]We used the public Web API service available at `https://services.d4science.org/`.

[14]We used the ρ scores assigned by TAGME.

[15]Excluding entity mentions using the relevance scores is the recommended practice described in the documentation: `https://services.d4science.org/web/tagme/documentation`

[16]We used the same configuration as described in Section 3.2.

Class	Count	SVM	BoE	TextEnt
grain	51	.824	.818	**.889**
ship	144	.781	.783	**.829**
interest	271	.745	.832	**.873**
money-fx	293	.687	.853	**.876**
trade	326	.897	.879	**.918**
crude	374	.929	**.958**	.929
acq	2,292	.956	**.978**	.977
earn	3,923	.986	**.990**	.988

Table 4: Class-level F1 scores with the number of documents in each class on the R8 dataset.

Class	Sentence	Nearest entities
sci.space	At one time there was speculation that the first spacewalk (Alexei Leonov?) was a staged fake.	Sputnik 1 (0.39), Soviet space program (0.38), Soyuz 5 (0.38), Vostok 1 (0.37)
rec.autos	I prefer a manual to an automatic as it should be.	Manual transmission (0.45), Automatic transmission (0.45), Dual-clutch transmission (0.43), Semi-automatic transmission (0.41)
sci.crypt	I change login passwords every couple of months.	Password (0.49), Login (0.46), Privilege (computing) (0.44), Privilege escalation (0.43)
soc.religion. christian	Which version of the Bible do you consider to be the most accurate translation?	Bible translations (0.37), King James Only movement (0.37), Biblical poetry (0.36), The Living Bible (0.36)
sci.med	The blood tests have shown that I have a little too much Hemoglobin	Blood (0.38), Introduction to genetics (0.38), Hemoglobin (0.37), Blood transfusion (0.35)

Table 5: Five example sentences with their top nearest entities using the TextEnt model.

5 Related Work

In recent years, various models for computing distributed representations of text (e.g., sentences and documents) have been proposed (Le and Mikolov, 2014; Kiros et al., 2015; Wieting et al., 2016; Hill et al., 2016). These models typically use large, unstructured corpora for training; however, certain models attempt to learn text representations from structured data. For instance, Hill et al. (2016) proposed a neural network model that learns text representations from online public dictionaries by predicting each dictionary word from its description. Further, Wieting et al. (2016) used a large set of paraphrase pairs obtained from the Paraphrase Database (Ganitkevitch et al., 2013) to learn text representations.

A number of recent models have attempted to learn distributed representations of entities from a KB. For example, Hu et al. (2015) extended the skip-gram model (Mikolov et al., 2013a) to learn entity representations using the hierarchical structure of the KB, and Li et al. (2016) modified the model by Hu et al. to learn both the category representations and entity representations using the category information of the KB. Additionally, *relational embedding* models (Bordes et al., 2013; Wang et al., 2014; Lin et al., 2015) learn the entity representations for link prediction tasks.

Furthermore, some models learn the representations of both words and entities from the KB. A simple method reported in the literature (Yaghoobzadeh and Schutze, 2015; Yamada et al., 2017) is used to derive the pre-trained representations in this study (i.e., preprocessing an entity-annotated corpus by replacing the name of each annotation with the unique identifier of the entity and feeding the corpus into a word embedding model (e.g., skip-gram)). Yamada et al. (2016) proposed Wikipedia2Vec, which extends this idea by using neighboring entities in the internal link graph of the KB as additional contexts for training the model. Note that we used Wikipedia2Vec as a baseline method in the two experiments conducted in this study. Similarly, in their subsequent work (Yamada et al., 2017), they proposed a neural network model that takes entity-annotated text as input and learns word and entity representations by predicting the annotated entities contained in each text. Furthermore, Mancini et al. (2017) proposed a model that maps words and entities in a lexical dictionary (i.e., BabelNet (Navigli and Ponzetto, 2012)) to a single vector space by extending the CBOW model. Unlike our proposed model, these models require users to design a composition function (e.g., vector averaging) to model the semantics of a document using words and entities in it. Moreover, we showed that our approach is highly effective for the two important tasks of fine-grained entity typing and multiclass text classification.

6 Conclusions

In this paper, we described *TextEnt*, a simple neural network model that learns distributed representations of entities and documents from large-scale KB descriptions. We evaluated the performance of the proposed model on fine-grained entity typing and text classification tasks, and achieved state-of-the-art results in both cases, which clearly demonstrates the effectiveness of our approach. In future work, we will explore the applicability of our model to broader NLP tasks such as entity search and KB-based question answering.

Acknowledgements

We would like to thank the anonymous reviewers for their careful reading of our manuscript and their helpful comments and suggestions.

References

[Bordes et al.2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795.

[Debole and Sebastiani2005] Franca Debole and Fabrizio Sebastiani. 2005. An Analysis of the Relative Hardness of Reuters-21578 Subsets: Research Articles. *Journal of the American Society for Information Science and Technology*, 56(6):584–596.

[Ferragina and Scaiella2010] Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 1625–1628.

[Gabrilovich et al.2013] Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. 2013. FACC1: Freebase Annotation of ClueWeb Corpora, Version 1 (Release date 2013-06-26, Format version 1, Correction level 0).

[Ganitkevitch et al.2013] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764.

[Hajishirzi et al.2013] Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke Zettlemoyer. 2013. Joint Coreference Resolution and Named-Entity Linking with Multi-Pass Sieves. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 289–299.

[Hasibi et al.2016] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting Entity Linking in Queries for Entity Retrieval. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 209–218.

[Hill et al.2016] Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to Understand Phrases by Embedding the Dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

[Hu et al.2015] Zhiting Hu, Poyao Huang, Yuntian Deng, Yingkai Gao, and Eric Xing. 2015. Entity Hierarchy Embedding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1292–1300.

[Iyyer et al.2015] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep Unordered Composition Rivals Syntactic Methods for Text Classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691.

[Jin et al.2016] Peng Jin, Yue Zhang, Xingyuan Chen, and Yunqing Xia. 2016. Bag-of-embeddings for Text Classification. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2824–2830.

[Kingma and Ba2014] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980v9*.

[Kiros et al.2015] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-Thought Vectors. In *Advances in Neural Information Processing Systems 28*, pages 3294–3302.

[Lang1995] Ken Lang. 1995. NewsWeeder: Learning to Filter Netnews. *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339.

[Le and Mikolov2014] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning*, volume 32, pages 1188–1196.

[Lewis1992] David D. Lewis. 1992. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37–50.

[Li et al.2016] Yuezhang Li, Ronghuo Zheng, Tian Tian, Zhiting Hu, Rahul Iyer, and Katia Sycara. 2016. Joint Embedding of Hierarchical Categories and Entities for Concept Categorization and Dataless Classification. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 2678–2688.

[Lin et al.2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 2181–2187.

[Ling and Weld2012] Xiao Ling and Daniel S. Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, pages 94–100.

[Ling et al.2015] Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design Challenges for Entity Linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

[Liu et al.2015] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical Word Embeddings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2418–2424.

[Mancini et al.2017] Massimiliano Mancini, Jose Camacho-Collados, Ignacio Iacobacci, and Roberto Navigli. 2017. Embedding Words and Senses Together via Joint Knowledge-Enhanced Training. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, pages 100–111.

[Mikolov et al.2013a] Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the 2013 International Conference on Learning Representations*, pages 1–12.

[Mikolov et al.2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

[Navigli and Ponzetto2012] Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *Artificial Intelligence*, 193(Supplement C):217–250.

[Neelakantan and Chang2015] Arvind Neelakantan and Ming-Wei Chang. 2015. Inferring Missing Entity Type Instances for Knowledge Base Completion: New Dataset and Methods. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–525.

[Srivastava et al.2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

[Wang et al.2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph and Text Jointly Embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1591–1601.

[Wieting et al.2016] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards Universal Paraphrastic Sentence Embeddings. *Proceedings of the 2016 International Conference on Learning Representations.*

[Xiong et al.2016] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2016. Bag-of-Entities Representation for Ranking. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 181–184.

[Yaghoobzadeh and Schutze2015] Yadollah Yaghoobzadeh and Hinrich Schutze. 2015. Corpus-level Fine-grained Entity Typing Using Contextual Information. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 715–725.

[Yaghoobzadeh and Schütze2017] Yadollah Yaghoobzadeh and Hinrich Schütze. 2017. Multi-level Representations for Fine-Grained Typing of Knowledge Base Entities. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 578–589.

[Yamada et al.2016] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259.

[Yamada et al.2017] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning Distributed Representations of Texts and Entities from Knowledge Base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

[Zeiler2012] Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *arXiv preprint arXiv:1212.5701v1.*