Deep Short Text Classification with Knowledge Powered Attention

Jindong Chen,[1,2] **Yizhou Hu,**[1] **Jingping Liu,**[1] **Yanghua Xiao,**[1,3,4,5*] **Haiyun Jiang**[1]
[1]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University, China
[2]CETC Big Data Research Institute Co.,Ltd., Guizhou, China
[3]Shanghai Institute of Intelligent Electronics & Systems, Shanghai, China
[4]Shuyan Technology, Shanghai, China
[5]Alibaba Group, Zhejiang, China
{chenjd16, huyz15, jpliu17, shawyh, jianghy16}@fudan.edu.cn

Abstract

Short text classification is one of important tasks in Natural Language Processing (NLP). Unlike paragraphs or documents, short texts are more ambiguous since they have not enough contextual information, which poses a great challenge for classification. In this paper, we retrieve knowledge from external knowledge source to enhance the semantic representation of short texts. We take conceptual information as a kind of knowledge and incorporate it into deep neural networks. For the purpose of measuring the importance of knowledge, we introduce attention mechanisms and propose deep Short Text Classification with Knowledge powered Attention (STCKA). We utilize Concept towards Short Text (C-ST) attention and Concept towards Concept Set (C-CS) attention to acquire the weight of concepts from two aspects. And we classify a short text with the help of conceptual information. Unlike traditional approaches, our model acts like a human being who has intrinsic ability to make decisions based on observation (i.e., training data for machines) and pays more attention to important knowledge. We also conduct extensive experiments on four public datasets for different tasks. The experimental results and case studies show that our model outperforms the state-of-the-art methods, justifying the effectiveness of knowledge powered attention.

Introduction

Short text classification is one of important ways to understand short texts and is useful in a wide range of applications including sentiment analysis (Wang et al. 2014), dialog system (Lee and Dernoncourt 2016) and user intent understanding (Hu et al. 2009). Compared with paragraphs or documents, short texts are more *ambiguous* since they have not enough contextual information, which poses a great challenge for short text classification. Existing methods (Gabrilovich and Markovitch 2007; Wang et al. 2014; 2014) for short text classification can be mainly divided into two categories: *explicit representation* and *implicit representation* (Wang and Wang 2016).

For explicit representation, a short text is represented as a sparse vector where each dimension is an explicit feature, corresponding to syntactic information of the short text including n-gram, POS tagging and syntactic parsing (Pang, Lee, and Vaithyanathan 2002). Researchers develop effective features from many different aspects such as knowledge base and the results of dependency parsing. The explicit model is interpretable and easy to understand for human beings. However, the explicit representation usually ignores the context of short text and cannot capture deep semantic information.

In terms of implicit representation, a short text is usually mapped to an implicit space and represented as a dense vector (Mikolov et al. 2013). The implicit model is good at capturing syntax and semantic information in short text based on deep neural networks. However, it ignores important semantic relations such as *isA* and *isPropertyOf* that exist in Knowledge Bases (KBs). Such information is helpful for the understanding of short texts, especially when dealing with previously unseen words. For example, given a short text S1: "*Jay grew up in Taiwan*", the implicit model may treat `Jay` as a new word and cannot capture that `Jay` is a `singer` which is beneficial to classify the short text into the class *entertainment*.

In this paper, we integrate explicit and implicit representation of short texts into a unified deep neural network model. We enrich the semantic representation of short texts with the help of explicit KBs such as YAGO (Suchanek, Kasneci, and Weikum 2008) and Freebase (Bollacker et al. 2008). This allows the model to retrieve knowledge from an external knowledge source that is not explicitly stated in the short text but relevant for classification. As the example shown in S1, the conceptual information as a kind of knowledge is helpful for classification. Therefore, we utilize *isA* relation and associate each short text with its relevant concepts in KB by conceptualization[1]. Afterwards we incorporate the *conceptual information* as *prior knowledge* into deep neural networks.

Although it may seem intuitive to simply integrate conceptual information into a deep neural network, there are still two major problems. First, when conceptualizing the short text, some improper concepts are easily introduced due to the ambiguity of entities or the noise in KBs. For example, in the short text S2: "*Alice has been using Apple for more than 10 years*", we acquire the concepts `fruit` and

[1]Conceptualization refers to the process of retrieving the conceptual information of short text from KBs.

`mobile phone` of `apple` from KB. Obviously, `fruit` is not an appropriate concept here which is caused by the ambiguity of `apple`. Second, it is necessary to take into account the granularity of concepts and the relative importance of the concepts. For instance, in the short text S3: "*Bill Gates is one of the co-founders of Microsoft*", we retrieve the concepts `person` and `entrepreneur` of `Bill Gates` from KB. Although they are both correct concepts, `entrepreneur` is more specific than `person` and should be assigned a larger weight in such a scenario. Prior work (Gabrilovich and Markovitch 2007; Wang et al. 2017) exploited web-scale KBs for enriching the short text representation, but without carefully addressing these two problems.

To solve the two problems, we introduce *attention mechanisms* and propose deep **S**hort **T**ext **C**lassification with **K**nowledge Powered **A**ttention (STCKA). Attention mechanism has been widely used to acquire the weight of vectors in many NLP applications including machine translation (Bahdanau, Cho, and Bengio 2015), abstractive summarization (Zeng et al. 2016) and question answering (Hao et al. 2017). For the first problem, we use *Concept towards Short Text (C-ST) attention* to measure the semantic similarity between a short text and its corresponding concepts. Our model assigns a larger weight to the concept `mobile phone` in S2 since it is more semantically similar to the short text than the concept `fruit`. For the second problem, we use *Concept towards Concept Set (C-CS) attention* to explore the importance of each concept with respect to the whole concept set. Our model assigns a larger weight to the concept `entrepreneur` in S3 which is more discriminative for a specific classification task.

We introduce a *soft switch* to combine two attention weights into one and produce the final attention weight of each concept, which is adaptively learned by our model on different datasets. Then we calculate a weighted sum of the concept vectors to produce the concept representation. Besides, we make full use of both character and word level features of short texts and employ self-attention to generate the short text representation. Finally, we classify a short text based on the representation of short text and its concepts. The main contributions of this paper are summarized as follows:

- We propose deep Short Text Classification with Knowledge Powered Attention. As far as we know, this is the first attention model which combines prior knowledge in KBs to enrich the semantic information of the short text.
- We introduce two attention mechanisms (i.e., C-ST and C-CS attention) to measure the importance of each concept from two aspects and combine them by a soft switch to acquire the weight of concept adaptively.
- We conduct extensive experiments on four datasets for different tasks. The results show that our model outperforms the state-of-the-art methods.

Our Model

Our model STCKA is a knowledge-enhanced deep neural network shown in Figure 1. We provide a brief overview of our model before detailing it. The input of the network is a short text s, which is a sequence of words. The output of the network is the probability distribution of class labels. We use $p(y|s, \phi)$ to denote the probability of a short text being class y, where ϕ is the parameters in the network. Our model contains four modules. *Knowledge Retrieval* module retrieves conceptual information relevant to the short text from KBs. *Input Embedding* module utilizes the character and word level features of the short text to produce the representation of words and concepts. *Short Text Encoding* module encodes the short text by self-attention and produces short text representation q. *Knowledge Encoding* module applies two attention mechanisms on concept vectors to obtain the concept representation p. Next, we concatenate p and q to fuse the short text and conceptual information, which is fed into a fully connected layer. Finally, we use an output layer to acquire the probability of each class label.

Knowledge Retrieval

The goal of this module is to retrieve relevant knowledge from KBs. This paper takes *isA* relation as an example, and other semantic relations such as *isPropertyOf* can also be applied in a similar way. Specifically, given a short text s, we hope to find a concept set $\mathcal{C}$ relevant to it. We achieve this goal by two major steps: *entity linking* and *conceptualization*. Entity linking is an important task in NLP and is used to identify the entities mentioned in the short text (Moro, Raganato, and Navigli 2014). We acquire an entity set $\mathcal{E}$ of a short text by leveraging the existing entity linking solutions (Chen et al. 2018). Then, for each entity $e \in \mathcal{E}$, we acquire its conceptual information from an existing KB, such as YAGO (Suchanek, Kasneci, and Weikum 2008), Probase (Wu et al. 2012) and CN-Probase (Shuyan 2018) by conceptualization. For instance, given a short text "*Jay and Jolin are born in Taiwan*", we obtain the entity set $\mathcal{E}$ = {`Jay Chou`, `Jolin Tsai`} by entity linking. Then, we conceptualize the entity `Jay Chou` and acquire its concept set $\mathcal{C}$ = {`person`, `singer`, `actor`, `musician`, `director`} from CN-Probase.

Input Embedding

The input consists of two parts: short text s of length n and concept set $\mathcal{C}$ of size m. We use three kinds of embeddings in this module including character embedding, word embedding, and concept embedding. *Character embedding layer* is responsible for mapping each word to a high-dimensional vector space. We obtain the character level embedding of each word using Convolutional Neural Networks (CNN). Characters are embedded into vectors, which can be considered as 1D inputs to the CNN, and whose size is the input channel size of the CNN. The outputs of the CNN are max-pooled over the entire width to obtain a fixed-size vector for each word. *Word and concept embedding layer* also maps each word and concept to a high-dimensional vector space. We use pre-trained word vectors (Mikolov et al. 2013) to obtain the word embedding of each word. The dimension of word vectors, character vectors and concept vectors is $\frac{d}{2}$. We concatenate the character embedding vectors and

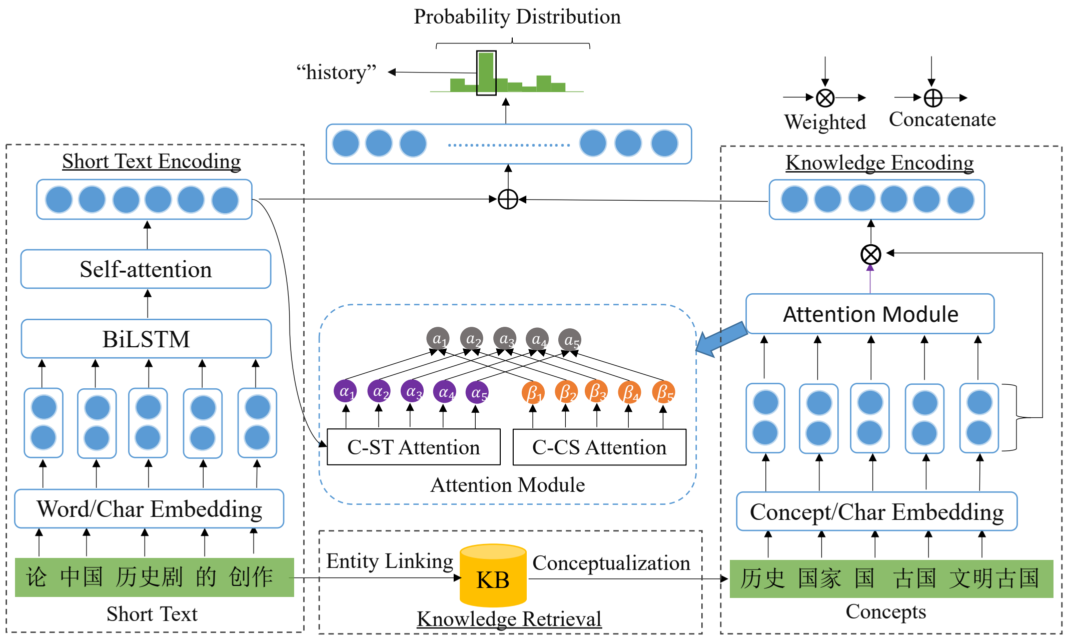


Figure 1: Model architecture. The input short text is *on the creation of Chinese historical plays*. The concepts include *history, country*, etc. The class label is *history*.

word/concept embedding vectors to obtain d-dimensional word/concept representation.

Short Text Encoding

The goal of this module is to produce the short text representation q for a given short text of length n which is represented as the sequence of d-dimensional word vectors $(x_1, x_2, ..., x_n)$. Self-attention is a special case of attention mechanism that only requires a single sequence to compute its representation (Vaswani et al. 2017). Before using self-attention, we add a recurrent neural network (RNN) to transform the inputs from the bottom layers. The reason is explained as follows. Attention mechanism uses weighted sum to generate output vectors, thus its representational power is limited. Meanwhile, RNN is good at capturing the contextual information of sequence, which can further increase the expressive power of attentional network.

In this paper, we employ bidirectional LSTM (BiLSTM) as (Hao et al. 2017) does, which consists of both forward and backward networks to process the short text:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(x_t, \overrightarrow{h_{t-1}}) \tag{1}$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(x_t, \overleftarrow{h_{t+1}}) \tag{2}$$

We concatenate each $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ to obtain a hidden state h_t. Let the hidden unit number for each unidirectional LSTM be u. For simplicity, we denote all the h_ts as $H \in \mathbb{R}^{n \times 2u}$:

$$H = (h_1, h_2, ...h_n) \tag{3}$$

Then, we use the scaled dot-product attention, which is a variant of dot-product (multiplicative) attention (Luong, Pham, and Manning 2015). The purpose is to learn the word dependence within the sentence and capture the internal structure of the sentence. Given a matrix of n query vectors $Q \in \mathbb{R}^{n \times 2u}$, keys $K \in \mathbb{R}^{n \times 2u}$ and values $V \in \mathbb{R}^{n \times 2u}$, the scaled dot-product attention computes the attention scores based on the following equation:

$$A = Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{2u}})V \tag{4}$$

Here Q, K, V are the same matrix and equal to H, $\frac{1}{\sqrt{2u}}$ is the scaling factor. The output of attention layer is a matrix denoted as $A \in \mathbb{R}^{n \times 2u}$. Next, we use max-pooling layer over A to acquire the short text representation $q \in \mathbb{R}^{2u}$. The idea is to choose the highest value on each dimension of vector to capture the most important feature.

Knowledge Encoding

The prior knowledge obtained from external resources such as knowledge bases provides richer information to help decide the class label given a short text. We take conceptual information as an example to illustrate knowledge encoding, and other prior knowledge can also be used in a similar way. Given a concept set $\mathcal{C}$ of size m denoted as $(c_1, c_2, ..., c_m)$ where c_i is the i-th concept vector, we aim at producing its vector representation p. We first introduce two attention mechanisms to pay more attention to important concepts.

To reduce the bad influence of some improper concepts introduced due to the ambiguity of entities or the noise in KBs, we propose *Concept towards Short Text (C-ST) attention* based on vanilla attention (Bahdanau, Cho, and Bengio 2015) to measure the semantic similarity between the i-th concept and short text representation q. We use the following formula to calculate the C-ST attention:

$$\alpha_i = softmax(w_1^T f(W_1[c_i; q] + b_1)) \tag{5}$$

Here α_i denotes the weight of attention from i-th concept towards the short text. A larger α_i means that the i-th concept is more semantically similar to the short text. $f(\cdot)$ is a non-linear activation function such as hyperbolic tangent transformation and $softmax$ is used to normalize attention weight of each concept. $W_1 \in \mathbb{R}^{d_a \times (2u+d)}$ is a weight matrix and $w_1 \in \mathbb{R}^{d_a}$ is a weight vector where d_a is a hyper-parameter, and b_1 is the offset.

Besides, in order to take the relative importance of the concepts into consideration, we propose *Concept towards Concept Set (C-CS) attention* based on *source2token self-attention* (Lin et al. 2017) to measure the importance of each concept with respect to the whole concept set. We define the C-CS attention of each concept as follows:

$$\beta_i = softmax(w_2^T f(W_2 c_i) + b_2) \tag{6}$$

Here β_i denotes the weight of attention from the i-th concept towards whole concept set. $W_2 \in \mathbb{R}^{d_b \times d}$ is a weight matrix and $w_2 \in \mathbb{R}^{d_b}$ is a weight vector where d_b is a hyper-parameter, and b_2 is the offset. The effect of C-CS attention is similar to that of *feature selectio*n. It is a “soft” feature selection which assigns a larger weight to a vital concept, and a small weight (close to zero) to a trivial concept. More details are given in the experimental Section “Knowledge Attention”.

We combine α_i and β_i by the following formula to obtain the final attention weight of each concept:

$$\begin{aligned} a_i &= softmax(\gamma\alpha_i + (1-\gamma)\beta_i) \\ &= \frac{exp(\gamma\alpha_i + (1-\gamma)\beta_i)}{\sum_{k\in[1,m]} exp(\gamma\alpha_k + (1-\gamma)\beta_k)} \end{aligned} \tag{7}$$

Here a_i denotes the final attention weight from the i-th concept towards the short text, $\gamma \in [0, 1]$ is a *soft switch* to adjust the importance of two attention weights α_i and β_i. There are various ways to set the parameter γ. The simplest one is to treat γ as a hyper-parameter and manually adjust to obtain the best performance. Alternatively, γ can also be learned by a neural network automatically. We choose the latter approach since it adaptively assigns different values to γ on different datasets and achieves better experimental results. We calculate γ by the following formula:

$$\gamma = \sigma(w^T[\alpha; \beta] + b) \tag{8}$$

where vectors w and scalar b are learnable parameters and σ is the sigmoid function. In the end, the final attention weights are employed to calculate a weighted sum of the concept vectors, resulting in a semantic vector that represents the concepts:

$$p = \sum_{i=1}^{m} a_i c_i \tag{9}$$

Training

To train the model, we denote all the parameters to be trained as a set ϕ. The training target of the network is used to maximize the log-likelihood with respect to ϕ:

$$\phi \longmapsto \sum_{s\in S} log\ p(y|s, \phi) \tag{10}$$

where S is the training short text set and y is the correct class of short text s.

Experiment

Dataset

We conduct experiments on four datasets, as shown in Table 1. The first one is a *Chinese Weibo emotion analysis*[2] dataset from NLPCC2013 (Zhou et al. 2017). There are 7 kinds of emotions in these weibos, such as anger, disgust, fear and etc. The second one is *product review*[3] dataset from NLPCC2014 (Zhou, Xu, and Gui 2017). The polarity of each review is binary, either positive or negative. The third one is the *Chinese news title*[4] dataset with 18 classes (e.g., entertainment, game, food) from NLPCC2017 (Qiu, Gong, and Huang 2017).

The average word length of the above-mentioned three datasets is over 12. To test whether our model works on much shorter texts, we build the *Topic* dataset whose average word length is 7.99. The Topic dataset is collected from Sogou news (Fu et al. 2015) where each news contains a title, document and topic (e.g., military, politics). We acquire the title as short text and topic as label. Besides, we also report the average number of entities and concepts for each dataset in Table 1. All four datasets are tokenized through the jieba tool[5].

Compared Methods

We compare our proposed model STCKA with the following methods:

- CNN (Kim 2014): This model is a classic baseline for text classification. It uses CNN based on the pre-trained word embedding.
- RCNN (Lai et al. 2015): This method uses a recurrent convolutional neural network for text classification. It applies RNN to capture contextual information and CNN to capture the key components in texts.
- CharCNN (Zhang, Zhao, and LeCun 2015). This method uses CNN with only character level features as the input.
- BiLSTM-MP (Lee and Dernoncourt 2016): This model is proposed for sequential short text classification. It uses a LSTM in each direction, and use max-pooling across all LSTM hidden states to get the sentence representation, then use a multi-layer perceptron to output the classification result.

[2]http://tcci.ccf.org.cn/conference/2013/pages/page04_sam.html
[3]http://tcci.ccf.org.cn/conference/2014/pages/page04_sam.html
[4]http://tcci.ccf.org.cn/conference/2017/taskdata.php
[5]https://github.com/fxsjy/jieba

Datasets	# Class	Training/Validation/Test set	Avg. Chars	Avg. Words	Avg. Ent	Avg. Con
Weibo	7	3771/665/500	26.51	17.23	1.35	3.01
Product Review	2	7648/1350/1000	64.71	40.31	1.82	4.87
News Title	18	154999/27300/10000	20.63	12.02	1.35	2.72
Topic	20	6170/1090/700	15.64	7.99	1.77	4.50

Table 1: Details of the experimental datasets.

Model	Weibo	Topic	Product Review	News Title
CNN	0.3900	0.8243	0.7290	0.7706
RCNN	0.4040	0.8257	0.7280	0.7853
CharCNN	0.4100	0.8500	0.7010	0.7493
BiLSTM-MP	0.4160	0.8186	0.7290	0.7719
BiLSTM-SA	0.4120	0.8200	0.7310	0.7802
KPCNN	0.4240	0.8643	0.7340	0.7878
STCKA	**0.4320**	**0.8814**	**0.7430**	**0.8011**

Table 2: Accuracy of compared models on different datasets.

- BiLSTM-SA (Lin et al. 2017): This method uses BiLSTM and source2token self-attention to encode a sentence into a fixed size representation which is used for classification.
- KPCNN (Wang et al. 2017): This model is the state-of-the-art method for short text classification. It utilizes CNN to perform classification based on word and character level information of short text and concepts.

Settings and Metrics

For *all models*, we use Adam (Kingma and Ba 2014) for learning, with a learning rate of 0.01. The batch size is set to 64. The training epochs are set to 20. We use 50-dimension skip-gram character and word embedding (Mikolov et al. 2013) pre-trained on Sogou News corpus[6]. If a word is unknown, we will randomly initialize its embedding. We also use 50-dimension concept embedding which is randomly initialized. All character embedding, word embedding and concept embedding are trainable and fine-tuned in the training stage, since we hope to learn task-oriented representation. We use 1D CNN with filters of width [2,3,4] of size 50 for a total of 150.

For *our model*, the following hyper-parameters are estimated based on the validation set and used in the final test set: $u = 64, d_a = 70, d_b = 35$. And γ is automatically learned by the neural network, because this method achieves better classification results than using a fixed hyper-parameter. The evaluation metric is *accuracy*, which is widely used in text classification tasks (Lee and Dernoncourt 2016; Wang et al. 2017).

Results

We compare our model STCKA with six strong baselines and the results are shown in Table 2. Our model outperforms traditional Deep Neural Networks (DNNs), including CNN, RCNN, CharCNN, BiLSTM-MP and BiLSTM-SA, without using any knowledge. The main reason is that our model enriches the information of short texts with the help of KBs. Specifically, we incorporate prior knowledge in KBs into DNNs as explicit features which have a great contribution to short text classification. Compared with traditional DNNs, our model is more like a human being who has intrinsic ability to make decisions based on observation (i.e., training data for machines) as well as existing knowledge. In addition, our model also performs better than KPCNN since our model is able to pay more attention to important knowledge due to the attention mechanism. We use C-ST and C-CS attention to measure the importance of knowledge from two aspects and adaptively assign a proper weight to each knowledge of different short texts.

Knowledge Attention

The goal of this part is to verify the effectiveness of two attention mechanisms (i.e., C-ST and C-CS attention). We manually tune the hyper-parameter γ to explore the relative importance of C-ST and C-CS attention. We vary γ from 0 to 1 with an interval of 0.25, and the results are shown in Table 3. In general, the model with $\gamma = 0.25$ works better, but the advantage is not always there for different datasets. For instance, the model with $\gamma = 0.50$ performs the best on Topic dataset. When γ is equal to 0 or 1, the model performs poorly on all four datasets. Using C-ST attention only ($\gamma = 1.00$), the model neglects the relative importance of each concept, which leads to poor performance. On the other hand, merely using C-CS attention ($\gamma = 0.00$), the model ignores the semantic similarity between the short text and concepts. In this case, an improper concept may be assigned with a larger weight which also results in poor performance.

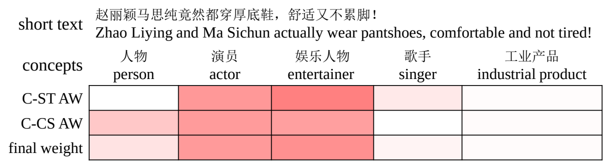
To check whether the attention results conform to our intuition, we also pick some testing examples from the test set of News Title datasets and visualize their attention results in Figure 2. In general, an important concept for classification is assigned with a large weight and vice versa. We also discover some characteristics of our model. First, it is

[6]http://www.sogou.com/labs/resource/list_news.php

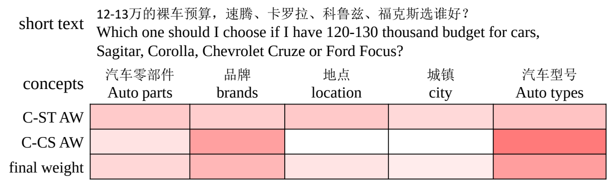
Model	Weibo	Topic	Product Review	News Title
STCKA($\lambda = 0.00$)	0.4280	0.8600	0.7390	0.7972
STCKA($\lambda = 0.25$)	**0.4320**	0.8700	**0.7430**	**0.8007**
STCKA($\lambda = 0.50$)	0.4260	**0.8786**	0.7380	0.8002
STCKA($\lambda = 0.75$)	0.4220	0.8643	0.7380	0.7959
STCKA($\lambda = 1.00$)	0.4160	0.8557	0.7360	0.7965

Table 3: The setting of hyper-parameter λ on our model

interpretable. Given a short text and its corresponding concepts, our model tells us the contribution of each concept for classification by attention mechanism. Second, it is *robust* to the noisy concepts. For example, as shown in Figure 2a, when conceptualizing the short text, we acquire some improper concepts such as `industrial product` which are not helpful for classification. Our model assigns a small attention weight to these concepts since they are irrelevant to the short text and have little similarity to the short text. Third, the effect of C-CS attention is similar to that of *feature selection*. To some extent, C-CS attention is a "soft" feature selection assigning a small weight (nearly to zero) to irrelevant concepts. Therefore, the solution (attention weight) produced by C-CS attention is sparse, which is similar to *L1 Norm Regularization* (Park and Hastie 2007).



(a) The lable of the short text is *fashion*.



(b) The lable of the short text is *car*.

Figure 2: Knowledge attention visualization. Attention Weight (AW) is used as the color-coding.

Power of Knowledge

We use conceptual information as prior knowledge to enrich the representation of short text and improve the performance of classification. The average number of entities and concepts of each dataset are shown in Table 1. To verify the power of knowledge in our model, we pick some testing examples from Topic dataset and illustrate them in Figure 3. These short texts are correctly classified by our model but misclassified by traditional DNNs that do not use any knowledge. In general, the conceptual information plays a crucial role in short text classification, especially when the context of short texts is not enough. As the first example shown in Figure 3, `Revolution of 1911` is a rare word, i.e., occurs less frequently in the training set, and thus is difficult to learn a good representation, resulting in poor performance of traditional DNNs. However, our model solves the *rare and unknown word problem* (Gulcehre et al. 2016) in some degree by introducing knowledge from KB. The concepts such as `history` and `historical event` used in our model are helpful for classifying the short text into the correct class *history*.

Short text: 辛亥革命是犯了“激进主义”错误吗？
Did the Revolution of 1911 make a mistake of "radicalism"?

Concepts:	历史 history	历史事件 historical event	历史书籍 history book

Short text: 南宁 铁路分局创安全新纪录
Nanning Railway Branch creates a new safety record.

Concepts:	交通术语 transportation term	组织结构 organization	城市 city

Figure 3: Two examples for power of knowledge. Underlined phrases are the entities, and the class labels of these two short texts are *history* and *transport* respectively.

Embedding Tunning

We totally use three embeddings in our model. Concept embeddings are randomly initialized and fine-tuned in the training stage. As for character and word embedding, we try three embedding tuning strategies:

- STCKA-rand: The embedding is randomly initialized and then modified in the training stage.
- STCKA-static: Using pre-trained embedding which is kept static in the training.
- STCKA-non-static: Using pre-trained embedding initially, and tuning it in the training stage.

As shown in Table 4, in general, STCKA-non-static performs the best on all four datasets since it makes full use of pre-trained word embedding and fine-tunes it during training phrase to capture specific information on different tasks. Besides, STCKA-rand performs more poorly than STCKA-static on small training datasets such as Weibo and Topic. The reason could be twofold: (1) The amount of labeled samples in the two experimental datasets is too small to tune reliable embeddings from scratch for the in-vocabulary words (i.e., existing in the training data); (2) A lot of out-of-vocabulary words, i.e., absent from the training data, but exist in the testing data. However, STCKA-rand outperforms STCKA-static on large-scale training data such as News Title. Because large-scale training data alleviates two above-

Model	Weibo	Topic	Product Review	News Title
STCKA-rand	0.3780	0.8414	0.7290	0.7930
STCKA-static	0.4240	0.8600	0.7350	0.7889
STCKA-non-static	**0.4320**	**0.8814**	**0.7430**	**0.8011**

Table 4: The impact of different embedding tunning methods on our model

mentioned reasons and enables STCKA-rand to learn task-oriented embeddings which are better for different classification tasks.

Error Analysis

We analyze the bad cases induced by our proposed model on News Title dataset. Most of the bad cases can be generalized into two categories. First, long-tailed entities lack discriminative knowledge in KB due to the incompleteness of KB. For example, in short text "*what does a radio mean to sentry in the cold night alone*", the entity `sentry` is a long-tailed entity without useful concepts in KB. Thus, the short text cannot be classified into the correct class *military*. Second, some short texts are too short and lack contextual information. Even worse, there are no entities mentioned in these short texts which leads to the failure of conceptualization. Therefore, it is difficult to classify the short text "*don't pay money, it's all routines*" into the class *fashion*.

Related Works

Short Text Classification Existing methods for text classification can be divided into two categories: explicit representation and implicit representation. The explicit model depends on human-designed features and represents a short text as a sparse vector. (Cavnar, Trenkle, and others 1994) made full use of simple n-gram features for text classification. (Pang, Lee, and Vaithyanathan 2002; Post and Bergsma 2013) exploited more complex features such as POS tagging and dependency parsing to improve the performance of classification. Some researches introduced knowledge from KBs to enrich the information of short texts. (Gabrilovich and Markovitch 2007) utilized the Wikipedia information to enrich the text representation. (Wang et al. 2014) conceptualized a short text to a set of relevant concepts which are used for classification by leveraging Probase. The explicit model is interpretable and easily understood by human beings, but neglects the context of short texts and cannot capture deep semantic information.

Recently, implicit models are widely used in text classification due to the development of deep learning. The implicit model maps a short text to an implicit space and represents it as a dense vector. (Kim 2014) used CNN with pre-trained word vectors for sentence classification. (Lai et al. 2015) presented a model based on RNN and CNN for short text classification. (Zhang, Zhao, and LeCun 2015) offered an empirical exploration on the use of character-level CNN for text classification. (Lee and Dernoncourt 2016) proposed a RNN model for sequential short text classification. It used BiLSTM and max-pooling across all LSTM hidden states to produce the representation of short text. (Lin et al. 2017) classified the texts by relying on BiLSTM and self-attention. The implicit model is good at capturing syntax and semantic information in short text, but ignores the important prior knowledge that can be acquired from KBs. (Wang et al. 2017) introduced knowledge from Probase into deep neural networks to enrich the representation of short text. However, Wang et al.'s work has two limitations: 1) failing to consider the semantic similarity between short texts and knowledge; 2) ignoring the relative importance of each knowledge.

Attention Mechanism has been successfully used in many NLP tasks. According to the attention target, it can be divided into vanilla attention and self-attention. (Bahdanau, Cho, and Bengio 2015) first used vanilla attention to compute the attention score between a query and each input token in machine translation task. (Hao et al. 2017) employed vanilla attention to measure the similarity between question and answer in question answering task. Self-attention can be divided into two categories including token2token self-attention and source2token self-attention. (Vaswani et al. 2017) applied token2token self-attention to neural machine translation and achieved the state-of-the-art performance. (Lin et al. 2017) used source2token self-attention to explore the importance of each token to the entire sentences in sentence representation task. Inspired by these work, we employ two attention mechanisms from two aspects to measure the importance of knowledge.

Conclusion and Future work

In this paper, we propose deep Short Text Classification with Knowledge Powered Attention. We integrate the conceptual information in KBs to enhance the representation of short text. To measure the importance of each concept, we apply two attention mechanisms to automatically acquire the weight of concepts that is used for generating the concept representation. We classify a short text based on the text and its relevant concepts. Finally, we demonstrate the effectiveness of our model on four datasets for different tasks, and the results show that it outperforms the state-of-the-art methods.

In the future, we will incorporate *property-value information* into deep neural networks to further improve the performance of short text classification. We find that some entities mentioned in short texts lack concepts due to the incompleteness of KB. Apart from conceptual information, entity properties and their values can also be injected into deep neural networks as explicit features. For example, the entity `Aircraft Carrier` has a property-value pair `domain-military`, which is an effective feature for classification.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. *Computer Science*.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 1247–1250. AcM.

Cavnar, W. B.; Trenkle, J. M.; et al. 1994. N-gram-based text categorization. *Ann arbor mi* 48113(2):161–175.

Chen, L.; Liang, J.; Xie, C.; and Xiao, Y. 2018. Short text entity linking with fine-grained topics. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 457–466. ACM.

Fu, J.; Qiu, J.; Wang, J.; and Li, L. 2015. Name disambiguation using semi-supervised topic model. In *International Conference on Intelligent Computing*, 471–480. Springer.

Gabrilovich, E., and Markovitch, S. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis.

Gulcehre, C.; Ahn, S.; Nallapati, R.; Zhou, B.; and Bengio, Y. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Hao, Y.; Zhang, Y.; Liu, K.; He, S.; Liu, Z.; Wu, H.; Zhao, J.; Hao, Y.; Zhang, Y.; and Liu, K. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Meeting of the Association for Computational Linguistics*, 221–231.

Hu, J.; Wang, G.; Lochovsky, F.; Sun, J.-t.; and Chen, Z. 2009. Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web*, 471–480. ACM.

Kim, Y. 2014. Convolutional neural networks for sentence classification. *Eprint Arxiv*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lai, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*, volume 333, 2267–2273.

Lee, J. Y., and Dernoncourt, F. 2016. Sequential short-text classification with recurrent and convolutional neural networks. 515–520.

Lin, Z.; Feng, M.; Santos, C. N. D.; Yu, M.; Xiang, B.; Zhou, B.; and Bengio, Y. 2017. A structured self-attentive sentence embedding.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *Computer Science*.

Moro, A.; Raganato, A.; and Navigli, R. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics* 2:231–244.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up?: sentiment classification using machine learning techniques. *Proceedings of Emnlp* 79–86.

Park, M. Y., and Hastie, T. 2007. L1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69(4):659–677.

Post, M., and Bergsma, S. 2013. Explicit and implicit syntactic features for text classification. 866–872.

Qiu, X.; Gong, J.; and Huang, X. 2017. Overview of the nlpcc 2017 shared task: Chinese news headline categorization. In *National CCF Conference on Natural Language Processing and Chinese Computing*, 948–953. Springer.

Shuyan, T. 2018. Cn-probase concept api. Accessed May 22, 2018. http://shuyantech.com/api/cnprobase/concept.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2008. Yago a large ontology from wikipedia and wordnet. *Web Semantics Science Services and Agents on the World Wide Web* 6(3):203–217.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need.

Wang, Z., and Wang, H. 2016. Understanding short texts. In *the Association for Computational Linguistics (ACL) (Tutorial)*.

Wang, F.; Wang, Z.; Li, Z.; and Wen, J. R. 2014. Concept-based short text classification and ranking. In *ACM International Conference on Conference on Information and Knowledge Management*, 1069–1078.

Wang, J.; Wang, Z.; Zhang, D.; and Yan, J. 2017. Combining knowledge with deep convolutional neural networks for short text classification. In *Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2915–2921.

Wu; Wentao; Li; Hongsong; Wang; Haixun; Zhu; and Kenny, Q. 2012. Probase: a probabilistic taxonomy for text understanding. 481–492.

Zeng, W.; Luo, W.; Fidler, S.; and Urtasun, R. 2016. Efficient summarization with read-again and copy mechanism.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, 649–657.

Zhou, H.; Huang, M.; Zhang, T.; Zhu, X.; and Liu, B. 2017. Emotional chatting machine: Emotional conversation generation with internal and external memory.

Zhou, Y.; Xu, R.; and Gui, L. 2017. A sequence level latent topic modeling method for sentiment analysis via cnn based diversified restrict boltzmann machine. In *International Conference on Machine Learning and Cybernetics*, 356–361.