Convolution-deconvolution word embedding: an end-to-end multi-prototype fusion embedding method for natural language processing

Convolution-deconvolution word embedding: an end-to-end multi-prototype fusion embedding method for natural language processing

Kai Shuang[a], Zhixuan Zhang[a,*], Jonathan Loo[b], Sen Su[a]

[a]State Key Laboratory of Networking & Switching Technology, Beijing University of Posts and Telecommunications, 100876, Beijing, P.R.China
[b]School of Computing and Engineering, University of West London, W5 5RF, UK

Abstract

Existing unsupervised word embedding methods have been proved to be effective to capture latent semantic information on various tasks of Natural Language Processing (NLP). However, existing word representation methods are incapable of tackling both the polysemous-unaware and task-unaware problems that are common phenomena in NLP tasks. In this work, we present a novel Convolution-Deconvolution Word Embedding (CDWE), an end-to-end multi-prototype fusion embedding that fuses context-specific information and task-specific information. To the best of our knowledge, we are the first to extend deconvolution (e.g. convolution transpose), which has been widely used in computer vision, to word embedding generation. We empirically demonstrate the efficiency and generalization ability of CDWE by applying it to two representative tasks in NLP: text classification and machine translation. The models of CDWE significantly outperform the baselines and achieve state-of-the-art results on both tasks. To validate the efficiency of CDWE further, we demonstrate how CDWE solves the polysemous-unaware and task-unaware problems via analyzing the Text Deconvolution Saliency, which is an existing strategy for evaluating the outputs of deconvolution.

Keywords: word embedding, multi-prototype, neural network, natural language processing

*Corresponding author
Email addresses: shuangk@bupt.edu.cn (Kai Shuang), yqzfm123@gmail.com (Zhixuan Zhang), jonathan.loo@uwl.ac.uk (Jonathan Loo), susen@bupt.edu.cn (Sen Su)

1. Introduction

Deep learning based neural network models have achieved great success in many Natural Language Processing (NLP) tasks in the past few years, including learning distributed word, sentence and document representation [32, 63, 62], parsing [24], abstract meaning representation [11, 27], machine translation [46, 41, 12], text classification [22, 45, 37], [21], aspect extraction and sentiment analysis [42, 4, 51, 16, 17, 30, 26], etc. The best results obtained on supervised learning tasks involve an unsupervised learning phase, usually in an unsupervised pre-training step to learn distributed word representations (also known as word embeddings) [63]. Word embedding is a powerful approach to capture the latent semantic information of language by capturing the co-occurrence patterns of words [32, 35], which allows for reasoning about the usage and meaning of words.

Despite the impressive progress, previous methods on word embedding still have limitations on word representation, due to the polysemes, lacking context and unsupervised pre-training process. In embedding models, the object of interest is the conditional probability of a target given its context [28]. For instance, the target corresponds to a word in a given position and the context refers to the words in a window around the target. However, most of the existing methods create a single prototype embedding for each word which are problematic, because many words are intrinsically polysemous. A single-prototype model cannot capture the phenomena of polysemy and task variance, which makes it incapable of dealing with the following problems:

- Polysemous-unaware: Single-prototype representation naturally cannot deal with the phenomena of polysemy. For instance, the word "Hawk" in the sentence "Hawks won the NBA Regular Season" is the name of a basketball team. However, the existing methods will probably treat it as an animal according to the corpus for word embedding.

- Task-unaware: The unsupervised process of existing methods misses the information to distinguish the word with specific meaning contributing to different tasks. For instance, the words "happy" and "sad" have completely contrary contribution in tasks like sentiment analysis. However, in the question classification tasks, samples containing any of these words may belong to the same "emotion" category.

Such characteristics bring major challenges to many NLP tasks, such as text classification, machine translation, etc. For example, misunderstanding the semantic meaning of a key word (e.g. the word contains sentiment information in sentiment analysis) will directly influence the text classification results. Meanwhile, misunderstanding the semantic meaning of the source word in the encoder will influence the generation of target word for the decoder in machine translation.

In order to overcome these issues, researchers need to capture more semantic as well as syntax information. A crucial step to reach this goal is to utilize more advanced text representation models. In this work, we propose a Convolution-Deconvolution Word Embedding (CDWE), an end-to-end multi-prototype fusion embedding that fuses context-specific information and task-specific information to fill this gap. We utilize the deconvolution layer to generate multiple prototypes and select the proper one according to different context words and different tasks.

To the best of our knowledge, we are the first to extend deconvolution (i.e., transposed convolution) to word embedding generation. The motivation of deconvolution neural network comes from its usages in Computer Vision (CV). Deconvolution neural network has been used in CV to capture mid and high level image structure [55] or to generate feature maps with high resolution [15, 36], which achieved remarkable performance during the up-sampling process. In our work, as the polysemous-unaware problem exists in single-prototype models, the up-sampling operation using the deconvolution layer can generate multi-prototype word embeddings to deal with the problem. To be specific, we employed the 2-dimension deconvolution layer after a 1-dimension convolution and pooling layer. The spatial resolution of the embedding matrix gradually increases and a new axis of "prototype" appears after deconvolution operation. In this way, each word corresponds to multiple vectors. Finally, the proper multi-prototype fusion vector for each word is chosen according to its context-specific information along the axis of "prototype'.

On the one hand, the CDWE can tackle the polysemous-unaware problems, because it is designed to learning context-specific information during the generation and fusion process of the multiple prototypes. On the other hand, the CDWE will be fed into a Convolution Neural Network (CNN) [22] or a Bidirectional Long-Short Term Memory (BLSTM) [14, 40] to deal with various NLP tasks in this work. As the CDWE-CNN and the CDWE-

BLSTM are end-to-end models, the parameters in the CDWE are trained according to the specific NLP task and learning task-specific information by transferring the unsupervised representation into supervised model.

In general, the main contributions of this paper are summarized as follows:

- We generated the CDWE, which is a multi-prototype fusion embedding that can solve the polysemous-unaware and task-unaware problems.
- We empirically demonstrate the efficiency of CDWE by applying it to two representative tasks in NLP: text classification and machine translation. The models of CDWE-CNN and CDWE-BLSTM have significantly outperformed the baselines on both tasks.
- The proposed CDWE-BLSTM model with only one fully-connected layer outperforms the existing state-of-the-art methods for text classification on three widely-used datasets.
- The proposed CDWE-BLSTM model with an asynchronous bidirectional decoding strategy [59] outperforms state-of-the-art machine translation systems on NIST Chinese-English datasets.
- We visualized the linguistic information detected by CDWE via an existing strategy, Text Deconvolution Saliency (TDS) [45], and demonstrated how CDWE solved the polysemous-unaware problems and task-unaware problems.

2. Related work

2.1. Word embedding

With the rapid development of distributed word representation, the Neural Language Model (NLM) was proposed and solved the data sparsity problem [39, 47] by using distributed word representations, also known as word embedding [2, 32, 35, 53], which can capture meaningful syntactic and semantic information of words.

Previous works have leveraged existing lexical resources to improve word embedding, such as RCM [54], knowledge embedding [52], [10] SensEmbed [18] and WordNet-based

approaches [20] . Without additional resources, previous works have also utilized character-level information to improve the word embedding, e.g., Santos and Zadrozny [38] and Chen et al. [6]. Most recently there have been many studies on learning the representation of texts on different aspects: Zheng et al. [63] focused on context-specific embedding and tried to tackle the polysemous-unaware problems with larger number of parameters compared with our proposed model. Jiang et al. [19] proposed a framework to measure text distance with latent topics under the assumption that words on the same topic follow a Gaussian distribution. We will compare our model with this framework in Section 4.1.

2.2. Text classification

Text classification is a crucial and fundamental task in many applications, such as web searching, ads matching, sentiment analysis [47, 3, 42]. Previous studies on text classification either rely on human designed features [50] or deep neural networks on distributed representation of texts [32, 47]. Existing literature has demonstrated the potential benefits of using CNN and Recurrent Neural Network (RNN) to extract structural information from short texts. Collobert et al. [7] first used CNN with pre-trained word embedding for text classification. Kim [22] further improved the performance by using multi-channel embedding. Zhang et al. [60] proposed deep neural networks with only character-level information as input. Chen et al. [5] proposed a sequence model to embed these user and product relations information so as to improve the performance of document-level sentiment analysis, which relied on external information on users and products. While such methods work well for large documents, they perform poorly on short texts due to the limited information provided by them. Wang et al. [47] proposed the WCCNN model associated relevant concepts by leveraging explicit knowledge and generating the implicit representation of the short text, which achieves the best performance compared with other existing models based on CNNs, such as MGNC-CNN [61] and CNN-UNI [25]. Most recently, Ruder and Howard [37] proposed the Universal Language Model Fine-tuning for Text classification (ULMFiT), which achieved the best performance compared with other models based on RNNs, such as RCNN [23], Tree-LSTM [43], SA-LSTM [8] and [58]. We will compare our model with the existing state-of-the-art results above in Section 4.1.

2.3. Machine translation

Neural Machine Translation (NMT) is a representative task. Many efficient sequence-to-sequence models have been proposed for NMT in these years and can also be used on other various tasks, such as image caption generation, summarization, question and answering, etc. NMT has shown remarkable progress in recent years compared with conventional Statical Machine Translation (SMT), which needs to explicitly design features to capture translation regularities. Currently, the dominant NMT model mainly consists of a neural encoder and a neural decoder with an attention network [1, 46, 41, 12]. The structure of encoder or decoder can be mainly divided into three types: the RNN, convolutional structure and the transformer [12]. The RNN-based NMT-model achieve better performances compared with other types of encoder or decoder structures based on previous works [41, 59]. Generally, for RNN-based NMT models, the encoder is a bidirectional RNN learning hidden representations of a source sentence in the forward and backward directions. The learned hidden states in two directions are then concatenated to form source annotations. Likewise, the decoder is a forward RNN that adopts the nonlinear function to sequentially generate the target words [1]. Su et al. [41] proposed a variational neural decoder for Variational Recurrent Neural Machine Translation (VRNMT) model and improved the results on NIST Chinese-English tasks [1] compared with other existing methods, such as DL4MT [1], VNMT [56], ATNMT [29], etc. Meanwhile, Zhang et al. [59] proposed an asynchronous bidirectional decoding strategy and achieved state-of-the-art performances on NIST Chinese-English tasks. We will compare our model with the existing state-of-the-art results above in Section 4.2.

2.4. Deconvolution neural network

Deconvolution (i.e., transposed convolution) neural networks have been widely-used in generative models for CV [15, 36], but they have hardly ever been used in NLP. Extending deconvolution networks for text is not straightforward because it brings in an extra dimension to the output compared to the input when applied it to NLP tasks. Zhang et al. [62] proposed a powerful convolutional-deconvolution framework for learning sen-

[1] https://catalog.ldc.upenn.edu/organization/downloads

tence representations, which is used for long paragraph representation learning. This work has demonstrated the effectiveness of the deconvolution in long text representation, but it cannot deal with the polysemous-unaware problem and the task-unaware problem in text classification. Vanni et al. [45] proposed the TDS for visualizing linguistic information detected by neural network. We utilize the TDS for demonstrating the information detected by the CDWE and its ability of solving polysemous-unaware problems and task-unaware problems.

3. Model

We will introduce the generation of CDWE and the overall structure of the models using CDWE as inputs in the following subsection.

3.1. The generation process of CDWE

The broad intuition of tackling polysemy problems can be divided into four steps: First, extract the semantic feature further from the Pre-trained Word Embedding (PWE) through the convolution layer. Second, capture the most common features for each word through the cross-channel pooling layer while reducing the feature redundancy. Third, as the deconvolution operation can expand an extra dimension, it can be utilized to generate multiple prototypes for each word and the new dimension carries the meaning of "prototype". Fourth, choose the proper prototype for each word according to different context and text information along the "prototype" axis. The former three steps are illustrated as the flow above in the structure of Figure 1. The context vectors are generated through the flow below in the structure, consisting of the "Average of context vectors" and the "Broadcasting" in Figure 1, where the "Compute index" and "Selection" processes indicate the fourth step.

The generation of PWE. PWE is effective to capture the basic syntax and semantic information of words, which has been proved in previous studies [2, 32, 35]. In our work, the Continuous Bag-Of-Words (CBOW) [32], one of the main model families of unsupervised pre-training models, is applied to generating PWE, kept static in the training process.

Convolution layer. The width of each filter is set to the same value as the dimension of the PWE and the height of it as 1, which will not involve any interaction feature between

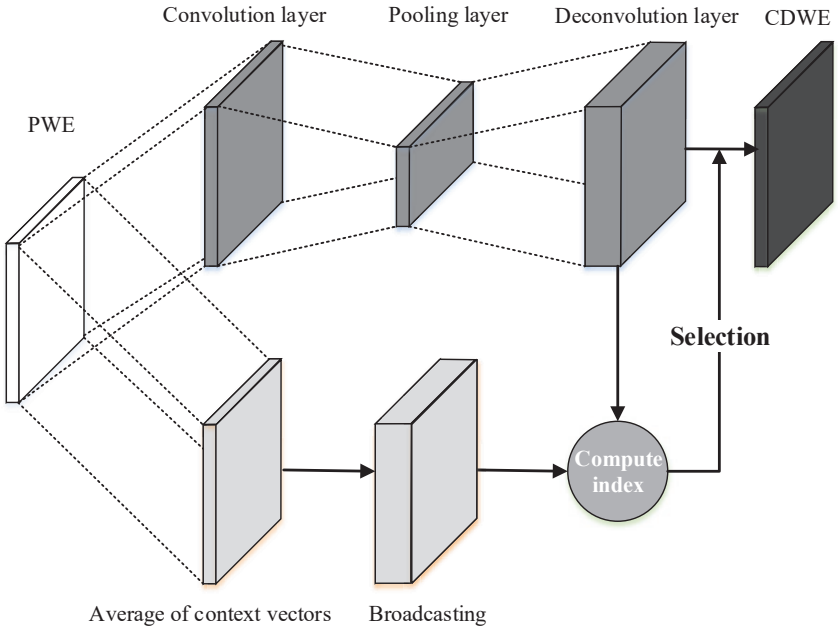

Figure 1: The generation process of CDWE

words. More formally,

$$s_i^j = ReLU(\omega \bullet v_i^p + b) \tag{1}$$

Here, s_i^j denotes the feature extracted from the pre-trained word vector of i-th word v_i^p by the j-th filter. Besides, b is a bias and $ReLU$ is used as the non-linear function for convolution layers in this work.

Pooling layer. The max-pooling operation is applied across different channels, which allows the layer to capture more complex and learnable interactions of cross channel information. [47] More formally,

$$c_i^{\lceil \frac{j}{k} \rceil} = \max\{s_i^j, s_i^{j+1}, ..., s_i^{j+k-1}\} \tag{2}$$

Here, $c_i^{\lceil \frac{j}{k} \rceil}$ denotes the $\lceil \frac{j}{k} \rceil$-th feature extracted from features generated by k convolutional filters, where $\lceil \rceil$ is the ceil operation. And the max-pooling word vector generated by

max-pooling layers is $\widehat{c}_i$.

$$\widehat{c}_i = c_i^1 \oplus c_i^2 \oplus ... \oplus c_i^{\frac{n}{k}} \tag{3}$$

Here n denotes the number of convolution filters, which is set to the same value as the PWE dimension.

Deconvolution layer. The 2-dimension deconvolution (i.e., convolutional transpose) is applied with the window size of (1, k) and the stride of (1, k), to decode the latent representation $\widehat{c}$. The intuition of using the deconvolution layer for CDWE generation is inspired by its applications in the generative process of images [36]. As the deconvolution operation proceeds, the spatial resolution gradually increases and a new axis of "prototype" appears as illustrated in Figure 1. Different prototypes for each word can learn different context-specific information, which also contain the most common information captured by the pooling layer during the training and generative process. The spatial dimension is expanded from $\widehat{c} \in R^{T \times n/k}$ to $D^j \in R^{T \times n}$ in order to match the spatial dimension of the convolution layer of convolution, where T denotes the max length of a sentence. The outputs of the deconvolution layer is $D \in R^{T \times n \times d}$, where d denotes the number of deconvolutional filters, indicating the number of prototypes for each word.

Multi-prototype selection. $D_i \in R^{n \times d}$ denotes the multiple prototypes of the i-th word generated by the deconvolution layer, which consists of d different word vectors e_i^j. The "winner-takes-all" principle is applied to selecting the proper prototypes with context-specific information. That is, the prototype with highest similarity to the context representation vector is selected accordingly. More formally,

$$\begin{aligned} idx_i &= \underset{j=1,2,...,d}{Argmax}(Similarity(e_i^j, ctx_i)) \\ v_i^c &= e_i^{idx_i} \end{aligned} \tag{4}$$

Here, idx_i denotes the index of selected prototype e_i , $v_i^c \in R^n$ is the n-dimension selected word vector of the i-th word and the ctx_i denotes the context representation vector of the i-the words. We utilize a window approach that assumes the meaning of a word depends mainly on its context. More formally,

$$ctx_i = (v_{i-2}^p + v_{i-1}^p + v_i^p + v_{i+1}^p + v_{i+2}^p)/5 \tag{5}$$

For the *Similarity* function, *cosine* similarity is used as the measurement of the similarity between the two vectors.

$$Similarity(e_i^j, ctx_i) = \frac{e_i^j \bullet ctx_i}{\left\|e_i^j\right\| \left\|ctx_i\right\|} \tag{6}$$

When a word appears in different contexts, it may carry different meanings. A number of prototypes should be created for the word so that each prototype can carry a specific meaning. Thus, in order to learn multiple prototypes, each word could be associated with more than one prototypes e_i^j. The network is trained to distinguish the selected word vector from each other so that one of the word vectors e_i^j is chosen in the inferring process by how well the vector fits into the specific context.

3.2. The overall structure

The CNN and BLSTM using the CDWE as input to construct an end-to-end model are shown in Figure 2 and Figure 3. The details of each layers are as follows:

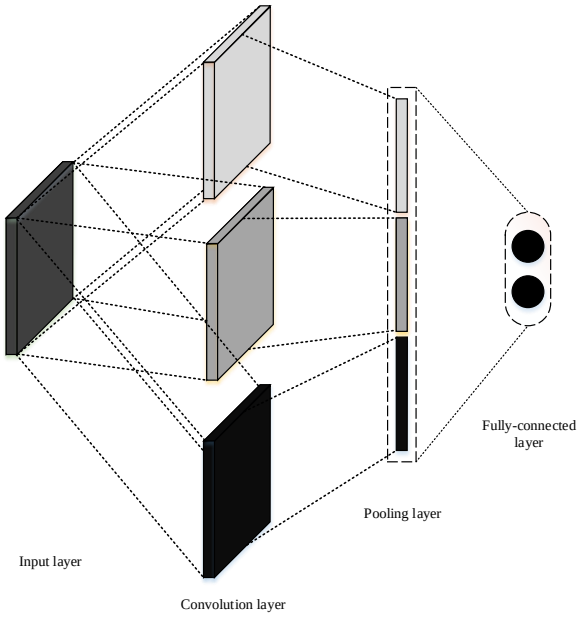

Figure 2: The overall structure of CDWE-CNN

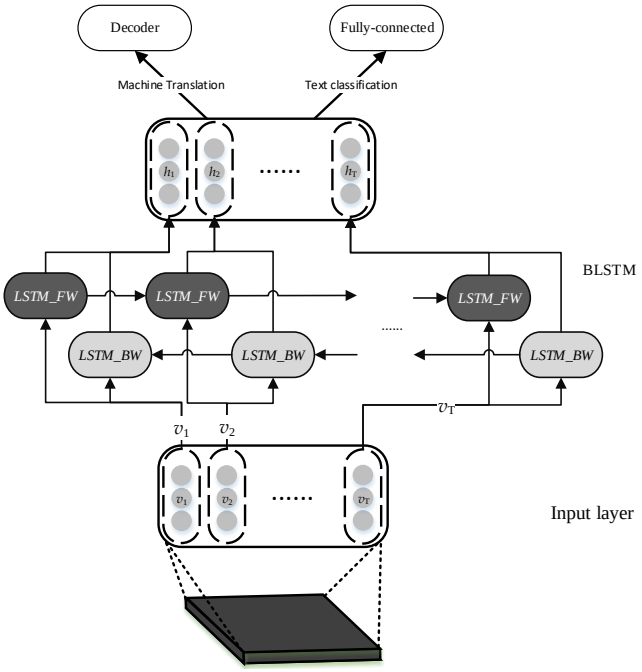

Figure 3: The overall structure of CDWE-BLSTMs

Input layer. It transforms the input into a matrix of CDWE, denoted as $W \in R^{T \times \mathrm{n}}$, where T denotes the max length of a sentence. [2] If the length of a sentence is less than T, 0 is used for padding. The dimension of word embedding of the CDWE is set to n corresponding to the input dimension of PWE. The way to construct W is rather straightforward: suppose $v_i^c \in R^{\mathrm{n}}$ is the m-dimension context-task word vector of the i-th word. W can be obtained by concatenating v_i^c :

$$W = v_1 \oplus v_2 \oplus ... \oplus v_T \tag{7}$$

CNN layer. In the CDWE-CNN model, to get different kinds of features, convolutional filters with three different window sizes are applied in this layer as is shown in Figure 2. The widths of convolutional filters are set as n and the height h of them is treated as a

[2] In the following description of this work, a "sentence" can be an arbitrary span of contiguous text, rather than an actual linguistic sentence. A "sentence" refers to the input token sequence the model, which may be a single sentence or a small paragraph

hyper-parameter. The number of filters with the same size is set to a, so the output of a certain size of h in this layer is $f \in R^{(T-h+1)\times a}$. A max-over-time pooling operation is applied over each feature map [22]. With pooling layers, a fixed-length vector can be obtained from feature maps, which makes this pooling scheme able to deal with variable sentence lengths naturally.

BLSTM layer. In the CDWE-BLSTM models, the CDWE is fed into a standard BLSTM [14, 40], whose number of hidden units is the same as the number of convolutional filters of the CNN layer in total. The BLSTM layer consists of a forward LSTM cell ("LSTM_FW") and a backward LSTM cell, whose inputs are the same at each time step and the results are concatenated, as is shown in Figure 3. The output vectors at all the time step of BLSTM are concatenated and fed into a fully-connected layer or a decoder for text classification or machine translation, respectively. The decoder has the same structure as Zhang et al. [59], which contains two asynchronous LSTMs of different directions with attention mechanism for generating the target words.

Fully-connected layer. The non-linear fully-connected layer is applied to combining different features extracted by the pooling layer (in Figure 2) or BLSTM layer (in Figure 3). The activation function to generate the conditional probability distribution for text classification is $Softmax$. Formally,

$$p_i = \frac{\exp(y_i)}{\sum_{k=1}^{c} \exp(y_k)} \tag{8}$$

Here, y_k is the output of linear operation in fully-connected layer and c denotes the number of classes.

4. Performance evaluations

To demonstrate the efficiency and generalization ability of the CDWE, we perform a series of experiments on two representative tasks: text classification and machine translation. The experimental details and results are shown in the following subsection.

4.1. Experiments on text classification

The experiments are conducted on three widely-used datasets: TREC [33], AG News [47] and Movie Review [22]. Both the CDWE-CNN and the CDWE-BLSTM models significantly outperform the baselines, and the CDWE-BLSTM achieves state-of-the-art performances.

4.1.1. Datasets

The details of the three widely-used datasets in our experiments are listed in Table 1.

Datasets	#class	Training set	Test set
TREC	6	5,452	500
AG News	4	120,000	7,600
Movie Review	2	8,530	2,132

Table 1: A summary of datasets

TREC. This is a question answering dataset[3]. It involves 6 different types of questions, such as whether the question is about a location.

AG News. This dataset is the same as that used in Wang et al. [47]. The data consists of the title AG's corpus of news of four classes, including Business, Sci / Tech, World and Entertainment.

Movie Review. This dataset consists of one sentence per comment on movies. Classification involves detecting positive/negative reviews [22]. The dataset is randomly split into two parts, 80% as the training set and the remaining 20% as test set. In this process, we keep a balanced number of items with each label in the training set.

4.1.2. Experiment settings

The publicly available word2vec[4] tools are used to generate the PWE. Words not presented in the set of pre-trained words are initialized randomly. There are several hyper-parameters in our model, which are set empirically and shown in Table 2.

[3]http://cogcomp.cs.illinois.edu/Data/QA/QC/

[4]https://code.google.com/archive/p/word2vec/

Hyper parameter	Values
number of filters of $h = 4$ in the CNN	100
number of filters of $h = 5$ in the CNN	100
number of filters of $h = 6$ in the CNN	100
number of hidden units in BLSTM layer	300
dropout rate	0.5
PWE dimension	300
pooling windows size in CDWE	10
deconvolution window size	10
number of deconvolution filters	100
batch size	50
learning rate	0.01

Table 2: Hyper parameters

All the trainable parameters in this work are set as Θ. The training target of the model is to maximize the log-likelihood over the training set with respect to Θ:

$$\Theta \rightarrow \sum_{x \in X} \log(p(y|x, \Theta)) \tag{9}$$

Here X denotes the set of training data and Y denotes the set of class. For each $x \in X$, the CNN model computes a score $p(y; x, \Theta)$, $y \in Y$ for each class. The metric for evaluating each model is the accuracy of prediction.

The Adagrad [9] optimizer is applied to optimizing the training process. At the t-th epoch, the parameters are updated as:

$$\Theta_t = \Theta_{t-1} - \frac{\alpha}{\sqrt{\sum_{i-1}^{t} g_i}} g_t \tag{10}$$

Here α is the learning rate and g_t is the gradient at epoch t. All the parameters are initialized from a uniform distribution, which is followed with the settings of many previous studies [47].

4.1.3. Baseline methods

We have compared the proposed model against the following state-of-the-art feature-based methods and deep learning methods:

Word-Concept Embedding + LR. The baseline uses the weighted word embedding as well as concept embedding to represent each sample. This method is applied according to the process in Wang et al. [47].

BoW + SVM. The baseline is proposed by Wang and Manning [50]. The basic idea is to use the traditional Support Vector Machine (SVM) algorithm to build a classifier. For text classification, the unigrams are used as the feature and the weight of each feature is the frequency of each unigram.

LTTR-kNN, LTTR+SVM These baselines represent text via Latent Topic Text Representation (LTTR), which can be used for k-Nearest Neighbor (k-NN) and SVM to classify text introduced in Jiang et al. [19].

CNN. The method utilizes a one-layer CNN for text classification [22]. It utilizes a multi-channel architecture for text embedding. We obtain its source code from the author[5] and use its default settings.

CharCNN. The baseline [60] utilizes a 12-layer convolutional neural network with only character level features as the input. We obtain the source code from the author.[6]

DSCNN. The baseline [58] hierarchically builds textual representations by processing pretrained word embeddings via Long Short-Term Memory networks and subsequently extracting features with convolution operators. As its source code isn't available, we directly use the data results from the paper. Its experimental data didn't contain the results on AG News.

KPCNN, WCCNN These baselines [47] are based on a joint CNN that combines explicit and implicit representations for text classification, which aims to associate relevant concepts with short texts by leveraging explicit knowledge and generating the implicit representation of the short text. The KPCNN integrated the character-level features into the joint embedding to capture fine-grained semantic information while the WCCNN did not.

[5]https://github.com/yoonkim/CNN sentence
[6]https://github.com/zhangxiangxiao/Crepe

ULMFiT. This state-of-the-art method [37] can be used to achieve CV-like transfer learning for any task for NLP. ULMFiT, consisting of three stages and multiple LSTM layers, is now an end-to-end model for text classification. We obtain the source code from the author.[7]

PWE-CNN, PWE-BLSTM These three baselines proposed in our work are presented to demonstrate the efficiency of the proposed structure. We feed the PWE directly into the CNN and BLSTM, respectively.

4.1.4. Results and discussion

The results on all the datasets are shown in Table 3. The CDWE-CNN model significantly outperforms all the baselines in AG News dataset and Movie Review dataset, while the CDWE-BLSTM achieves state-of-the-art in all three datasets. Although the accuracy of CDWE-BLSTM only outperforms that of ULMFiT by 0.50% on TREC datasets, the ULMFiT consists of more complex structures by using three layers of LSTM and three training stages, which is not an end-to-end model. What's more, the results of ULMFiT on the Movie Review is 1.61% lower than CDWE-CNN and 3.16% lower than CDWE-BLSTM due to the polysemous-unaware problem, which will be illustrated in Section 5.

Although character-level features are helpful, it is worth noticing that the CharCNN model [60] does not perform well in our experiment. Due to the texts in these three datasets are relative short, CharCNN is unable to capture enough features with only character-level information. Moreover, WCCNN model integrates the character-level features with concept vectors [47] and has better performances compared with other baseline methods, but it still has polysemous-unaware and task-unaware problems.

We compare the proposed CDWE-CNN and CDWE-BLSTM model with the PWE-CNN and PWE-BLSTM model respectively. The accuracy on three datasets of the layer (the CNN or the BLSTM) with the CDWE as inputs can be promoted by about 3%-6% compared with the original input of PWE, which directly demonstrate the efficiency of CDWE for promoting the performance of existing models in text classification.

Moreover, we investigate the ability of the convolution-deconvolution structure in our

[7] http://nlp.fast.ai/ulmfit

Model	TREC	AG News	Movie Review
WC + LR	53.10%	61.16%	60.34%
BoW + SVM	85.44%	73.17%	77.49%
LTTR + kNN	91.55%	84.98%	81.35%
LTTR + SVM	90.22%	84.45%	82.17%
CNN	90.13%	85.89%	81.60%
CharCNN	76.23%	79.01%	78.53%
DSCNN	95.60%	–	82.20%
KPCNN	93.33%	88.16%	83.11%
WCCNN	91.07%	85.77%	83.57%
ULMFiT	96.40%	88.35%	83.21%
PWE-CNN	89.45%	85.40%	81.20%
PWE-BLSTM	91.21%	86.09%	82.25%
CDWE-CNN	95.90%	89.38%	84.82%
CDWE-BLSTM	96.90%	89.43%	86.37%

Table 3: Comparison of results of different models

model for capturing contextual information in further detail. On the one hand, the performance of a deconvolution layer is influenced by the number of filters. As is illustrated in Figure 4, the performances of both the CDWE-CNN and the CWE-BLSTM on the TREC dataset decline when the number of filters is smaller than 100 or larger than 150. This is because, different from previous word embedding approaches, the convolution filters and deconvolution filters share the same parameters when generating the prototypes of different words. As a result, the multi-prototypes cannot be distinguished by the specific meaning for each word corresponding to different contexts when the number of deconvolution filters goes too small. However, when the number of deconvolution filters goes too large, it is difficult to train the all the deconvolution filters and generate proper context-specific vectors. In this case, a certain prototype vector generated by the deconvolution layer cannot represent a complete meaning, which will cause the information loss during the selection process.

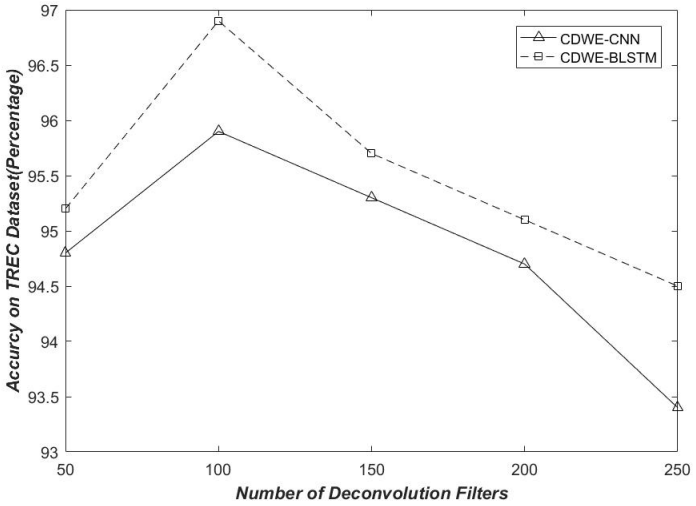

Figure 4: Results of different number of filters on TREC

On the other hand, the representation of context is directly influenced by the context window size. A small window may result in a loss of some long-distance patterns, whereas large windows will lead to ignoring the importance of the closest word. We consider all odd window sizes from 1 to 9 on TREC dataset to train and test the influence on the proposed

model, which is illustrated in Figure 5. The tendencies of the accuracy of the two models are similar and reach the best results when the window size of context is set as 5.

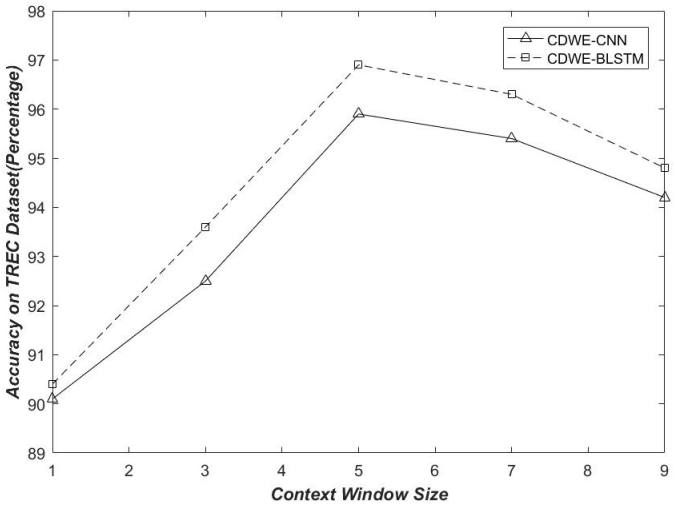

Figure 5: Results of different context window sizes on TREC

4.2. Experiments on machine translation

In order to demonstrate the efficiency of CDWE further, and the generalization ability to other NLP task (e.g. sequence to sequence tasks), the Chinese-English translation experiments are performed. The experimental details and results are shown in the following subsection.

4.2.1. Datasets

For Chinese-English translation, the training data consists of 1.25M bilingual sentences with 27.9M Chinese words and 34.5M English words[41]. These sentence pairs are mainly extracted from LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06. We chose NIST 2002 (MT02) dataset as our development set, and the NIST 2003 (MT03), 2004 (MT04), 2005 (MT05), and 2006 (MT06) datasets as our test sets. Finally, we evaluated the translations using BLEU [34].

4.2.2. Experiment settings

To efficiently train NMT models, we trained each model with sentences of length up to 50 words. In this way, 90.12% Chinese-English parallel sentences were covered in the experiments. Besides, we set the vocabulary size to 30K for Chinese-English translation and mapped all the out-of-vocabulary words in the Chinese-English corpus to a special token UNK. Finally, such vocabularies contained 97.4% Chinese words and 99.3% English words of the Chinese-English corpus. We applied RMSprop [13] to train models for 5 epochs and selected the best model parameters according to the model performance on the development set. There are several hyper-parameters in our model, which are set empirically and shown in Table 4. All the other settings are the same as that in Bahdanau et al. [1].

Hyper parameter	Values
hidden units of BLSTM in encoder	1000
hidden units of LSTMs in decoder	1000
dropout rate	0.3
PWE dimension	620
pooling windows size in CDWE	10
deconvolution window size	10
number of deconvolution filters	100
batch size	80
learning rate	5×10^4
gradient norm	1.0

Table 4: Hyper parameters

4.2.3. Baseline methods

We have compared the proposed CDWE-BLSTM model against the following state-of-the-art SMT and NMT systems:

Moses. This is an open source phrase-based SMT system [8] with default settings and a 4-gram language model trained on the target portion of the training data.

[8] http://www.statmt.org/moses/

DL4MT. This is a re-implementation of the first successful RNN-based NMT system with attention mechanism [1] with slight changes from dl4mt tutorial[9].

ATNMT. This is an Attention-Based NMT (ATNMT) system with two directional decoders [29] which explores the agreement on target-bidirectional NMT.

VRNMT. This is a Variational Recurrent NMT (VRNMT) system that not only explores the utilization of high-level latent random variables but also efficiently captures the strong and complex dependencies between neighboring target words for NMT. [41]

ABDNMT. This Asynchronous Bidirectional Decoding NMT (ABDNMT) model equips the conventional attentional encoder-decoder framework with a backward decoder [59], in order to explore bidirectional decoding for NMT and achieves the state-of-the-art performance in Chinese-English dataset currently. we obtain the source code from the author website [10].

CDWE-BLSTM-DL. This proposed system is the CDWE-BLSTM model equipped with the decoder of DL4MT. As a result, the CDWE of this system is the only component different from the DL4MT.

CDWE-BLSTM-ABD. This proposed system is the CDWE-BLSTM model equipped with the asynchronous bidirectional decoder of ABDNMT. The CDWE of this system is the only component different from the ABDNMT.

In order to make the experimental results comparable, the amount of parameters in the NMT models above are nearly the same and the encoder of all the NMT models are constructed by one BLSTM layer only.

4.2.4. Results and discussion

The experimental results on Chinese-English translation are displayed in Table 5. We also show the performances of some dominant individual models such as COVERAGE [44], MemDec [48], DeepLAU [49] and DMAtten [57] on the dataset. Specifically, the proposed CDWE-BLSTM-ABD model significantly outperforms all the baselines and achieve state-of-the-art performances on all test datasets. By comparing the results between the ABDNMT and CDWE-BLSTM-ABD, the only difference lies in the word embedding matrix. Simi-

[9]https://github.com/nyu-dl/dl4mt-tutorial
[10]https://github.com/DeepLearnXMU/ABDNMT

Model	MT03	MTO4	MT05	MT06	Average
COVERAGE	34.49	38.34	34.91	34.25	35.50
MemDec	36.16	39.81	35.91	35.98	36.97
DeepLAU	39.35	41.15	38.07	37.29	38.97
DMAtten	38.33	40.11	36.71	35.29	37.61
Moses	32.93	34.76	31.31	31.05	32.51
DL4MT	36.59	39.57	35.56	35.29	36.75
ATNMT	38.29	41.01	36.97	36.21	38.10
VRNMT	38.08	41.07	36.82	36.72	38.17
ABDNMT	40.02	42.32	38.84	38.38	39.89
CDWE-BLSTM-DL	40.14	42.11	37.92	38.33	39.64
CDWE-BLSTM-ABD	42.57	44.31	40.39	41.18	41.92

Table 5: Evaluation of the NIST Chinese-English translation task using BLEU scores. Here we displayed the experimental results of the first four models (COVERAGE, MemDec, DeepLAU, DMAtten) reported in [49, 57]

larly, the only difference between DL4MT and CDWE-BLSTM-DL is the word embedding as well. The CDWE-BLSTM-DL can reach comparable results to the existing state-of-the-art system (ABDNMT) though its decoder's performance is not as good as that of the other baselines. We can reach the conclusion that utilizing CDWE as the input of existing NMT system can significantly improve the performance (e.g. more than 2 BLEU scores on all test sets in our experiment). As the understanding of words in source language can directly influence the generation of words in target language, solving the polysemous-unaware problems and task-unaware problems by CDWE will definitely improve the performances of translation. We will introduce how CDWE solve these problems in Section 5.

5. Visualizing linguistic information

In order to visualize the linguistic information detected by CDWE, we choose two samples in Movie Review datasets and compute their z-test scores and TDS scores to demonstrate CDWE's ability to solve the polysemous-unaware problems. We choose a news title in AG News dataset and feed it in the *testing* process of Movie Review dataset and AG

News dataset, respectively, and compute the TDS scores to demonstrate CDWE's ability to solve the task-unaware problem.

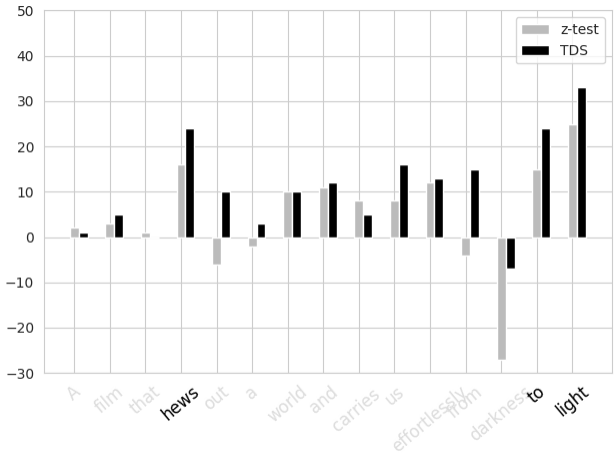

Figure 6: Results of "A film that hews out a world and carries us effortlessly from darkness to light."

Z-test is one of the standard metrics used in linguistic statistics, in particular to measure the occurrences of word collocations. [31] The z-test provides a statistical score of the co-occurrence of a sequence of words appearing more frequently than any other sequence of words of the same length. This score results from the comparison between the frequency of the observed word sequence with the frequency expected in the case of a "Normal" distribution [37]. The TDS score for each word corresponds to the sum along the embedding axis of CDWE. To make the two values comparable, we normalize them. We distinguish between two thresholds for the z-test: over 2 a word is considered as specific and over 5 it is strongly specific (and the opposite with negative values). For the TDS, it is just a matter of activation strength. [37]

As is shown in Figure 6 and Figure 7, two sentences both contain the word "light". When the word "light" is used as a noun, it contains a positive emotion, which may directly determine the classification result. On the contrary, when it is used as a verb or an adjective, it probably contains no emotion information. Both z-test and TDS achieve the highest score on "light" in Figure 6. However, the z-test "misunderstands" the meaning of "light" and emphasizes the wrong part of the sentence, as is illustrated in Figure 7, because the word

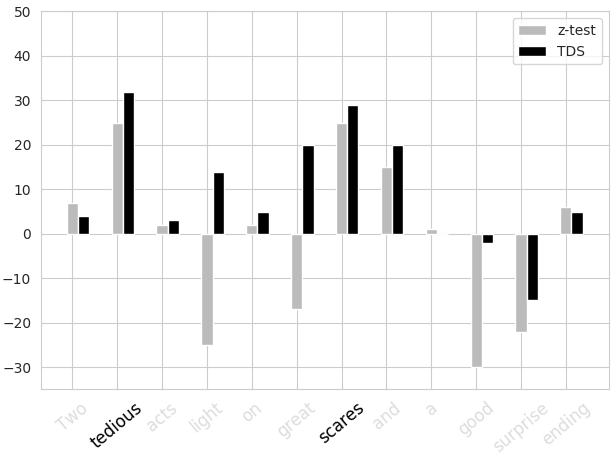

Figure 7: Results of "Two tedious acts light on great scares and a good surprise ending."

"light" in Figure 7 is no longer a key word, which has different meanings. The CDWE treats the same word differently in different samples according to their context-specific information, which results in solving the polysemous-unaware problems. Besides, the word "light" has three more different meanings and usages(e.g. as a verb, a noun or an adjective), appearing 53 times, which is a considerable number, in the Movie Review dataset. There are many other words that have two or more meanings presented in the datasets, such as "bear", "check", "kind", etc. As the proportion of polysemes in the Movie Review dataset is much larger than that in the other datasets, the models utilizing CDWE as inputs obtain more significant improvement of performances on this dataset.

In Figure 8, the word "wins" and the word "tournament" get the highest TDS scores in sentiment classification and news classification, which illustrated that the CDWE can emphasize the different task-specific key words in the *testing* process corresponding to the positive emotion and the "Entertainment" news class, respectively, because the CDWE can learn task-specific information in the *training* process of different datasets.

6. Conclusion and future work

We presented a novel CDWE, which is an end-to-end multi-prototype fusion embedding, to tackle polysemous-unaware and task-unaware problems in NLP tasks. To the best of

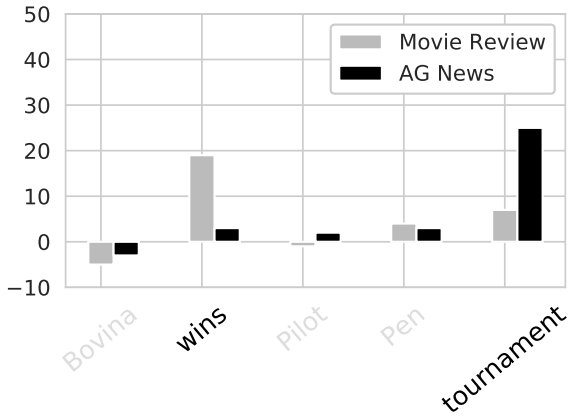

Figure 8: TDS of "Bovina wins Pilot Pen tournament."

our knowledge, we are the first to extend deconvolution to word embedding generation. We empirically demonstrated the efficiency and generalization ability of CDWE by applying it to two representative tasks in NLP: text classification and machine translation. The performances of models can be significant improved by utilizing CDWE as inputs and achieve state-of-the-art results on both tasks. We visualized the linguistic information detected by CDWE and illustrated how CDWE detected and fused context-specific and task-specific information to solve the polysemous-unaware and task-unaware problems via analyzing the TDS scores, which further demonstrated the efficiency of the end-to-end multi-prototype fusion embedding.

In the future, we will explore generalization ability of CDWE by feeding it into other tasks or models, such as language models, parsing, abstract meaning representation, etc. When predicting words in these NLP tasks, the polysemous-unaware problems and task-unaware problems may also influence the results significantly. On the other hand, apart from demonstrating the efficiency of CDWE empirically, we will try figuring out how the convolution-deconvolution structure work and how multiple prototypes fuse the information during the training process theoretically.

7. Acknowledgments

The authors would like to thank the anonymous reviewers for the constructive comments. This work was supported in part by the National Key Research and Development Program of China (No. 2017YFB1400603).

[1] D. Bahdanau, K. Cho, Y. Bengio, 2014. Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473. URL: http://arxiv.org/abs/1409.0473, arXiv:1409.0473.

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

[3] I. Chaturvedi, E. Cambria, R. E. Welsch, F. Herrera, 2018. Distinguishing between facts and opinions for sentiment analysis: Survey and challenges. Information Fusion 44, 65 – 77.

[4] H. Chen, J. Liu, Y. Lv, M. H. Li, M. Liu, Q. Zheng, 2018. Semi-supervised clue fusion for spammer detection in sina weibo. Information Fusion 44, 22 – 32.

[5] T. Chen, R. Xu, Y. He, Y. Xia, X. Wang, 2016. Learning user and product distributed representations using a sequence model for sentiment analysis. IEEE Computational Intelligence Magazine 11, 34–44.

[6] X. Chen, L. Xu, Z. Liu, M. Sun, H.-B. Luan, 2015. Joint learning of character and word embeddings., in: Proceedings of the 24th International Joint Conference on Artificial Intelligence, pp. 1236–1242.

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

[8] A. M. Dai, Q. V. Le, 2015. Semi-supervised sequence learning. CoRR abs/1511.01432. arXiv:1511.01432.

[9] J. Duchi, E. Hazan, Y. Singer, 2011. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research 12, 2121–2159.

[10] L. Fang, Y. Luo, K. Feng, K. Zhao, A. Hu, 2019. Knowledge-enhanced ensemble learning for word embeddings, in: The World Wide Web Conference, WWW, San Francisco, CA, USA, May 13-17, 2019, pp. 427–437.

[11] W. Foland, J. H. Martin, 2017. Abstract meaning representation parsing using LSTM recurrent neural networks, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 463–472.

[12] G. Foster, A. Vaswani, J. Uszkoreit, W. Macherey, L. Kaiser, O. Firat, L. Jones, N. Shazeer, Y. Wu, A. Bapna, M. Johnson, M. Schuster, Z. Chen, M. Hughes, N. Parmar, M. X. Chen, 2018. The best of both worlds: Combining recent advances in neural machine translation, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 76–86.

[13] A. Graves, 2013. Generating sequences with recurrent neural networks. CoRR abs/1308.0850. URL: http://arxiv.org/abs/1308.0850.

[14] A. Graves, S. Ndez, J. Schmidhuber, rgen, 2005. Bidirectional lstm networks for improved phoneme classification and recognition, in: Proceedings of International Conference on Artificial Neural Networks: Formal MODELS and Their Applications - ICANN, pp. 799–804.

[15] I. Gulrajani, K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, A. Courville, 2016. Pixelvae: A latent variable model for natural images. arXiv preprint arXiv:1611.05013 .

[16] R. He, W. S. Lee, H. T. Ng, D. Dahlmeier, 2018. Effective attention modeling for aspect-level sentiment classification, in: Proceedings of the 27th International Conference on Computational Linguistics, COLING, pp. 1121–1131.

[17] B. Huang, K. M. Carley, 2018. Parameterized convolutional neural networks for aspect level sentiment classification, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 1091–1096.

[18] I. Iacobacci, M. T. Pilehvar, R. Navigli, 2015. Sensembed: Learning sense embeddings for word and relational similarity, 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), pp. 95–105.

[19] B. Jiang, Z. Li, H. Chen, A. G. Cohn, 2018. Latent topic text representation learning on statistical manifolds. IEEE Transactions on Neural Networks and Learning Systems 29, 5643–5654.

[20] S. Jimenez, F. A. Gonzalez, A. Gelbukh, G. Duenas, 2019. word2set: Wordnet-based word representation rivaling neural word embedding for lexical similarity and sentiment analysis. IEEE Computational Intelligence Magazine 14, 41–53.

[21] K. Kim, Y. Kim, J. Lee, J. Lee, S. Lee, 2019. From small-scale to large-scale text classification, in: The World Wide Web Conference, WWW, San Francisco, CA, USA, May 13-17, 2019, pp. 853–862.

[22] Y. Kim, 2014. Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, pp. 1746–1751.

[23] S. Lai, L. Xu, K. Liu, J. Zhao, 2015. Recurrent convolutional neural networks for text classification., in: Proceedings of the 29th AAAI Conference on Artificial Intelligence, pp. 2267–2273.

[24] E. Laparra, D. Xu, S. Bethard, 2018. From characters to time intervals: New paradigms for evaluation and neural parsing of time normalizations. TACL 6, 343–356.

[25] S. Li, Z. Zhao, T. Liu, R. Hu, X. Du, 2017. Initializing convolutional filters with semantic features for text classification, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics. pp. 1884–1889.

[26] X. Li, L. Bing, P. Li, W. Lam, Z. Yang, 2018. Aspect term extraction with history attention and selective transformation, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, Stockholm, Sweden., pp. 4194–4200.

[27] K. Liao, L. Lebanoff, F. Liu, 2018. Abstract meaning representation for multi-document summarization, in: Proceedings of the 27th International Conference on Computational Linguistics, COLING, pp. 1178–1190.

[28] L. Liu, F. Ruiz, S. Athey, D. Blei, 2017. Context selection for embedding models, in: Advances in Neural Information Processing Systems, pp. 4819–4828.

[29] L. Liu, M. Utiyama, A. M. Finch, E. Sumita, 2016. Agreement on target-bidirectional neural machine translation, in: Proceedings of The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT, pp. 411–416.

[30] Y. Ma, H. Peng, E. Cambria, 2018. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, pp. 5876–5883.

[31] C. D. Manning, H. Schütze, 1999. Foundations of statistical natural language processing. Journal of Object Technology & Matching Approach for Object-oriented Formal Specifications 26, 37–38.

[32] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, 2013. Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, pp. 3111–3119.

[33] M. Olvera-Lobo, J. Gutiérrez-Artacho, 2015. Question answering track evaluation in trec, CLEF and NTCIR, in: New Contributions in Information Systems and Technologies - Volume 1 [WorldCIST'15, Azores, Portugal]., pp. 13–22.

[34] K. Papineni, S. Roukos, T. Ward, W. Zhu, 2002. Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics ACL, pp. 311–318.

[35] J. Pennington, R. Socher, C. D. Manning, 2014. Glove: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–43.

[36] Y. Pu, W. Yuan, A. Stevens, C. Li, L. Carin, 2016. A deep generative deconvolutional image model, in: Artificial Intelligence and Statistics, pp. 741–750.

[37] S. Ruder, J. Howard, 2018. Universal language model fine-tuning for text classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, pp. 328–339.

[38] C. D. Santos, B. Zadrozny, 2014. Learning character-level representations for part-of-speech tagging, in: Proceedings of the 31st International Conference on Machine Learning (ICML-14), pp. 1818–1826.

[39] D. Shen, R. Pan, J.-T. Sun, J. J. Pan, K. Wu, J. Yin, Q. Yang, 2006. Query enrichment for web-query classification. ACM Transactions on Information Systems (TOIS) 24, 320–352.

[40] K. Shuang, Z. Zhang, H. Guo, J. Loo, 2018. A sentiment information collector-extractor architecture based neural network for sentiment analysis. Information Sciences 467, 549–558.

[41] J. Su, S. Wu, D. Xiong, Y. Lu, X. Han, B. Zhang, 2018. Variational recurrent neural machine translation, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. 5488–5495.

[42] S. Sun, C. Luo, J. Chen, 2017. A review of natural language processing techniques for opinion mining systems. Information Fusion 36, 10 – 25.

[43] K. S. Tai, R. Socher, C. D. Manning, 2015. Improved semantic representations from tree-structured long short-term memory networks, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Association for Computational Linguistics. pp. 1556–1566.

[44] Z. Tu, Z. Lu, Y. Liu, X. Liu, H. Li, 2016. Modeling coverage for neural machine translation, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 76–85.

[45] L. Vanni, M. Ducoffe, C. Aguilar, F. Precioso, D. Mayaffre, 2018. Textual deconvolution saliency (TDS) : a deep tool box for linguistic analysis, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, pp. 548–557.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, 2017. Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 5998–6008.

[47] J. Wang, Z. Wang, D. Zhang, J. Yan, 2017a. Combining knowledge with deep convolutional neural networks for short text classification, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, AAAI Press. pp. 2915–2921.

[48] M. Wang, Z. Lu, H. Li, Q. Liu, 2016. Memory-enhanced decoder for neural machine translation, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 278–286.

[49] M. Wang, Z. Lu, J. Zhou, Q. Liu, 2017b. Deep neural machine translation with linear associative unit, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 136–145.

[50] S. Wang, C. D. Manning, 2012. Baselines and bigrams: Simple, good sentiment and topic classification, in: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2, Association for Computational Linguistics. pp. 90–94.

[51] S. Wang, S. Mazumder, B. Liu, M. Zhou, Y. Chang, 2018. Target-sensitive memory networks for aspect sentiment classification, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL Volume 1: Long Papers, pp. 957–967.

[52] Z. Wang, J. Zhang, J. Feng, Z. Chen, 2014. Knowledge graph and text jointly embedding, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1591–1601.

[53] T. Young, D. Hazarika, S. Poria, E. Cambria, 2018. Recent trends in deep learning based natural language processing [review article]. IEEE Computational Intelligence Magazine 13, 55–75.

[54] M. Yu, M. Dredze, 2014. Improving lexical embeddings with semantic knowledge, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 545–550.

[55] M. D. Zeiler, G. W. Taylor, R. Fergus, 2011. Adaptive deconvolutional networks for mid and high level feature learning, in: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE. pp. 2018–2025.

[56] B. Zhang, D. Xiong, J. Su, H. Duan, M. Zhang, 2016a. Variational neural machine translation, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP, pp. 521–530.

[57] J. Zhang, M. Wang, Q. Liu, J. Zhou, 2017a. Incorporating word reordering knowledge into attention-based neural machine translation, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL, pp. 1524–1534.

[58] R. Zhang, H. Lee, D. R. Radev, 2016b. Dependency sensitive convolutional neural networks for modeling sentences and documents, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics. pp. 1512–1521.

[59] X. Zhang, J. Su, Y. Qin, Y. Liu, R. Ji, H. Wang, 2018. Asynchronous bidirectional decoding for neural machine translation, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. 5698–5705.

[60] X. Zhang, J. Zhao, Y. LeCun, 2015. Character-level convolutional networks for text classification, in: Advances in Neural Information Processing Systems, pp. 649–657.

[61] Y. Zhang, S. Roller, B. C. Wallace, 2016c. Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics. pp. 1522–1527.

[62] Y. Zhang, D. Shen, G. Wang, Z. Gan, R. Henao, L. Carin, 2017b. Deconvolutional paragraph representation learning, in: Advances in Neural Information Processing Systems, pp. 4172–4182.

[63] X. Zheng, J. Feng, Y. Chen, H. Peng, W. Zhang, 2017. Learning context-specific word/character embeddings., in: Proceedings of the 31rt AAAI Conference on Artificial Intelligence, pp. 3393–3399.