Learning Domain-Sensitive and Sentiment-Aware Word Embeddings[*]

Bei Shi[1], Zihao Fu[1], Lidong Bing[2] and Wai Lam[1]
[1]Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong, Hong Kong
[2]Tencent AI Lab, Shenzhen, China
{bshi,zhfu,wlam}@se.cuhk.edu.hk
lyndonbing@tencent.com

Abstract

Word embeddings have been widely used in sentiment classification because of their efficacy for semantic representations of words. Given reviews from different domains, some existing methods for word embeddings exploit sentiment information, but they cannot produce domain-sensitive embeddings. On the other hand, some other existing methods can generate domain-sensitive word embeddings, but they cannot distinguish words with similar contexts but opposite sentiment polarity. We propose a new method for learning domain-sensitive and sentiment-aware embeddings that simultaneously capture the information of sentiment semantics and domain sensitivity of individual words. Our method can automatically determine and produce domain-common embeddings and domain-specific embeddings. The differentiation of domain-common and domain-specific words enables the advantage of data augmentation of common semantics from multiple domains and capture the varied semantics of specific words from different domains at the same time. Experimental results show that our model provides an effective way to learn domain-sensitive and sentiment-aware word embeddings which benefit sentiment classification at both sentence level and lexicon term level.

[*] This work was partially done when Bei Shi was an intern at Tencent AI Lab. This project is substantially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Code: 14203414).

1 Introduction

Sentiment classification aims to predict the sentiment polarity, such as "positive" or "negative", over a piece of review. It has been a long-standing research topic because of its importance for many applications such as social media analysis, e-commerce, and marketing (Liu, 2012; Pang et al., 2008). Deep learning has brought in progress in various NLP tasks, including sentiment classification. Some researchers focus on designing RNN or CNN based models for predicting sentence level (Kim, 2014) or aspect level sentiment (Li et al., 2018; Chen et al., 2017; Wang et al., 2016). These works directly take the word embeddings pre-trained for general purpose as initial word representations and may conduct fine tuning in the training process. Some other researchers look into the problem of learning task-specific word embeddings for sentiment classification aiming at solving some limitations of applying general pre-trained word embeddings. For example, Tang et al. (2014b) develop a neural network model to convey sentiment information in the word embeddings. As a result, the learned embeddings are **sentiment-aware** and able to distinguish words with similar syntactic context but opposite sentiment polarity, such as the words "good" and "bad". In fact, sentiment information can be easily obtained or derived in large scale from some data sources (e.g., the ratings provided by users), which allows reliable learning of such sentiment-aware embeddings.

Apart from these words (e.g. "good" and "bad") with consistent sentiment polarity in different contexts, the polarity of some sentiment words is **domain-sensitive**. For example, the word "lightweight" usually connotes a positive sentiment in the electronics domain since a lightweight device is easier to carry. In contrast, in the movie domain, the word "lightweight" usually connotes a negative opinion describing movies that do not invoke deep thoughts among the audience. This observation motivates the study of learning domain-sensitive word representations (Yang et al., 2017; Bollegala et al., 2015, 2014). They basically learn separate embeddings of the same word for different domains. To bridge the semantics of individual embedding spaces, they select a subset of words that are likely to be domain-insensitive and align the dimensions of their embeddings. However, the sentiment information is not exploited in these methods although they intend to tackle the task of sentiment classification.

In this paper, we aim at learning word embeddings that are both domain-sensitive and sentiment-aware. Our proposed method can jointly model the sentiment semantics and domain specificity of words, expecting the learned embeddings to achieve superior performance for the task of sentiment classification. Specifically, our method can automatically determine and produce domain-common embeddings and domain-specific embeddings. Domain-common embeddings represent the fact that the semantics of a word including its sentiment and meaning in different domains are very similar. For example, the words "good" and "interesting" are usually domain-common and convey consistent semantic meanings and positive sentiments in different domains. Thus, they should have similar embeddings across domains. On the other hand, domain-specific word embeddings represent the fact that the sentiments or meanings across domains are different. For example, the word "lightweight" represents different sentiment polarities in the electronics domain and the movie domain. Moreover, some polysemous words have different meanings in different domains. For example, the term "apple" refers to the famous technology company in the electronics domain or a kind of fruit in the food domain.

Our model exploits the information of sentiment labels and context words to distinguish domain-common and domain-specific words. If a word has similar sentiments and contexts across domains, it indicates that the word has common semantics in these domains, and thus it is treated as domain-common. Otherwise, the word is considered as domain-specific. The learning of domain-common embeddings can allow the advantage of data augmentation of common semantics of multiple domains, and meanwhile, domain-specific embeddings allow us to capture the varied semantics of specific words in different domains. Specifically, for each word in the vocabulary, we design a distribution to depict the probability of the word being domain-common. The inference of the probability distribution is conducted based on the observed sentiments and contexts. As mentioned above, we also exploit the information of sentiment labels for the learning of word embeddings that can distinguish words with similar syntactic context but opposite sentiment polarity.

To demonstrate the advantages of our domain-sensitive and sentiment-aware word embeddings, we conduct experiments on four domains, including books, DVSs, electronics, and kitchen appliances. The experimental results show that our model can outperform the state-of-the-art models on the task of sentence level sentiment classification. Moreover, we conduct lexicon term sentiment classification in two common sentiment lexicon sets to evaluate the effectiveness of our sentiment-aware embeddings learned from multiple domains, and it shows that our model outperforms the state-of-the-art models on most domains.

2 Related Works

Traditional vector space models encode individual words using the one-hot representation, namely, a high-dimensional vector with all zeroes except in one component corresponding to that word (Baeza-Yates et al., 1999). Such representations suffer from the curse of dimensionality, as there are many components in these vectors due to the vocabulary size. Another drawback is that semantic relatedness of words cannot be modeled using such representations. To address these shortcomings, Rumelhart et al. (1988) propose to use distributed word representation instead, called word embeddings. Several techniques for generating such representations have been investigated. For example, Bengio et al. propose a neural network architecture for this purpose (Bengio et al., 2003; Bengio, 2009). Later, Mikolov et al. (2013) propose two methods that are considerably more efficient, namely skip-gram and CBOW. This work has made it possible to learn word embeddings from large data sets, which has led to the current popularity of word embeddings. Word embedding models have been applied to many tasks, such as named entity recognition (Turian et al., 2010), word sense disambiguation (Collobert et al., 2011; Iacobacci et al., 2016; Zhang and Hasan, 2017; Dave et al., 2018), parsing (Roth and Lapata, 2016), and document classification (Tang et al., 2014a,b; Shi et al., 2017).

Sentiment classification has been a long-standing research topic (Liu, 2012; Pang et al., 2008; Chen et al., 2017; Moraes et al., 2013). Given a review, the task aims at predicting the sentiment polarity on the sentence level (Kim, 2014) or the aspect level (Li et al., 2018; Chen et al., 2017). Supervised learning algorithms have been widely used in sentiment classification (Pang et al., 2002). People usually use different expressions of sentiment semantics in different domains. Due to the mismatch between domain-specific words, a sentiment classifier trained in one domain may not work well when it is directly applied to other domains. Thus cross-domain sentiment classification algorithms have been explored (Pan et al., 2010; Li et al., 2009; Glorot et al., 2011). These works usually find common feature spaces across domains and then share learned parameters from the source domain to the target domain. For example, Pan et al. (2010) propose a spectral feature alignment algorithm to align words from different domains into unified clusters. Then the clusters can be used to reduce the gap between words of the two domains, which can be used to train sentiment classifiers in the target domain. Compared with the above works, our model focuses on learning both domain-common and domain-specific embeddings given reviews from all the domains instead of only transferring the common semantics from the source domain to the target domain.

Some researchers have proposed some methods to learn task-specific word embeddings for sentiment classification (Tang et al., 2014a,b). Tang et al. (2014b) propose a model named SSWE to learn sentiment-aware embedding via incorporating sentiment polarity of texts in the loss functions of neural networks. Without the consideration of varied semantics of domain-specific words in different domains, their model cannot learn sentiment-aware embeddings across multiple domains. Some works have been proposed to learn word representations considering multiple domains (Yang et al., 2017; Bach et al., 2016; Bollegala et al., 2015). Most of them learn separate embeddings of the same word for different domains. Then they choose pivot words according to frequency-based statistical measures to bridge the semantics of individual embedding spaces. A regularization formulation enforcing that word representations of pivot words should be similar in different domains is added into the original word embedding framework. For example, Yang et al. (2017) use Sørensen-Dice coefficient (Sørensen, 1948) for detecting pivot words and learn word representations across domains. Even though they evaluate the model via the task of sentiment classification, sentiment information associated with the reviews are not considered in the learned embeddings. Moreover, the selection of pivot words is according to frequency-based statistical measures in the above works. In our model, the domain-common words are jointly determined by sentiment information and context words.

3 Model Description

We propose a new model, named DSE, for learning **D**omain-sensitive and **S**entiment-aware word **E**mbeddings. For presentation clarity, we describe DSE based on two domains. Note that it can be easily extended for more than two domains, and we remark on how to extend near the end of this section.

3.1 Design of Embeddings

We assume that the input consists of text reviews of two domains, namely $\mathcal{D}^p$ and $\mathcal{D}^q$. Each review r in $\mathcal{D}^p$ and $\mathcal{D}^q$ is associated with a sentiment label y which can take on the value of 1 and 0 denoting that the sentiment of the review is positive and negative respectively.

In our DSE model, each word w in the whole vocabulary Λ is associated with a domain-common vector U_w^c and two domain-specific vectors, namely U_w^p specific to the domain p and U_w^q specific to the domain q. The dimension of these vectors is d. The design of U_w^c, U_w^p and U_w^q reflects one characteristic of our model: allowing a word to have different semantics across different domains. The semantic of each word includes not only the semantic meaning but also the sentiment orientation of the word. If the semantic of w is consistent in the domains p and q, we use the vector U_w^c for both domains. Otherwise, w is represented by U_w^p and U_w^q for p and q respectively.

In traditional cross-domain word embedding methods (Yang et al., 2017; Bollegala et al., 2015, 2016), each word is represented by different vectors in different domains without differentiation of domain-common and domain-specific words. In contrast to these methods, for each word w, we use a latent variable z_w to depict its domain commonality. When $z_w = 1$, it means that w is common in both domains. Otherwise, w is specific to the domain p or the domain q.

In the standard skip-gram model (Mikolov et al., 2013), the probability of predicting the context words is only affected by the relatedness with the target words. In our DSE model, predicting the context words also depends on the domain-commonality of the target word, i.e z_w. For example, assume that there are two domains, e.g. the electronics domain and the movie domain. If $z_w = 1$, it indicates a high probability of generating some domain-common words such as "good", "bad" or "satisfied". Otherwise, the domain-specific words are more likely to be generated such as "reliable", "cheap" or "compacts" for the electronics domain. For a word w, we assume that the probability of predicting the context word w_t is formulated as follows:

$$p(w_t|w) = \sum_{k\in\{0,1\}} p(w_t|w, z_w = k)p(z_w = k) \tag{1}$$

If w is a domain-common word without differentiating p and q, the probability of predicting w_t can be defined as:

$$p(w_t|w, z_w = 1) = \frac{\exp(U_w^c \cdot V_{w_t})}{\sum_{w'\in\Lambda} \exp(U_w^c \cdot V_{w'})} \tag{2}$$

where Λ is the whole vocabulary and $V_{w'}$ is the output vector of the word w'.

If w is a domain-specific word, the probability of $p(w_t|w, z_w = 0)$ is specific to the occurrence of w in $\mathcal{D}^p$ or $\mathcal{D}^q$. For individual training instances, the occurrences of w in $\mathcal{D}^p$ or $\mathcal{D}^q$ have been established. Then the probability of $p(w_t|w, z_w = 0)$ can be defined as follows:

$$p(w_t|w, z_w = 0) = \begin{cases} \frac{\exp(U_w^p\cdot V_{w_t})}{\sum_{w'\in\Lambda}\exp(U_w^p\cdot V_{w'})}, \text{if } w \in \mathcal{D}^p \\ \\ \frac{\exp(U_w^q\cdot V_{w_t})}{\sum_{w'\in\Lambda}\exp(U_w^q\cdot V_{w'})}, \text{if } w \in \mathcal{D}^q \end{cases} \tag{3}$$

3.2 Exploiting Sentiment Information

In our DSE model, the prediction of review sentiment depends on not only the text information but also the domain-commonality. For example, the domain-common word "good" has high probability to be positive in different reviews across multiple domains. However, for the word "lightweight", it would be positive in the electronics domain, but negative in the movie domain. We define the polarity y_w of each word w to be consistent with the sentiment label of the review: if we observe that a review is associated with a positive label, the words in the review are associated with a positive label too. Then, the probability of predicting the sentiment for the word w can be defined as:

$$p(y_w|w) = \sum_{k\in\{0,1\}} p(y_w|w, z_w = k)p(z_w = k) \tag{4}$$

If $z_w = 1$, the word w is a domain-common word. The probability $p(y_w = 1|w, z_w = 1)$ can be defined as:

$$p(y_w = 1|w, z_w = 1) = \sigma(U_w^c \cdot \mathbf{s}) \tag{5}$$

where $\sigma(\cdot)$ is the sigmoid function and the vector $\mathbf{s}$ with dimension d represents the boundary of the sentiment. Moreover, we have:

$$p(y_w = 0|w, z_w = 1) = 1 - p(y_w = 1|w, z_w = 1) \tag{6}$$

If w is a domain-specific word, similarly, the probability $p(y_w = 1|w, z_w = 0)$ is defined as:

$$p(y_w = 1|w, z_w = 0) = \begin{cases} \sigma(U_w^p \cdot \mathbf{s}) & \text{if } w \in \mathcal{D}^p \\ \sigma(U_w^q \cdot \mathbf{s}) & \text{if } w \in \mathcal{D}^q \end{cases} \tag{7}$$

3.3 Inference Algorithm

We need an inference method that can learn, given $\mathcal{D}^p$ and $\mathcal{D}^q$, the values of the model parameters, namely, the domain-common embedding U_w^c, and the domain-specific embeddings U_w^p and U_w^q, as well as the domain-commonality distribution $p(z_w)$ for each word w. Our inference method combines the Expectation-Maximization (EM) method with a negative sampling scheme. It is summarized in Algorithm 1. In the E-step, we use the Bayes rule to evaluate the posterior distribution of z_w for each word and derive the objective function. In the M-step, we maximize the objective function with the gradient descent method and

Algorithm 1 EM negative sampling for DSE

```
1:  Initialize $U_w^c$, $U_w^p$, $U_w^q$, $V$, s, $p(z_w)$
2:  for $iter = 1$ to $Max\_iter$ do
3:      for each review $r$ in $\mathcal{D}^p$ and $\mathcal{D}^q$ do
4:          for each word $w$ in $r$ do
5:              Sample negative instances from the distribution P.
6:              Update $p(z_w|w, c_w, y_w)$ by Eq. 11 and Eq. 15 respectively.
7:          end for
8:      end for
9:      Update $p(z_w)$ using Eq. 13
10:     Update $U_w^c$, $U_w^p$, $U_w^q$, $V$, s via Maximizing Eq. 14
11: end for
```

update the corresponding embeddings U_w^c, U_w^p and U_w^q.

With the input of $\mathcal{D}^p$ and $\mathcal{D}^q$, the likelihood function of the whole training set is:

$$\mathcal{L} = \mathcal{L}^p + \mathcal{L}^q \tag{8}$$

where $\mathcal{L}^p$ and $\mathcal{L}^q$ are the likelihood of $\mathcal{D}^p$ and $\mathcal{D}^q$ respectively.

For each review r from $\mathcal{D}^p$, to learn domain-specific and sentiment-aware embeddings, we wish to predict the sentiment label and context words together. Therefore, the likelihood function is defined as follows:

$$\mathcal{L}^p = \sum_{r \in \mathcal{D}^p} \sum_{w \in r} \log p(y_w, c_w|w) \tag{9}$$

where y_w is the sentiment label and c_w is the set of context words of w. For the simplification of the model, we assume that the sentiment label y_w and the context words c_w of the word w are conditionally dependent. Then the likelihood $\mathcal{L}^p$ can be rewritten as:

$$\mathcal{L}^p = \sum_{r \in \mathcal{D}^p} \sum_{w \in r} \sum_{w_t \in c_w} \log p(w_t|w) + \sum_{r \in \mathcal{D}^p} \sum_{w \in r} \log p(y_w|w) \tag{10}$$

where $p(w_t|w)$ and $p(y_w|w)$ are defined in Eq. 1 and Eq. 4 respectively. The likelihood of the reviews from $\mathcal{D}^q$, i.e $\mathcal{L}^q$, is defined similarly.

For each word w in the review r, in the E-step, the posterior probability of z_w given c_w and y_w is:

$$p(z_w = k|w, c_w, y_w) = \frac{p(z_w = k) p(y_w|w, z_w = k) \prod_{w_t \in c_w} p(w_t|w, z_w = k)}{\sum_{k' \in \{0,1\}} p(z_w = k') p(y_w|w, z_w = k') \prod_{w_t \in c_w} p(w_t|w, z_w = k')} \tag{11}$$

In the M-step, given the posterior distribution of z_w in Eq. 11, the goal is to maxmize the following Q function:

$$\begin{aligned}\mathbf{Q} &= \sum_{r \in \{\mathcal{D}^p, \mathcal{D}^q\}} \sum_{w \in r} \sum_{z_w} p(z_w|w, y_w, w_{t+j}) \\ &\quad \times \log(p(z_w) p(c_w, y|z, w_t)) \\ &= \sum_{r \in \{\mathcal{D}^p, \mathcal{D}^q\}} \sum_{w \in r} \sum_{z_w} p(z_w|w, y_w, c_w) \\ &\quad [\log p(z_w) + \log(y_w|z, w) + \\ &\quad \sum_{w_t \in c_w} \log p(w_t|z_w, w)]\end{aligned} \tag{12}$$

Using the Lagrange multiplier, we can obtain the update rule of $p(z_w)$, satisfying the normalization constraints that $\sum_{z_w \in 0,1} p(z_w) = 1$ for each word w:

$$p(z_w) = \frac{\sum_{r \in \{\mathcal{D}^p, \mathcal{D}^q\}} \sum_{w \in r} p(z_w|w, y_w, c_w)}{\sum_{r \in \{\mathcal{D}^p, \mathcal{D}^q\}} n(w, r)} \tag{13}$$

where $n(w, r)$ is the number of occurrence of the word w in the review r.

To obtain U_w^c, U_w^p and U_w^q, we collect the related items in Eq. 12 as follows:

$$\begin{aligned}\mathbf{Q_U} &= \sum_{r \in \{\mathcal{D}^p, \mathcal{D}^q\}} \sum_{w \in r} \sum_{z_w} p(z_w|w, y_w, w_{t+j}) \\ &\quad [\log(y_w|z_w, w) + \sum_{w_t \in c_w} \log p(w_t|z_w, w)]\end{aligned} \tag{14}$$

Note that computing the value $p(w_t|w, z_w)$ based on Eq. 2 and Eq. 3 is not feasible in practice, given that the computation cost is proportional to the size of Λ. However, similar to the skip-gram model, we can rely on negative sampling to address this issue. Therefore we estimate the probability of predicting the context word $p(w_t|w, z_w = 1)$ as follows:

$$\begin{aligned}\log p(w_t|w, z_w = 1) &\propto \log \sigma(U_w^c \cdot V_{w_t}) \\ &+ \sum_{i=1}^{n} \mathbf{E}_{w_i \sim P}[\log \sigma(-U_w^c \cdot V_{w_i})]\end{aligned} \tag{15}$$

where w_i is a negative instance which is sampled from the word distribution $P(.)$. Mikolov et al. (2013) have investigated many choices for $P(w)$ and found that the best $P(w)$ is equal to the unigram distribution $\mathit{Unigram}(w)$ raised to the $3/4rd$ power. We adopt the same setting. The probability $p(w_t|w, z_w = 0)$ in Eq. 3 can be approximated in a similar manner.

After the substitution of $p(w_t|w, z_w)$, we use the Stochastic Gradient Descent method to maximize Eq. 14, and obtain the update of U_w^c, U_w^p and U_w^q.

3.4 More Discussions

In our model, for simplifying the inference algorithm and saving the computational cost, we assume that the target word w_t in the context and the sentiment label y_w of the word w are conditionally independent. Such technique has also been used in other popular models such as the bi-gram language model. Otherwise, we need to consider the term $p(w_t|w, y_w)$, which complicates the inference algorithm.

We define the formulation of the term $p(w_t|w, z)$ to be similar to the original skip-gram model instead of the CBOW model. The CBOW model averages the context words to predict the target word. The skip-gram model uses pairwise training examples which are much easier to integrate with sentiment information.

Note that our model can be easily extended to more than two domains. Similarly, we use a domain-specific vector for each word in each domain and each word is also associated with a domain-common vector. We just need to extend the probability distribution of z_w from Bernoulli distribution to Multinomial distribution according to the number of domains.

4 Experiment

4.1 Experimental Setup

We conducted experiments on the Amazon product reviews collected by Blitzer et al. (2007). We use four product categories: books (**B**), DVDs (**D**), electronic items (**E**), and kitchen appliances (**K**). A category corresponds to a domain. For each domain, there are 17,457 unlabeled reviews on average associated with rating scores from 1.0 to 5.0 for each domain. We use unlabeled reviews with rating score higher than 3.0 as positive reviews and unlabeled reviews with rating score lower than 3.0 as negative reviews for embedding learning. We first remove reviews whose length is less than 5 words. We also remove punctuations and the stop words. We also stem each word to its root form using Porter Stemmer (Porter, 1980). Note that this review data is used for embedding learning, and the learned embeddings are used as feature vectors of words to conduct the experiments in the later two subsections.

Given the reviews from two domains, namely, $\mathcal{D}^p$ and $\mathcal{D}^q$, we compare our results with the following baselines and state-of-the-art methods:

SSWE The SSWE model[1] proposed by Tang et al. (2014b) can learn sentiment-aware word embeddings from tweets. We employ this model on the combined reviews from $\mathcal{D}^p$ and $\mathcal{D}^q$ and then obtain the embeddings.

Yang's Work Yang et al. (2017) have proposed a method[2] to learn domain-sensitive word embeddings. They choose pivot words and add a regularization item into the original skip-gram objective function enforcing that word representations of pivot words for the source and target domains should be similar. The method trains the embeddings of the source domain first and then fixes the learned embedding to train the embedding of the target domain. Therefore, the learned embedding of the target domain benefits from the source domain. We denote the method as Yang in short.

EmbeddingAll We learn word embeddings from the combined unlabeled review data of $\mathcal{D}^p$ and $\mathcal{D}^q$ using the skip-gram method (Mikolov et al., 2013).

EmbeddingCat We learn word embeddings from the unlabeled reviews of $\mathcal{D}^p$ and $\mathcal{D}^q$ respectively. To represent a word for review sentiment classification, we concatenate its learned word embeddings from the two domains.

EmbeddingP and EmbeddingQ In EmbeddingP, we use the original skip-gram method (Mikolov et al., 2013) to learn word

[1]We use the implementation from `https://github.com/attardi/deepnl/wiki/Sentiment-Specific-Word-Embeddings`.

[2]We use the implementation from `http://statnlp.org/research/lr/`.

	B & D		B & E		B & K		D & E		D & K		E & K	
	Acc.	F1	*Acc.*	F1	*Acc.*	F1	*Acc.*	F1	*Acc.*	F1	*Acc.*	F1
BOW	0.680	0.653	0.738	0.720	0.734	0.725	0.705	0.685	0.706	0.689	0.739	0.715
EmbeddingP	0.753	0.740	0.752	0.745	0.742	0.741	0.740	0.746	0.707	0.702	0.761	0.760
EmbeddingQ	0.736	0.732	0.697	0.697	0.706	0.701	0.762	0.759	0.758	0.759	0.783	0.780
EmbeddingCat	0.769	0.731	0.768	0.763	0.763	0.763	0.787	0.773	0.770	0.770	0.807	0.803
EmbeddingAll	0.769	0.759	0.765	0.740	0.775	0.767	0.783	0.779	0.779	0.776	0.819	0.815
Yang	0.767	0.752	0.775	0.766	0.760	0.755	0.791	0.785	0.762	0.760	0.805	0.804
SSWE	0.783	0.772	0.791	0.780	**0.801**	0.792	0.825	0.815	0.795	0.790	0.835	0.824
DSE$_c$	0.773	0.750	0.783	0.781	0.775	0.773	0.797	0.792	0.784	0.776	0.806	0.800
DSE$_w$	**0.794**$^{\dagger\natural}$	**0.793**$^{\dagger\natural}$	**0.806**$^{\dagger\natural}$	**0.802**$^{\dagger\natural}$	0.797†	**0.793**†	**0.843**$^{\dagger\natural}$	**0.832**$^{\dagger\natural}$	**0.829**$^{\dagger\natural}$	**0.827**$^{\dagger\natural}$	**0.856**$^{\dagger\natural}$	**0.853**$^{\dagger\natural}$

Table 1: Results of review sentiment classification. The markers † and $^{\natural}$ refer to p-value < 0.05 when comparing with Yang and SSWE respectively.

embeddings only from the unlabeled reviews of $\mathcal{D}^p$. Similarly, we only adopt the unlabeled reviews from $\mathcal{D}^q$ to learn embeddings in EmbeddingQ.

BOW We use the traditional bag of words model to represent each review in the training data.

For our DSE model, we have two variants to represent each word. The first variant DSE$_c$ represents each word via concatenating the domain-common vector and the domain-specific vector. The second variant DSE$_w$ concatenates domain-common word embeddings and domain-specific word embeddings by considering the domain-commonality distribution $p(z_w)$. For individual review instances, the occurrences of w in $\mathcal{D}_p$ or $\mathcal{D}_q$ have been established. The representation of w is specific to the occurrence of w in $\mathcal{D}_p$ or $\mathcal{D}_q$. Specifically, each word w can be represented as follows:

$$U_w = \begin{cases} \text{if } w \in \mathcal{D}^p \\ \quad U_w^c \times p(z_w) \oplus U_w^p \times (1.0 - p(z_w)) \\ \text{if } w \in \mathcal{D}^q \\ \quad U_w^c \times p(z_w) \oplus U_w^q \times (1.0 - p(z_w)) \end{cases} \tag{16}$$

where $\oplus$ denotes the concatenation operator.

For all word embedding methods, we set the dimension to 200. For the skip-gram based methods, we sample 5 negative instances and the size of the windows for each target word is 3. For our DSE model, the number of iterations for the whole reviews is 100 and the learning rate is set to 1.0.

4.2 Review Sentiment Classification

For the task of review sentiment classification, we use 1000 positive and 1000 negative sentiment reviews labeled by Blitzer et al. (2007) for each domain to conduct experiments. We randomly select 800 positive and 800 negative labeled reviews from each domain as training data, and the remaining 200 positive and 200 negative labeled reviews as testing data. We use the SVM classifier (Fan et al., 2008) with linear kernel to train on the training reviews for each domain, with each review represented as the average vector of its word embeddings.

We use two metrics to evaluate the performance of sentiment classification. One is the standard accuracy metric. The other one is Macro-F1, which is the average of F1 scores for both positive and negative reviews.

We conduct multiple trials by selecting every possible two domains from books (**B**), DVDs (**D**), electronic items (**E**) and kitchen appliances (**K**). We use the average of the results of each two domains. The experimental results are shown in Table 1.

From Table 1, we can see that compared with other baseline methods, our DSE$_w$ model can achieve the best performance of sentiment classification across most combinations of the four domains. Our statistical t-tests for most of the combinations of domains show that the improvement of our DSE$_w$ model over Yang and SSWE is statistically significant respectively (p-value < 0.05) at 95% confidence level. It shows that our method can capture the domain-commonality and sentiment information at the same time.

Even though both of the SSWE model and our DSE model can learn sentiment-aware word embeddings, our DSE$_w$ model can outperform SSWE. It demonstrates that compared with general sentiment-aware embeddings, our learned domain-common and domain-specific word em-

	B & D		B & E		B & K		D & E		D & K		E & K	
	HL	MPQA	HL	MPQA	HL	MPQA	HL	MPQA	HL	MPQA	HL	MPQA
EmbeddingP	0.740	0.733	0.742	0.734	0.747	0.735	0.744	0.701	0.745	0.709	0.628	0.574
EmbeddingQ	0.743	0.701	0.627	0.573	0.464	0.453	0.621	0.577	0.462	0.450	0.465	0.453
EmbeddingCat	0.780	0.772	0.773	0.756	0.772	0.751	0.744	0.728	0.755	0.702	0.683	0.639
EmbeddingAll	0.777	0.769	0.773	0.730	0.762	0.760	0.712	0.707	0.749	0.724	0.670	0.658
Yang	0.780	0.775	0.789	0.762	0.781	0.770	0.762	0.736	0.756	0.713	0.634	0.614
SSWE	**0.816**	**0.801**	0.831	0.817	0.822	**0.808**	**0.826**	0.785	0.784	0.772	0.707	0.659
DSE	0.802	0.788	**0.833**	**0.828**	**0.832**	0.799	0.804	**0.797**	**0.796**	**0.786**	**0.725**	**0.683**

Table 2: Results of lexicon term sentiment classification.

beddings can capture semantic variations of words across multiple domains.

Compared with the method of Yang which learns cross-domain embeddings, our DSE$_w$ model can achieve better performance. It is because we exploit sentiment information to distinguish domain-common and domain-specific words during the embedding learning process. The sentiment information can also help the model distinguish the words which have similar contexts but different sentiments.

Compared with EmbeddingP and EmbeddingQ, the methods of EmbeddingAll and EmbeddingCat can achieve better performance. The reason is that the data augmentation from other domains helps sentiment classification in the original domain. Our DSE model also benefits from such kind of data augmentation with the use of reviews from $\mathcal{D}^p$ and $\mathcal{D}^q$.

We observe that our DSE$_w$ variant performs better than the variant of DSE$_c$. Compared with DSE$_c$, our DSE$_w$ variant adds the item of $p(z_w)$ as the weight to combine domain-common embeddings and domain-specific embeddings. It shows that the domain-commonality distribution in our DSE model, i.e $p(w_z)$, can effectively model the domain-sensitive information of each word and help review sentiment classification.

4.3 Lexicon Term Sentiment Classification

To further evaluate the quality of the sentiment semantics of the learned word embeddings, we also conduct lexicon term sentiment classification on two popular sentiment lexicons, namely **HL** (Hu and Liu, 2004) and **MPQA** (Wilson et al., 2005). The words with neutral sentiment and phrases are removed. The statistics of **HL** and **MPQA** are shown in Table 3.

We conduct multiple trials by selecting every possible two domains from books (**B**), DVDs (**D**), electronics items (**E**) and kitchen appliances (**K**).

Lexicon	Positive	Negative	Total
HL	1,331	2,647	3,978
MPQA	1,932	2,817	3,075

Table 3: Statistics of the sentiment lexicons.

For each trial, we learn the word embeddings. For our DSE model, we only use the domain-common part to represent each word because the lexicons are usually not associated with a particular domain. For each lexicon, we select 80% to train the SVM classifier with linear kernel and the remaining 20% to test the performance. The learned embedding is treated as the feature vector for the lexicon term. We conduct 5-fold cross validation on all the lexicons. The evaluation metric is Macro-F1 of positive and negative lexicons.

Table 2 shows the experimental results of lexicon term sentiment classification. Our DSE method can achieve competitive performance among all the methods. Compared with SSWE, our DSE is still competitive because both of them consider the sentiment information in the embeddings. Our DSE model outperforms other methods which do not consider sentiments such as Yang, EmbeddingCat and EmbeddingAll. Note that the advantage of domain-sensitive embeddings would be insufficient for this task because the sentiment lexicons are not domain-specific.

5 Case Study

Table 4 shows the probabilities of "lightweight", "die", "mysterious", and "great" to be domain-common for different domain combinations. For "lightweight", its domain-common probability for the books domain and the DVDs domain ("B & D") is quite high, i.e. $p(z = 1) = 0.999$, and the review examples in the last column show that the word "lightweight" expresses the meaning of lacking depth of content in books or movies. Note that most reviews of DVDs are about movies.

Term	Domain	$p(z = 1)$	Sample Reviews
"lightweight"	B & D	0.999	- I find Seth Godin's books incredibly **lightweight**. There is really nothing of any substance here.(B) - I love the fact that it's small and **lightweight** and fits into a tiny pocket on my camera case so I never lose track of it.(E) - These are not "**lightweight**" actors. (D) - This vacuum does a pretty good job. It is **lightweight** and easy to use.(K)
	B & E	0.404	
	B & K	0.241	
	D & E	0.380	
	D & K	0.013	
	E & K	0.696	
"die"	B & E	0.435	- I'm glad Brando lived long enough to get old and fat, and that he didn't **die** tragically young like Marilyn, JFK, or Jimi Hendrix.(B) - Like many others here, my CD-changer **died** after a couple of weeks and it wouldn't read any CD.(E) - I had this toaster for under 3 years when I came home one day and it smoked and **died**. (K)
	B & K	0.492	
	E & K	0.712	
"mysterious"	B & E	0.297	- This novel really does cover the gamut: stunning twists, genuine love, beautiful settings, desire for riches, **mysterious** murders, detective investigations, false accusations, and self vindication.(B) - Caller ID functionality for Vonage **mysteriously** stopped working even though this phone's REN is rated at 0.1b. (E)
"great"	B & D	0.760	- This is a **great** book for anyone learning how to handle dogs.(B) - This is a **great** product, and you can get it, along with any other products on Amazon up to $500 Free!(E) - I grew up with drag racing in the 50s, 60s & 70s and this film gives a **great** view of what it was like.(D) - This is a **great** mixer its a little loud but worth it for the power you get.(K)
	B & E	0.603	
	B & K	0.628	
	D & E	0.804	
	D & K	0.582	
	E & K	0.805	

Table 4: Learned domain-commonality for some words. $p(z = 1)$ denotes the probability that the word is domain-common. The letter in parentheses indicates the domain of the review.

In the electronics domain and the kitchen appliances domain ("E & K"), "lightweight" means light material or weighing less than average, thus the domain-common probability for these two domains is also high, 0.696. In contrast, for the other combinations, the probability of "lightweight" to be domain-common is much smaller, which indicates that the meaning of "lightweight" varies. Similarly, "die" in the domains of electronics and kitchen appliances ("E & K") means that something does not work any more, thus, we have $p(z = 1) = 0.712$. While for the books domain, it conveys meaning that somebody passed away in some stories. The word "mysterious" conveys a positive sentiment in the books domain, indicating how wonderful a story is, but it conveys a negative sentiment in the electronics domain typically describing that a product breaks down unpredictably. Thus, its domain-common probability is small. The last example is the word "great", and it usually has positive sentiment in all domains, thus has large values of $p(z = 1)$ for all domain combinations.

6 Conclusions

We propose a new method of learning domain-sensitive and sentiment-aware word embeddings. Compared with existing sentiment-aware embeddings, our model can distinguish domain-common and domain-specific words with the consideration of varied semantics across multiple domains. Compared with existing domain-sensitive methods, our model detects domain-common words according to not only similar context words but also sentiment information. Moreover, our learned embeddings considering sentiment information can distinguish words with similar syntactic context but opposite sentiment polarity. We have conducted experiments on two downstream sentiment classification tasks, namely review sentiment classification and lexicon term sentiment classification. The experimental results demonstrate the advantages of our approach.

References

Ngo Xuan Bach, Vu Thanh Hai, and Tu Minh Phuong. 2016. Cross-domain sentiment classification with word embeddings and canonical correlation analysis. In *Proceedings of the Seventh Symposium on Information and Communication Technology*. ACM, pages 159–166.

Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. 1999. *Modern Information Retrieval*, volume 463. ACM press New York.

Yoshua Bengio. 2009. Learning deep architectures for ai. *Foundations and Trends in Machine Learning* 2(1):1–127.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3:1137–1155.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. pages 440–447.

Danushka Bollegala, Takanori Maehara, and Ken-ichi Kawarabayashi. 2015. Unsupervised cross-domain word representation learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*. pages 730–740.

Danushka Bollegala, Tingting Mu, and John Yannis Goulermas. 2016. Cross-domain sentiment classification using sentiment sensitive embeddings. *IEEE Transactions on Knowledge and Data Engineering* 28(2):398–410.

Danushka Bollegala, David Weir, and John Carroll. 2014. Learning to predict distributions of words across domains. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. pages 613–623.

Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pages 452–461.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Vachik Dave, Baichuan Zhang, Pin-Yu Chen, and Mohammad Al Hasan. 2018. Neural-brane: Neural bayesian personalized ranking for attributed network embedding. *arXiv preprint arXiv:1804.08774* .

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research* 9:1871–1874.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning*. pages 513–520.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pages 168–177.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for word sense disambiguation: An evaluation study. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. pages 897–907.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. pages 1746–1751.

Tao Li, Vikas Sindhwani, Chris Ding, and Yi Zhang. 2009. Knowledge transformation for cross-domain sentiment classification. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 716–717.

Xin Li, Lidong Bing, Wai Lam, and Bei Shi. 2018. Transformation networks for target-oriented sentiment classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies* 5(1):1–167.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*. pages 3111–3119.

Rodrigo Moraes, JoãO Francisco Valiati, and Wilson P GaviãO Neto. 2013. Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications* 40(2):621–633.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web*. pages 751–760.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*. pages 79–86.

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2(1–2):1–135.

Martin F Porter. 1980. An algorithm for suffix stripping. *Program* 14(3):130–137.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. pages 1192–1202.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5(3):1.

Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. 2017. Jointly learning word embeddings and latent topics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 375–384.

Thorvald Sørensen. 1948. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biologiske Skrifter* 5:1–34.

Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. 2014a. Building large-scale twitter-specific sentiment lexicon: A representation learning approach. In *Proceedings of the 25th International Conference on Computational Linguistics*. pages 172–182.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014b. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. volume 1, pages 1555–1565.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. pages 384–394.

Yequan Wang, Minlie Huang, Li Zhao, et al. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. pages 606–615.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing*. pages 347–354.

Wei Yang, Wei Lu, and Vincent Zheng. 2017. A simple regularization-based algorithm for learning cross-domain word embeddings. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pages 2898–2904.

Baichuan Zhang and Mohammad Al Hasan. 2017. Name disambiguation in anonymized graphs using network embedding. In *Proceedings of the 26th ACM International on Conference on Information and Knowledge Management*. pages 1239–1248.