Knowledge-Based Systems 155 (2018) 1–10

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Domain attention model for multi-domain sentiment classification

Zhigang Yuan*[,a], Sixing Wu[a], Fangzhao Wu[b], Junxin Liu[a], Yongfeng Huang[a]

[a] Department of Electronic Engineering, Tsinghua University, Beijing, China
[b] Microsoft Research Asia, China

ARTICLE INFO

Keywords:
Sentiment analysis
Multi-domain
Attention mechanism

ABSTRACT

Sentiment classification is widely known as a domain-dependent problem. In order to learn an accurate domain-specific sentiment classifier, a large number of labeled samples are needed, which are expensive and time-consuming to annotate. Multi-domain sentiment analysis based on multi-task learning can leverage labeled samples in each single domain, which can alleviate the need for large amount of labeled data in all domains. In this paper, we propose a domain attention model for multi-domain sentiment analysis. In our approach, the domain representation is used as attention to select the most domain-related features in each domain. The domain representation is obtained through an auxiliary domain classification task, which works as domain regularizer. In this way, both shared and domain-specific features for sentiment classification are extracted simultaneously. In contrast with existing multi-domain sentiment classification methods, our approach can extract the most discriminative features from a shared hidden layer in a more compact way. Experimental results on two multi-domain sentiment datasets validate the effectiveness of our approach.

1. Introduction

Sentiment classification is a key problem in sentiment analysis and a hot topic in research area [1]. The goal of sentiment classification is to classify a piece of text into different sentiment categories, such as positive, negative or neutral [2,3]. Neural network based models have achieved inspiring results on most natural language processing tasks including sentiment classification. However, sentiment classification is widely known as a highly domain-dependent problem. This is because in different domains users may use different expressions to express their sentiments. Even the same expression may convey different sentiments in different domains. For example, in *Electronics* domain, the word "*easy*" in "*the phone is easy to use*" is a positive word. However in *DVD* domain, "*easy*" may convey negative sentiment, such as "*the plot is easy to guess*". As a result, the sentiment classifier trained in one domain often performs unsatisfactorily in another domain.

A natural way to solve this problem is to train a unique classifier for each domain [2,4]. For example, Pang et al. [2] used machine learning based methods to classify the sentiments of movie reviews. However, this method requires a large number of labeled samples in each domain, which is difficult to obtain because manual annotation is very costly and time-consuming. Without enough labeled samples, it is quite challenging to learn an accurate classifier for each domain.

In different domains, users may use some shared expressions (such as "*good*" and "*bad*") as well as domain-specific expressions to express their sentiments. As a result, training sentiment classifiers for different domains can be seen as related tasks and many researchers use multi-task learning methods to solve this problem [5,6]. Multi-domain sentiment classification based on multi-task learning can use samples from different domains to improve generalization ability of classification model, which can alleviate the need for large amount of labeled data in all domains. The key difference between different multi-task learning methods is how they model task relatedness [7]. The effectiveness of these methods are constrained by the assumptions they made, but some of them may not hold in sentiment classification scenarios such as the shared sparse space assumption in [8]. This is because different words may be used to express sentiments in different domains, and the same words may convey opposite sentiments in different domains. Some studies extend this idea by seeking both shared and domain-specific features. In [5,6], Wu et al. decomposed the sentiment classifier of each domain into a general one and a domain-specific one. They then used various lexical resources to guide the training of these two kinds of classifiers. Liu et al. [9] used adversarial training to divide the task-specific and shared space in a more precise way, rather than roughly sharing parameters. Dragoni et al. [10] combined results from domain classification and sentiment classification to get the final decision. However, most of these multi-task learning methods model tasks at layer-level or classifier-level, which would cause the network difficult

* Corresponding author.
E-mail addresses: yuanzg14@mails.tsinghua.edu.cn (Z. Yuan), wu-sx15@mails.tsinghua.edu.cn (S. Wu), fangzwu@microsoft.com (F. Wu), ljx16@mails.tsinghua.edu.cn (J. Liu), yfhuang@tsinghua.edu.cn (Y. Huang).

https://doi.org/10.1016/j.knosys.2018.05.004
Received 27 November 2017; Received in revised form 2 May 2018; Accepted 5 May 2018
Available online 07 May 2018
0950-7051/ © 2018 Elsevier B.V. All rights reserved.

to train as the number of domains increase.

Attention mechanism has shown great success in neural models for many NLP tasks, such as machine translation [11], semantic entailment [12] and aspect-level sentiment analysis [13]. Attention mechanism allows the network to select the most related features for desired tasks. Besides, machine learning methods would achieve better performance if the data distribution is known in advance [14]. Similarly, in multi-domain sentiment classification scenarios, the sentiment classifiers can benefit from the data distribution knowledge in each domain [5,6]. Since whether a word is positive or negative depends heavily on respective domains, the sentiment classifier can benefit from the domain knowledge.

Motivated by these observations, we propose a domain attention model for multi-domain sentiment classification. In this model, different domains share the same feature extraction layer to select the most sentiment-discriminative features. A representation encoded with domain knowledge is then used as attention to select the most domain-related features. These domain-related features include common features across all domains as well as domain-specific features for every single domain. To obtain this domain representation, we utilize domain classification as an auxiliary task. In this way, both domain-shared and domain-specific features can be selected automatically without extra layers for each task. The high level illustration of this model is shown in Fig. 1. In Fig. 1, the domain extraction part learns a good domain representation through an auxiliary domain classification task. The domain representation is then send into a shared hidden layer to trigger an attention process. In this way, the most domain-related features for each domain are automatically selected to predict sentiment labels.

To evaluate the performance of our model, we conduct experiments on two multi-domain sentiment datasets. Experimental results demonstrate the effectiveness of the proposed approach. By using domain representation as attention, features desired for each task can be extracted effectively.

Our contributions are listed as follows:

- We propose a multi-task learning model using attention mechanism for multi-domain sentiment classification task. Different from most existing neural network based multi-task learning methods which model related tasks at layer-level, our proposed attention based model can model tasks at feature-level. Instead of designing separate layers for each task, our model can select desired features for each task from a shared hidden layer in a more compact way.
- We incorporate domain information into multi-domain sentiment classification model. In our model, the domain representation obtained from an auxiliary domain classification task is used as attention to select the most domain-related features for each domain.
- Experimental results show that our approach outperforms all baseline methods. Our approach can effectively improve multi-domain sentiment classification performance.

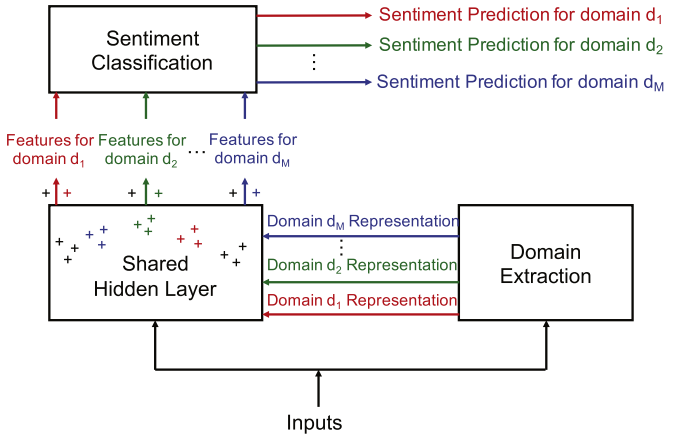

Fig. 1. High level illustration of the Domain Attention Model. The "+" signs in shared hidden layer represent extracted features for sentiment classification. Black "+" signs denote shared features across all domains, while the colored signs denote domain-specific features in each domain respectively. Through domain attention process, both domain-shared and domain-specific features can be selected.

The rest of the paper is organized as follows. In Section 2 we introduce several representative related works. In Section 3 we present our domain attention model in detail. In Section 4 we give the experimental results and analysis. In Section 5 we conclude our work.

2. Related work

In this section, we introduce several related works on multi-domain sentiment classification and attention mechanism.

2.1. Multi-domain sentiment classification

Sentiment classification is widely known as a highly domain-dependent problem [5,6,15–21]. Following the terms widely used in the literature [15], we call **domain** a set of documents about similar topics, e.g., a set of reviews about similar products like books, movies or similar companies like Microsoft and Google, etc. Since users may use different expressions to express sentiments in different domains, classifiers trained in one domain may not perform well when tested in another domain. A natural way to solve this problem is to train an accurate sentiment classifier using enough labeled samples in each domain. For example, Pang et al. [2] used machine learning based methods to classify the sentiments of movie reviews. Chen et al. [4] explored political standpoints using opinion scoring model. However, it is very expensive and time-consuming to manually label enough samples for every domain, which severely limits the application of these models.

To tackle this problem, many researchers studied domain adaption methods based on transfer learning framework [15,16]. The assumption of domain adaptation is that labeled data is sufficient in source domain, while scarce or nonexistent in target domain. The goal of domain adaption is trying to reduce the performance degradation of sentiment classifier during adaptation. For example, Blizter et al. [15] proposed a structural correspondence learning (SCL) method to find the correspondence between features from different domains via pivot features. Pan et al. [16] proposed a spectral feature alignment (SFA) method to align domain-specific words from different domains into unified clusters. Glorot et al. [18] and Chen et al. [19] used neural autoencoder frameworks to learn transferable features. Li et al. [22] used adversarial network to extract domain-indiscriminative while sentiment-discriminative features. They further used a hierarchical attention network to simultaneously capture pivots and non-pivots during adaption [23]. Different from these studies, we assume that the labeled data in each domain is insufficient to train an accurate single domain sentiment classifier. Instead, we extract shared and domain-specific features using limited labeled samples from multiple domains to train a sentiment classifier that has a better performance than single domain classifiers.

In different domains, users may use some shared expressions as well as domain-specific expressions to express their sentiments. As a result, training sentiment classifiers for different domains can be seen as related tasks. Multi-domain sentiment analysis can be seen as a specific application of multi-task learning [24]. In multi-domain sentiment classification scenarios, classifiers of different domains are learned at the same time, during which commonalities and differences among these domains are exploited simultaneously. Roughly speaking, most of these methods fall into two categories: based on label propagation and based on neural network models. For different multi-task learning methods using label propagation, the key part is how they model task relatedness, or domain relatedness in sentiment analysis context. For

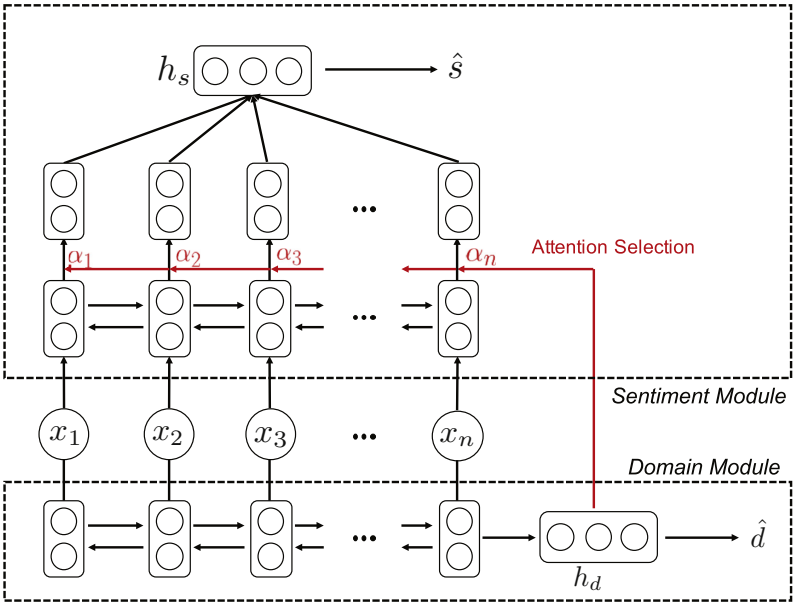

Fig. 2. Domain Attention Model (DAM) for multi-domain sentiment classification.

example, Evgeniou and Pontil [25] proposed a regularized multi-task learning model. They modeled the relation between tasks in terms of a kernel function that uses a task-coupling parameter. Liu et al. [8] proposed a multi-task feature learning framework in which different tasks share similar sparsity patterns. For these methods, the effectiveness depends heavily on the domain relatedness assumptions, which may not apply to various scenarios. Due to the powerful ability of distributed representations [26–28], neural network based multi-task learning models have been proven effective for many NLP tasks. Neural models are better at learning abstract representations for different domains. For example, Collobert et al. [29,30] proposed a unified architecture for natural language processing. In this model, different tasks share the same word embeddings. However, these tasks work independently at higher level. Thus, the domain relatedness is also not fully exploited. Liu et al. [31,32] proposed a multi-task RNN to model text with task-specific and shared layers. In this model, they augmented neural model with an external memory, which is shared by several tasks. Liu et al. [9] further used adversarial scheme to better separate shared and private feature spaces. These models require separate layers for each domain, which may suffer from model complexity as the number of domains increases. Dragoni et al. [10] proposed a neural word embeddings approach to combine the sentiment predictions from each domain. In this model, they also used domain classification layer. However, they only used the result of domain classification as weighting coefficients to combine domain-specific predictions, which cannot model the complex relatedness of related domains. Unlike these methods, in this paper we propose a multi-domain sentiment classification model using attention mechanism. In this model, different domains share the same hidden layer. Both domain-shared and domain-specific features are selected simultaneously using domain representation as attention. Instead of designing separate layers for each task, our method models related domains at feature-level. We use domain representation as attention to extract desired features for each domain from a shared hidden layer.

There also exist some studies based on classifiers combination [20,21]. In these methods, classifiers trained in each domain are combined using different combination schemes. These methods can be seen as special cases of classifier ensemble, which integrates domain information at classification stage. In contrast, our method encodes domain relatedness at learning stage and higher semantic level.

2.2. Attention mechanism

Attention mechanism is first used for machine translation [11] in an encoder-decoder framework. The key idea of attention is to allow the network revisit all parts of a source sentence for an output decision, instead of trying to encode all information of a source sentence into a fixed-length vector.

Successive studies on attention mechanism include image caption [33], semantic entailment [12], aspect-level sentiment analysis [13], etc. For sentiment analysis, Yang et al. [34] used a hierarchical attention network to form words representations into sentences and then into documents. This proposed architecture outperforms previous methods by a substantial margin.

Although attention mechanism has proved to be effective in finding the most discriminative features, it remains unclear how it can be used in multi-domain scenarios. Kumar et al. [35] proposed a dynamic memory networks for natural language processing, which has a similar working procedure to ours. In their work, they used the representation of a fixed query as attention to select the most related features for desired task. However, it is still limited to single domain application. In our work, we utilize an auxiliary task for domain classification. The representations learned for domain classification are used as attention to select the most domain-related (both shared and domain-specific) features for different domains.

3. Domain attention model

In this section, we introduce our Domain Attention Model (DAM) for multi-domain sentiment classification. As illustrated in Fig. 1, in DAM there are two important modules, *i.e.*, domain module and sentiment module. The domain module tries to predict which domain a text belongs to, through which a good domain representation will be learned. Then the domain representation triggers an attention selection of the most important domain-related features in the sentiment module. Fig. 2 gives a more detailed illustration of our model.

Suppose there are M domains. Denote x_i^j as the ith text from domain j ($j = 1, 2, ..., M$). It can be represented as a list of words, *i.e.*, $x_i^j = (w_1, w_2, \cdots, w_n)$, where n is the number of words within the text. Then we turn these words into vectors using an embedding lookup matrix E similar in [26]. This is actually a mapping C from any element i of E to a real-valued vector $C(i) \in \mathscr{R}^m$. In practice, this is done via

looking up an embedding matrix E, in which each row stands for a unique word token. To process multiple texts as a batch, we pad texts of various lengths within a batch to the same length using a special token <PAD>. By doing this, the text x_i^j is represented as a real-valued matrix $X_i^j \in \mathscr{R}^{n\times m}$.

3.1. Domain module

We can utilize shared features across all domains and domain-specific features to obtain better classification results for multiple domains. Also, motivated by Yang et al. [34] and Kumar et al. [35], we can use a high level representation as a query (such as "*what is the informative word*" in Yang et al.'s work [34]) to select the most discriminative features for desired domain. Unlike Kumar et al. [35], we employ a domain classification module to obtain a document-level context vector, rather than learning the representation of a fixed sentence. All these ideas lead to our domain attention model, *i.e.*, using a precise domain representation as the attention to select the most domain-related discriminative features for sentiment classification.

The goal of domain module is to obtain a good domain representation for a given text. Fortunately, domain classification, or so called topic classification, is a much simpler task than sentiment classification. Unlike sentiment which is always expressed subtly and semantically integrated, domain information can be easily acquired by the domain-related entities used within a text, such as *actor* in *DVD* domain and *battery* in *Electronics* domain.

In domain module, we use a bidirectional long-short term memory (BiLSTM) network to gain the domain representation. Using $f_{BiLSTM}(\cdot)$ to denote the process on text embeddings, we can formulate this process as

$$h_d = f_{BiLSTM}(X_i^j) = f_{BiLSTM}((wv_1, wv_2, \cdots, wv_n)^T), \tag{1}$$

where wv_i denotes the word vector representation of the i-th word in the text. In detail, h_d is the combination of the outputs from a forward LSTM network and a backward LSTM network, which is,

$$\begin{aligned} h_d &= \overrightarrow{h_d} \oplus \overleftarrow{h_d} \\ &= \overrightarrow{f_{LSTM}}(X_i^j) \oplus \overleftarrow{f_{LSTM}}(X_i^j), \end{aligned} \tag{2}$$

where $\oplus$ means concatenation and $f_{LSTM}(\cdot)$ means the transform on text embeddings. The domain representation h_d is then fed into a softmax classifier to predict its domain class $\hat{d}$, which can be formulated as $\hat{d} = f_d(h_d)$, where $f_d(\cdot)$ stands for the mapping from domain representation to its domain label. After training, we mainly use h_d for the following sentiment module. The domain classification accuracy can be seen as an indicator of the discriminative ability for domain classification.

Selection of recurrent network: We use the long short term memory networks (LSTM) for the recurrent part. Long short term memory network can be seen as a variant of the basic recurrent network, mainly to address the long-time dependency problem. It has special units that can pass their states to the next timestep. LSTM network replaces the hidden units with more flexible cells. The cell itself is a small network, which can decide its own reading, writing or resetting behavior. The LSTM network we use is similar to Graves [36]. We don't consider peephole connections.

Thus, given an input sequence $\mathbf{s} = (x_1, x_2, \cdots, x_n)$, the update procedure from timestep $t-1$ to t can be described as follows:

$$i_t = \sigma(W^{ix}x_t + U^{ih}h_{t-1} + b^i), \tag{3}$$

$$f_t = \sigma(W^{fx}x_t + U^{fh}h_{t-1} + b^f), \tag{4}$$

$$o_t = \sigma(W^{ox}x_t + U^{oh}h_{t-1} + b^o), \tag{5}$$

$$\widetilde{C}_t = \phi(W^{cx}x_t + U^{ch}h_{t-1} + b^c), \tag{6}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t, \tag{7}$$

$$h_t = o_t \odot \phi(c_t), \tag{8}$$

where $\odot$ denotes the element-wise multiplication, $i_t, f_t, o_t \in \mathscr{R}^h$ stand for the input gate, forget gate, output gate respectively. The LSTM cell maintains an internal cell state. In one timestep, the input gate determines how much the outside information can come in, the output gate determines how much the internal information can go out, and the forget gate determines how much the internal information should be forgotten when passed to next timestep.

3.2. Sentiment module

The sentiment module in our DAM is another bidirectional LSTM network with attention mechanism. Different from the domain module, the sentiment module needs to attend all outputs during the recurrent process, which is also the key idea behind attention mechanism. Using f^s_{BiLSTM} to denote the calculation of a bidirectional LSTM with stepped outputs, the outputs can be formulated as

$$H^s = f^s_{BiLSTM}(X_i^j) = f^s_{BiLSTM}((wv_1, wv_2, \cdots, wv_n)^T), \tag{9}$$

where H^s denotes the stepped outputs of BiLSTM network. Note that unlike h_d, which is a vector, H^s here is a list of vectors. Similar to the DMN model [35], H^s can be seen as the episodic memories of the text. Denote the ith vector of H^s as h_i^s, *i.e.*, $h_i^s = (H^s)_{(i,\cdot)}$. After this, we use domain representation h_d to judge the importance of all the episodes. Attention weights are obtained by a one-layer feed-forward network, which can be formulated as

$$y_i^{\text{att}} = f(W^{\text{att}}(h_d \oplus h_i^s) + b^{\text{att}}), \tag{10}$$

where W^{att} and b^{att} are attention parameters. All the attention weights are then fed into a softmax layer to obtain probabilistic attention weights. The final representation used for sentiment classification (denoted as h_s) is the weighted sum of all the episodic memories, which is,

$$\alpha_i = \frac{\exp(y_i^{(a)})}{\sum_{i=1}^n \exp(y_i^{(a)})}, \tag{11}$$

$$h_s = \sum_{i=1}^n \alpha_i h_i^s. \tag{12}$$

The weighted vector h_s can be seen as a hidden representation of a text for sentiment classification, which is then fed into a fully-connected layer and a softmax layer to predict sentiment labels of texts. This process can be formulated as $\hat{s} = f_s(h_s)$, where $f_s(\cdot)$ stands for the mapping from the obtained hidden representation to its sentiment label and $\hat{s}$ is the final predicted sentiment label.

3.3. Training

Suppose there are M domains. Denote $\mathscr{D}^k = \{(x_i^k, d_i^k, s_i^k)\}_{i=1}^{N_k}, (k = 1, 2, ..., M)$ as the training samples from domain k, where d_i^k and s_i^k stand for domain label and sentiment label respectively. N_k is the number of labeled samples from domain k.

Our global cost function is a linear combination of sentiment prediction loss and domain prediction loss, which is

$$C_\theta = \frac{1}{N_k}\sum_{k=1}^{M}\sum_{i=1}^{N_k} L(s_i^k, p(\hat{s}_i^k|x_i^k)) + \frac{\lambda}{N_k}\sum_{k=1}^{M}\sum_{i=1}^{N_k} L\left(d_i^k, p\left(\hat{d}_i^k \middle| x_i^k\right)\right), \tag{13}$$

where d_i^k and s_i^k stand for true domain label and sentiment label respectively, which are encoded as one-hot representations. $p(\hat{s}_i^k|x_i^k)$ and $p\left(\hat{d}_i^k \middle| x_i^k\right)$ are the probabilistic outputs of sentiment and domain

predictions. $L(\cdot, \cdot)$ is the loss function to measure the gap between predicted label and true label. For both the domain and sentiment module, we adopt the cross-entropy cost function.

In Eq. (13), λ is a parameter that controls the relative importance of domain attention. Bigger λ means the model has a stronger domain regularization. Smaller λ means the attention part gets less information from the domain module. Especially, when λ reduces to zero, the domain representation h_d works just like the global attention used in [34,37]. Parameters are updated by end-to-end training using the ADADELTA [38] rule.

4. Experiments

4.1. Datasets and experimental settings

Two sentiment datasets are used in our experiments. The first dataset is Amazon multi-domain sentiment dataset[1] (denoted as Amazon) built by Blitzer et al. [15]. The Amazon dataset contains product reviews crawled from Amazon.com. This dataset includes four domains, *i.e., Books, DVD, Electronics* and *Kitchen*. For each domain, there are 1000 positive reviews, 1000 negative reviews and a large number of unlabeled reviews. The second dataset is Sanders Twitter Sentiment Dataset[2] (denoted as Sanders), which consists of 5513 hand-classified tweets related to four IT companies, i.e., Apple, Google, Microsoft and Twitter. We regard each IT company as a domain. Following the work of Wu and Huang [5], we only use positive and negative samples. The statistical information of these multi-domain sentiment dataset is shown in Table 1.

The labeled samples are randomly splitted as training data, development data and testing data with the proportion of 70%, 20% and 10% respectively. We split each text into word tokens using punctuations and spaces as delimiters. We set the word embedding dimension to be 300, and LSTM state dimension to be 300. For training, we use stochastic gradient descent based on mini-batches with a batch size of 64. The training process is monitored by the validation loss for early-stopping. After training, the model is evaluated by a separate test dataset. We also use Glove[3] pre-trained vectors [39] to initialize all the word vectors. We set λ in Eq. (13) to be 0.04, which is selected by grid search. We repeat each experiment 10 times and average results are reported.[4]

4.2. Baseline methods

We compare our domain attention model with several baseline methods including some state-of-the-art methods. These methods include traditional machine learning methods such as support vector machine (SVM) as well as neural network based models. The models to be compared are listed as follows:

- **LS-single, SVM-single** Supervised sentiment classification methods, *i.e.*, Least Squares and Support Vector Machine, which are trained and tested in each single domain.
- **LS-all, SVM-all** Least Squares and Support Vector Machine, which are trained using samples from all domains and tested in each domain.
- **MDSC-Com** Multi-Domain Sentiment Classification using classifiers combination proposed by Li and Zong [20]. In each domain, an SVM classifier is trained using labeled samples of this domain. The final prediction is made by combining all the predictions of these classifiers.

[1] https://www.cs.jhu.edu/~mdredze/datasets/sentiment/ (accessed Nov. 27, 2017)
[2] http://www.sananalytics.com/lab/twitter-sentiment/ (accessed Feb. 16, 2018)
[3] https://nlp.stanford.edu/projects/glove/ (accessed Nov. 27, 2017)
[4] The codes of our work can be found at https://github.com/yuanzhigang10/domain-attention/

Table 1
Statistical information of Amazon and Sanders datasets.

Amazon	Books	DVD	Electronics	Kitchen
Positive	1000	1000	1000	1000
Negative	1000	1000	1000	1000
Sanders	Apple	Google	Microsoft	Twitter
Positive	191	218	93	68
Negative	377	61	138	78

- **RMTL** Regularized Multi-Task Learning method proposed in [25]. In RMTL, models of multiple domains are constrained to be similar to their average model.
- **MTL-Graph** Multi-Task Learning with Graph regularization proposed in [7]. In MTL-Graph, the pairwise domain relatedness is based on sentiment word distribution.
- **CMSC** Collaborative Multi-domain Sentiment Classification model proposed in [6]. In CMSC, the sentiment classifier for each domain is decomposed into a common one and a domain-specific one.
- **LSTM-single, LSTM-all** Long Short Term Memory network [36]. Similarly, LSTM-single is trained and tested in each single domain, while LSTM-all is trained using samples from all domains and tested in each domain.
- **MTL-CNN** Multi-Task Learning with Convolutional Neural Network proposed in [29]. In MT-CNN, the embedding layer, *i.e.*, lookup-tables are shared while other layers such as CNN are task-specific.
- **MTL-DNN** Multi-Task Learning with Deep Neural Network proposed in [40]. In MTL-DNN, a hidden layer is shared.
- **MTL-SM** Multi-Task Learning with Shared Memory proposed in [31]. In MTL-SM, an external memory is introduced to store the knowledge shared by different related tasks.
- **ASP-MTL** Adversarial Multi-task Learning proposed in [9]. ASP-MTL is an adversarial multi-task framework, in which the shared and private feature spaces are inherently disjoint by introducing orthogonality constraints.
- **NeuroSent** Neural word embeddings approach proposed in [10]. In NeuroSent, multiple output layers are used for combining domain overlap scores with domain-specific sentiment predictions.

4.3. Performance evaluation

The experimental results of the above methods on Amazon dataset and Sanders dataset are listed in Tables 2 and 3 respectively. We refer to our model as DAM.

From Table 2, we can see that our approach outperforms baseline methods in all four domains of the Amazon dataset. Our approach outperforms single domain sentiment classification methods such as *LS-single* and *SVM-single*. This indicates that using data from other related domains can help train more accurate sentiment classifiers when labeled data is scarce. The sentiment classification methods that use labeled data from all domains, such as *LS-all* and *SVM-all*, perform better than single domain sentiment classification methods. This also verifies the effectiveness of using labeled data from other related domains. However, our approach can outperform them. This is because in these methods only one global common sentiment classifier is trained, which cannot capture domain-specific expressions in different domains. Our approach can alleviate this by extracting domain-related features using domain attention. Our approach also performs better than *MDSC-com*, which is based on the combination of sentiment classifiers. This result validates that incorporating sentiment information from different domains at learning stage is more suitable than incorporating them at classification stage. Our approach also outperforms multi-domain sentiment classification methods such as *RMTL, MTL-Graph* and *CMSC*. This indicates our approach can model the sentiment relatedness between different domains more effectively. In *RMTL*, the models for different domains are constrained to be close to their average model.

Table 2
Sentiment classification accuracy (in percentage) of different domains on Amazon dataset.

Domain	Books	DVD	Electronics	Kitchen
LS-single	77.80	77.88	81.63	84.33
SVM-single	78.56	78.66	83.03	84.74
LS-all	78.40	79.76	84.67	85.73
SVM-all	79.16	80.97	85.15	86.06
MDSC-Com	79.07	80.09	83.77	85.54
RMTL	81.33	82.18	85.49	87.02
MTL-Graph	79.66	81.84	83.69	87.06
CMSC	81.16	82.08	85.85	87.13
LSTM-single	78.51	78.74	80.72	81.80
LSTM-all	83.74	79.70	82.87	83.56
MTL-CNN	80.20	81.00	83.40	83.00
MTL-DNN	79.70	80.50	82.50	82.80
MTL-SM	82.80	83.00	85.50	84.30
ASP-MTL	84.00	85.50	86.80	86.2
NeuroSent	79.66	80.90	86.41	86.86
DAM	**87.75**	**86.58**	**87.50**	**88.93**

Table 3
Sentiment classification accuracy (in percentage) of different domains on Sanders dataset.

Domain	Apple	Google	Microsoft	Twitter
LS-all	84.39	85.07	81.89	76.61
SVM-all	84.49	85.79	82.27	76.01
RMTL	85.63	85.87	82.28	77.37
MTL-Graph	85.41	84.68	77.45	73.15
CMSC	**87.03**	88.08	83.11	80.31
LSTM-all	79.59	85.19	79.17	78.57
MTL-CNN	83.67	85.23	83.67	79.36
MTL-DNN	82.40	84.50	81.10	78.80
ASP-MTL	86.40	87.60	85.20	81.30
NeuroSent	82.21	85.70	84.06	81.86
DAM	86.76	**89.46**	**86.36**	**82.71**

However, the relations between different domains are not fully exploited. In *MTL-Graph*, the pair-wise relations between different domains are exploited, while the shared sentiment knowledge is ignored. In contrast with *RMTL* and *MTL-Graph, CMSC* decomposes the sentiment classifier for each domain into a global one and a domain-specific one. In addition, both general sentiment knowledge and domain similarities are exploited. In contrast with *CMSC*, our approach also extracts shared and domain-specific features, but combines them at feature-level using distributed representations. The superior performance indicates that combining shared and domain-specific features at feature-level is more suitable for multi-domain sentiment classification. *LSTM-single* and *LSTM-all* show similar patterns as *LS-single* and *LS-all*, since *LSTM-all* uses labeled samples from all domains compared with *LSTM-single*. Our approach still outperforms *LSTM-all*. This is because *LSTM-all* simply combines labeled data from all domains, without any effective multi-task learning schemes. We also compare our approach with some state-of-the-art neural network based multi-task learning models, such as *MTL-CNN, MTL-DNN, MTL-SM, ASP-MTL* and *NeuroSent*. In *MTL-CNN*, only a common lookup layer is shared and different CNNs ared used for different tasks. In *MTL-DNN*, a hidden layer is shared to extract more high-level sentiment features. In *MTL-SM*, an external memory is introduced to store the knowledge shared by different related tasks. In *ASP-MTL*, an adversarial multi-task framework is used to better separate shared and private feature spaces. Both *MTL-SM* and *ASP-MTL* use different layers to model different tasks, which may suffer from large number of domains. In *NeuroSent*, the sentiment predictions of different domains are weighted linearly, which cannot model complex domain relatedness. In contrast with these methods, our approach extracts domain-related features at feature-level using domain attention, reducing the need for designing a specific layer for each task. The attention scheme can model more complex domain relationships than linear weighting used in *NeuroSent*. Our approach outperforms these methods consistently, which indicates that using domain attention to extract domain-related features is a more appropriate way for multi-domain sentiment classification.

For the Sanders dataset, we only report the results of multi-domain methods for simplicity. As can be seen in Table 3, the results on Sanders dataset are consistent with the above analysis. Our approach outperforms all baseline methods except that *CMSC* has slightly higher accuracy on *Apple* domain. An interesting observation is that neural baseline method *LSTM* fails to outperform models such as *LS* and *SVM* consistently due to the small size of this dataset. Since Amazon dataset has more training data than Sanders, we use Amazon dataset to conduct our following additional experiments.

To further testify the contribution of domain representation, we conduct some experiments in which the domain attention is taken out while all the other settings remain the same. Since the core idea of our approach is using domain representation to weight LSTM's outputs, we replace this procedure with a vanilla LSTM (denoted as *LSTM-vanilla*) and LSTM with mean-pooling (denoted as *LSTM-mean*). In *LSTM-vanilla*, the last output of LSTM (concatenation of both directions) is used for following sentiment classification. In *LSTM-mean*, the stepped outputs of LSTM are averaged. Since LSTM with mean-pooling gives average attention to all outputs, this contrast can provide a direct comparison of our approach. The experimental results are shown in Fig. 3.

From Fig. 3, we can see that *LSTM-mean* performs better than *LSTM-vanilla*. This is because *LSTM-mean* can attend the outputs from all timesteps, alleviating the forgetting problem of vanilla LSTM. However, *LSTM-mean* gives equal attentions to all outputs, regardless of the domain property. In contrast with *LSTM-mean*, our approach (*DAM*) pays different attentions to LSTM's outputs using domain representation as a selection scheme. The superior performance of our approach verifies the effectiveness of our approach.

4.4. Influence of training data size

We want to verify whether our model can alleviate the need for large amounts of labeled samples by using domain attention for multi-domain sentiment classification. We vary the number of labeled samples in each domain from 200 to 1000, with a step size of 200. The other settings remain the same. We also use *LSTM-vanilla* and *LSTM-mean* as baseline methods. The experimental results are shown in Fig. 4. For these experiments, we take *DVD* domain as example to observe the dynamic performance. Different domains show similar patterns in our experiments.

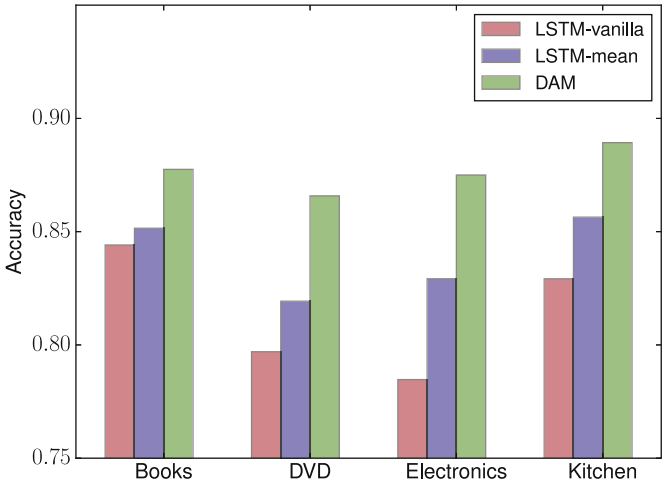

Fig. 3. The performance comparison of our approach (*DAM*) with vanilla LSTM (*LSTM-vanilla*) and LSTM with mean-pooling (*LSTM-mean*).

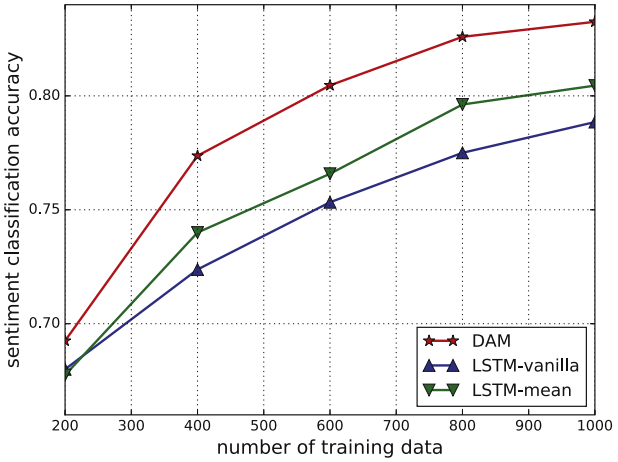

Fig. 4. Sentiment classification accuracy on *DVD* domain of our approach (*DAM*) with *LSTM-vanilla* and *LSTM-mean* with respect to different number of labeled samples.

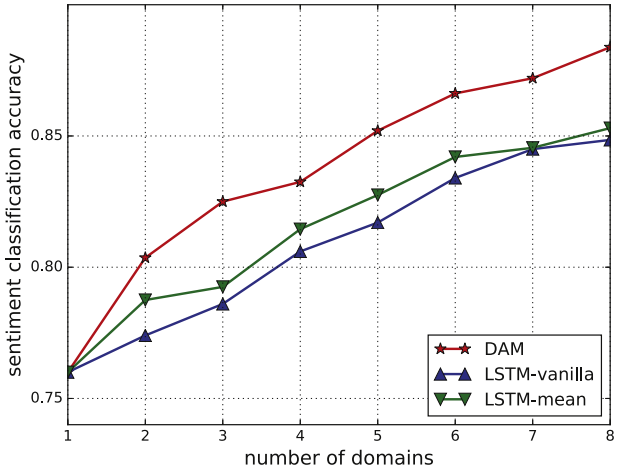

Fig. 5. Sentiment classification accuracy on *DVD* domain of our approach (*DAM*) with *LSTM-vanilla* and *LSTM-mean* with respect to different number of domains.

From Fig. 4, we can see that as the number of labeled samples increases from 200 to 1000, both our approach and baseline methods perform better. This is intuitive since the sentiment classifier can benefit from more training data. However, our approach performs better than baseline methods no matter what the number of labeled samples is. This further verifies our approach can pay more attention to domain-related features using domain representation as a selection scheme. From Fig. 4, the performance of our approach with 600 labeled samples is still better than *LSTM-vanilla* and *LSTM-mean* with 1000 labeled samples. These results validate that by training sentiment classifiers for multiple domains simultaneously and use domain representation as attention, our approach can effectively reduce the need for labeled samples and improve the sentiment classification performance when training data is scarce.

4.5. Influence of number of domains

The goal of our proposed approach is to exploit domain relatedness using domain attention to gain better performance in multi-domain sentiment classification scenarios. It would be more convincing if we increase the domains step by step and test the performance of our model. For Amazon dataset, it also contains another 21 domains besides the commonly used 4 domains mentioned above, such as *Beauty, Software, Music* and so on. We select domains which also contain 1000 positive samples and 1000 negative samples in each domain. The resulting dataset contains 8 domains, which are *Books, DVD, Electronics, Kitchen, Music, Sports, Video* and *Toys*. We also choose *DVD* as the focused domain and add the other domains one by one to see the corresponding performance for *DVD* domain. Similarly, we also use *LSTM-vanilla* and *LSTM-mean* as baseline methods. The experimental results are shown in Fig. 5.

According to Fig. 5, as the number of related domains increases, both our approach and baseline methods perform better. This is also intuitive since the labeled samples from other related domains can also provide sentiment information for sentiment classifier. However, compared with *LSTM-vanilla* and *LSTM-mean*, our approach *DAM* has better improvement no matter what the number of domains is. These results validate that our approach can better exploit the domain relatedness and make full use of the labeled samples from all domains to improve single-domain performance.

4.6. Parameter analysis

Next, we explore the influence of parameter settings. In Eq. (13), λ is a parameter that controls the relative importance of domain attention. Bigger λ means the model has a stronger domain regularization and vice versa. We evaluate the influence of λ to better understand the effect of domain attention. Experimental results are shown in Fig. 6.

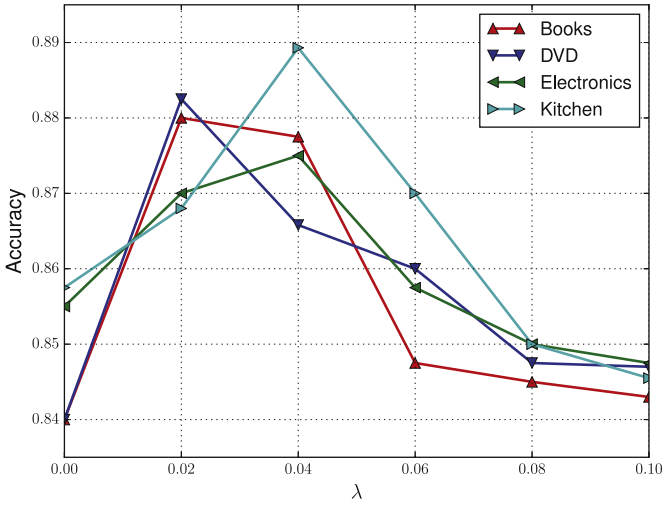

Fig. 6. The performance of our domain attention model with different λ.

From Fig. 6, we can see that the influences of λ on sentiment classification accuracy in different domains have similar patterns. As λ increases from a small value, the sentiment classification accuracy first increases and then decreases. When λ is too small, the domain information is almost ignored. As a result, the model cannot extract precise domain-related features in each domain. When λ increases to a proper value, the domain representation works as a good regularizer for multi-domain sentiment classification. But when λ becomes too large, the domain knowledge is overemphasized and the labeled samples for sentiment classification are not fully exploited. In practice, we find that a value between $0.02 \sim 0.04$ would be appropriate for all these four domains.

It is worth noting that when λ reduces to zero, the performance of our approach is still better than *LSTM-vanilla* and *LSTM-mean*. In this case, the representation serving as attention can be regarded as a global representation of the input text instead of domain representation. As studied in [34,37], this kind of attention mechanism can also improve sentiment classification accuracy. The superior performance when λ increases to a proper value also verifies the effectiveness of domain attention.

4.7. Alternative settings of DAM

In this section, we explore some alternative settings of our proposed model with respect to different sub-modules and explain why we choose current settings introduced in Section 3.

4.7.1. Selection of recurrent network

As introduced in Section 3.1, we uses LSTM as the recurrent layer. This selection is a trade-off between classification performance and time efficiency. Apart from LSTM, we also conduct additional experiments using vanilla RNN (denoted as RNN), Gated Recurrent Unit (denoted as GRU) and LSTM with peephole connections (denoted as LSTM-PH). The sentiment classification accuracy (averaged result on four domains) and training time cost[5] are reported in Table 4.

According to Table 4, LSTM and LSTM-PH perform better than vanilla RNN and GRU. In contrast with LSTM, LSTM-PH has only a slight improvement. However, the training time increases from 310 s/epoch to 411 s/epoch. As a tradeoff between classification performance and time cost, we choose the basic LSTM without peephole connections as the recurrent network.

4.7.2. Attention in domain module

The core part of DAM is the use of domain representation as attention. In fact, the domain module can also be assisted by the attention mechanism. We conduct additional experiments in which an attention process is also introduced into the domain module. The attention process is similar to that used in [34]. Experimental results on Amazon dataset are shown in Table 5. We denote DAM with attention mechanism in the domain module as DAM-datt.

As can be seen in Table 5, DAM-datt performs almost the same as DAM. We believe there are two reasons behind this observation. First, compared with sentiment classification, domain classification is a simpler task. In our experiments, the domain classification accuracy is always above 90%. Thus, the domain module benefit less than sentiment module when using attention mechanism. Second, the domain representation can be seen as a regularizer, which is not our focused task. As a result, the degradation of the domain module may not harm the final performance so much. Similarly, as a tradeoff between classification performance and time cost (model complexity), we ignore attention mechanism for the domain module.

4.8. Visualization

In order to get a deeper insight into our domain attention model, we conduct additional experiments to verify our idea of finding the most domain-related features. We visualize attention at two different levels, *i.e.*, feature-level and document-level.

4.8.1. Feature-level visualization

When training the domain attention model, we track and accumulate the attention weight of all tokens (including words and punctuation marks) that appeared, then calculate their average attention weights. In this way, we can get a better knowledge about what kind of tokens this model pays more attention to when trying to predict the sentiment of a text.

We organize tokens with largest attention weights into five groups. The first one is **Common** group, in which the tokens have high attention weights across all domains. By intuition, these tokens should be general sentiment expressions such as "*good*" and "*bad*". The other four groups are **Domain** groups, each one corresponding to one unique domain. For example, words that appeared in *DVD* group only have higher attention weights in *DVD* domain. Tokens within these five groups are shown in Table 6.

[5] All experiments run with Tensorflow on an NVIDIA K80 GPU.

Table 4
Sentiment classification accuracy (in percentage) and training time cost when using different recurrent units.

Recurrent type	RNN	GRU	LSTM	LSTM-PH
Accuracy (averaged)	75.01	83.31	84.38	84.41
Time cost (s/epoch)	71	246	310	411

Table 5
Sentiment classification accuracy (in percentage) of DAM with and without attention mechanism in the domain module.

Domain	Books	DVD	Electronics	Kitchen
DAM	87.75	86.58	87.50	88.93
DAM-datt	87.50	87.20	88.50	88.50

Table 6
Tokens with the largest attention weights in each group.

Groups		Tokens with the largest attention weights
Common		*money, recommend, great, buy, love, good, really*
Domain	Books	*best, ever, looking, better, want, not, n't, !, lot wonderful, stories, reading, ideas, text, books, highly gives, written, style, understand, help, readers, novel*
	DVD	*boring, collection, loved, stars, funny, watch, classic episodes, version, worst, pretty, fan, original, actors*
	Electronics	*labtop, working, ipod, battery, camera, digital antenna, getting, printer, service, hours, software*
	Kitchen	*fit, perfect, glass, easily, room, light, large, plastic received, clean, design, try, top, machine, vacuum*

As shown in Table 6, tokens that are most domain-related and sentiment-discriminative are automatically selected. For **Common** group, general sentiment words are captured, such as "*good*", "*best*" and "*recommend*". Besides, tokens with multiple attributes are also selected, such as verbs ("*buy*", "*try*"), negation words ("*not*", *n't*") and exclamation mark.

For each domain group, aspects that are most important are selected by this model, such as "*stories, styles*" in *Books*, "*stars, actors*" in *DVD*, "*battery, camera*" in *Electronics* and "*glass, design*" in *Kitchen*. Along with this, words that convey sentiment for these aspects are also selected, such as "*novel*" for a book, "*original*" for a movie plot, "*hours*" for battery use, and "*easily*" for kitchen appliance use.

In Table 6, the attention weights of tokens are calculated by domain, regardless of the sentiment polarity of the text they are taken from. To further illustrate that the same word may convey opposite sentiment in different domains, we calculate the average attention weight of each token in each sentiment group of each domain. In Section 1, we use the word "*easy*" as an example to highlight the domain-dependency problem. The detailed attention weights of the word "*easy*" are listed in Table 7. From Table 7, we can see that the word "*easy*" has larger attention weights in negative reviews in *Books* and *DVD* domain, while with larger attention weights in positive reviews in *Electronics* and *Kitchen* domain. This result is consistent with our analysis. For *Books* and *DVD* domain, "*easy*" is commonly used to describe the plot of a book or movie, which conveys negative sentiment. While in *Electronics* and *Kitchen* domain, "*easy*" is commonly used to describe the usability of electronic appliances, which conveys positive sentiment.

The results of feature-level visualization validate our assumption that both shared and domain-specific features for sentiment classification can be automatically learned by using domain representation as attention. The domain-dependency problem can also be alleviated by this method.

Table 7
The detailed attention weights of "*easy*".

Groups	Books		DVD		Electronics		Kitchen	
	pos	neg	pos	neg	pos	neg	pos	neg
Easy	0.2629	**0.2908**	0.2811	**0.2979**	**0.3080**	0.2966	**0.3112**	0.2821

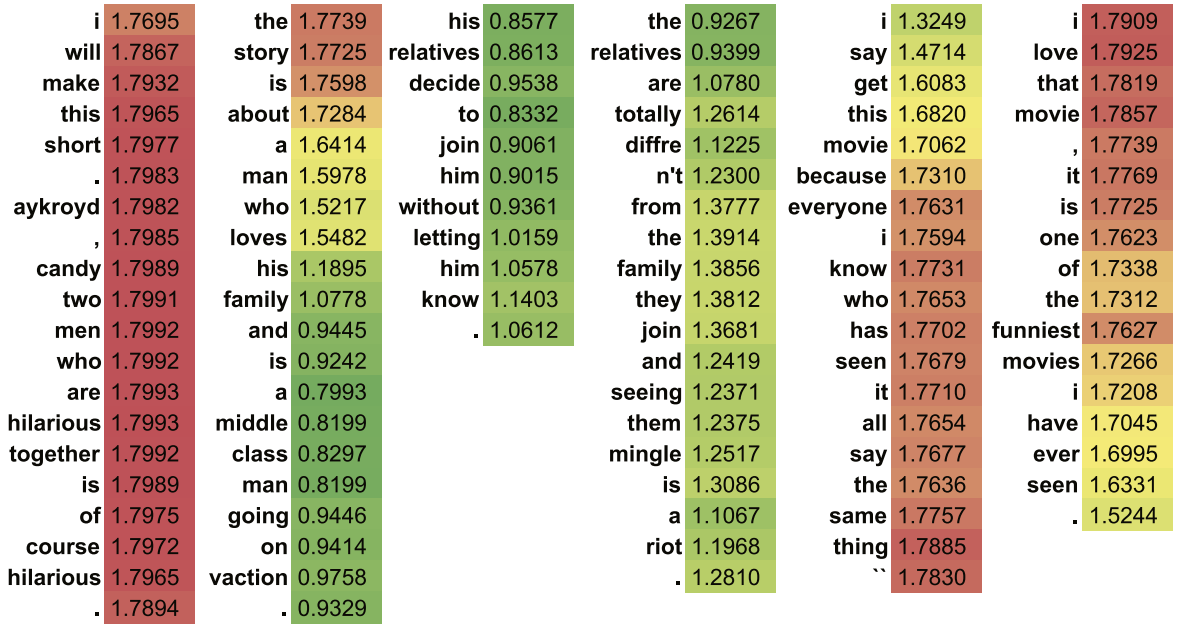

Fig. 7. Visualization of attention weights in a positive review.

4.8.2. Document-level visualization

To further demonstrate that our proposed domain attention model can pay attention to important features in different domains, we visualize our model at document-level.

To achieve this, we pick one review drawn from *DVD* domain which is correctly classified by our attention model, but misclassified by *LSTM-mean* model. We choose *LSTM-mean* as baseline because it gives equal attention weights to all the features. By visualizing the attention weights within a review, we can monitor the changing of attention weights along word series. The original text of this review and its corresponding weights are shown in Fig. 7. In Fig. 7, sentence segments are arranged in columns for better visualization. The red background means the highest attention weights, while green the lowest. The attention weights in Fig. 7 have been normalized according to the length of the review.

This review is related to the movie "The Great Outdoors". As can be seen within the text, although the author said "*it is one of the funniest movies I have ever seen*" in the end, there exists many descriptions of the movie plot which seem to be tragic. As a result, the plot descriptions mislead *LSTM-mean* to a wrong prediction. As shown in Fig. 7, we can see that our model gives more attention to the first two sentences, since they express a hilarious attitude. When comes to the plot description, this model gradually reduces its attention and the whole middle part is almost ignored. This means our model manages to eliminate irrelevant distractions from the plot description. At last, when the author starts to express his/her attitude again, this model gradually increases its attention, and highest attention is given to words like *love* and *funniest*. Using this example, we can see that the domain attention model is able to pay attention to the most important segments and eliminate irrelevant distractions.

5. Conclusions

In this paper, we propose a domain attention model for multi-domain sentiment classification. In our approach, a domain classifier is introduced to help improve sentiment classification. Specifically, a domain representation is used to trigger an attention process to select the most domain-related features. The domain representation is obtained through an auxiliary domain classification task, which works as domain regularizer. To fit the two tasks into one network architecture, the domain representation of a sentence is used as input of attention weights on the original sentence. Therefore, the attention weights naturally adapt to different domains. In contrast with existing multi-domain sentiment classification methods, our approach can extract the most discriminative features from a shared hidden layer in a more compact way and reduce the need for a large number of labeled samples. Experimental results on two multi-domain sentiment datasets show that our approach can effectively improve the performance of multi-domain sentiment classification.

Acknowledgments

The authors thank the reviewers for their insightful comments and constructive suggestions on improving this work. This research is supported by the National Key Research and Development Program of China (no. 2016YFB0800402) and the National Natural Science Foundation of China (no. U1636113, no. U1705261 and U1536201).

References

[1] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, Neural Netw. 18 (5–6) (2005) 602–610, http://dx.doi.org/10.1016/j.neunet.2005.06.042.
[2] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up? sentiment classification using machine learning techniques, cs.CL/0205070 (2002). http://arxiv.org/abs/cs.CL/

0205070
[3] B. Pang, L. Lee, Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales, ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25–30 June 2005, University of Michigan, USA, (2005), pp. 115–124. http://aclweb.org/anthology/P/P05/P05-1015.pdf
[4] B. Chen, L. Zhu, D. Kifer, D. Lee, What is an opinion about? exploring political standpoints using opinion scoring model, Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11–15, 2010, (2010). http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1863
[5] F. Wu, Y. Huang, Collaborative multi-domain sentiment classification, 2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14–17, 2015, (2015), pp. 459–468, http://dx.doi.org/10.1109/ICDM.2015.68.
[6] F. Wu, Z. Yuan, Y. Huang, Collaboratively training sentiment classifiers for multiple domains, IEEE Trans. Knowl. Data Eng. 29 (7) (2017) 1370–1383, http://dx.doi.org/10.1109/TKDE.2017.2669975.
[7] J. Zhou, J. Chen, J. Ye, Malsar: Multi-task learning via structural regularization, volume 21, 2011.
[8] J. Liu, S. Ji, J. Ye, Multi-task feature learning via efficient l2, 1-norm minimization, UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18–21, 2009, (2009), pp. 339–348. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1634&proceeding_id=25
[9] P. Liu, X. Qiu, X. Huang, Adversarial multi-task learning for text classification, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers, (2017), pp. 1–10, http://dx.doi.org/10.18653/v1/P17-1001.
[10] M. Dragoni, G. Petrucci, A neural word embeddings approach for multi-domain sentiment analysis, IEEE Trans. Affective Comput. 8 (4) (2017) 457–470, http://dx.doi.org/10.1109/TAFFC.2017.2717879.
[11] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, (2014). CoRR abs/1409.0473 Arxiv: 1409.0473.
[12] T. Rocktäschel, E. Grefenstette, K.M. Hermann, T. Kociský, P. Blunsom, Reasoning about entailment with neural attention, (2015). CoRR abs/1509.06664 Arxiv: 1509.06664.
[13] D. Tang, B. Qin, T. Liu, Aspect level sentiment classification with deep memory network, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016, (2016), pp. 214–224. http://aclweb.org/anthology/D/D16/D16-1021.pdf
[14] C.M. Bishop, Pattern recognition and machine learning, Information science and statistics, fifth ed., Springer, 2007. http://www.worldcat.org/oclc/71008143
[15] J. Blitzer, M. Dredze, F. Pereira, Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification, ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23–30, 2007, Prague, Czech Republic, (2007). http://aclweb.org/anthology/P07-1056
[16] S.J. Pan, X. Ni, J. Sun, Q. Yang, Z. Chen, Cross-domain sentiment classification via spectral feature alignment, Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26–30, 2010, (2010), pp. 751–760, http://dx.doi.org/10.1145/1772690.1772767.
[17] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V.S. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (2016) 59:1–59:35. http://jmlr.org/papers/v17/15-239.html
[18] X. Glorot, A. Bordes, Y. Bengio, Domain adaptation for large-scale sentiment classification: A deep learning approach, Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011, (2011), pp. 513–520.
[19] M. Chen, Z.E. Xu, K.Q. Weinberger, F. Sha, Marginalized denoising autoencoders for domain adaptation, Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012, (2012). http://icml.cc/2012/papers/416.pdf
[20] S. Li, C. Zong, Multi-domain sentiment classification, ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15–20, 2008, Columbus, Ohio, USA, Short Papers, (2008), pp. 257–260. http://www.aclweb.org/anthology/P08-2065
[21] S. Li, C. Huang, C. Zong, Multi-domain sentiment classification with classifier combination, J. Comput. Sci. Tech. 26 (1) (2011) 25–33, http://dx.doi.org/10.1007/s11390-011-9412-y.
[22] Z. Li, Y. Zhang, Y. Wei, Y. Wu, Q. Yang, End-to-end adversarial memory network for cross-domain sentiment classification, Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19–25, 2017, (2017), pp. 2237–2243, http://dx.doi.org/10.24963/ijcai.2017/311.
[23] Z. Li, Y. Wei, Y. Zhang, Q. Yang, Hierarchical attention transfer network for cross-domain sentiment classification, Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI 2018, New Orleans, Lousiana, USA, February 2–7, 2018, (2018). https://hsqmlzno1.github.io/assets/publications/HATN2018.pdf
[24] R. Caruana, Multitask Learning, volume 28, 1997, pp. 41–75. doi:10.1023/A:1007379606734.
[25] T. Evgeniou, M. Pontil, Regularized multi–task learning, Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22–25, 2004, (2004), pp. 109–117, http://dx.doi.org/10.1145/1014052.1014067.
[26] Y. Bengio, R. Ducharme, P. Vincent, C. Janvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (2003) 1137–1155. http://www.jmlr.org/papers/v3/bengio03a.html
[27] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, (2013). CoRR abs/1301.3781. Arxiv: 1301.3781.
[28] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5–8, 2013, Lake Tahoe, Nevada, United States. (2013), pp. 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality
[29] R. Collobert, J. Weston, A unified architecture for natural language processing: deep neural networks with multitask learning, Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5–9, 2008, (2008), pp. 160–167, http://dx.doi.org/10.1145/1390156.1390177.
[30] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P.P. Kuksa, Natural language processing (almost) from scratch, J. Mach. Learn. Res. 12 (2011) 2493–2537. http://dl.acm.org/citation.cfm?id=2078186
[31] P. Liu, X. Qiu, X. Huang, Deep multi-task learning with shared memory for text classification, Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1–4, 2016, (2016), pp. 118–127. http://aclweb.org/anthology/D/D16/D16-1012.pdf
[32] P. Liu, X. Qiu, X. Huang, Recurrent neural network for text classification with multi-task learning, Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9–15 July 2016, (2016), pp. 2873–2879. http://www.ijcai.org/Abstract/16/408
[33] K. Xu, J. Ba, R. Kiros, K. Cho, A.C. Courville, R. Salakhutdinov, R.S. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015, (2015), pp. 2048–2057. http://jmlr.org/proceedings/papers/v37/xuc15.html
[34] Z. Yang, D. Yang, C. Dyer, X. He, A.J. Smola, E.H. Hovy, Hierarchical attention networks for document classification, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12–17, 2016, (2016), pp. 1480–1489. http://aclweb.org/anthology/N/N16/N16-1174.pdf
[35] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, R. Socher, Ask me anything: Dynamic memory networks for natural language processing, Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19–24, 2016, (2016), pp. 1378–1387. http://jmlr.org/proceedings/papers/v48/kumar16.html
[36] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, volume 385 of Studies in Computational Intelligence, Springer, 2012, http://dx.doi.org/10.1007/978-3-642-24797-2.
[37] T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015, (2015), pp. 1412–1421. http://aclweb.org/anthology/D/D15/D15-1166.pdf
[38] M.D. Zeiler, ADADELTA: an adaptive learning rate method, (2012). CoRR abs/1212.5701 Arxiv:1212.5701.
[39] J. Pennington, R. Socher, C.D. Manning, Glove: Global vectors for word representation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, (2014), pp. 1532–1543. http://aclweb.org/anthology/D/D14/D14-1162.pdf
[40] X. Liu, J. Gao, X. He, L. Deng, K. Duh, Y. Wang, Representation learning using multi-task deep neural networks for semantic classification and information retrieval, NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015, (2015), pp. 912–921. http://aclweb.org/anthology/N/N15/N15-1092.pdf