Collaboratively Training Sentiment Classifiers for Multiple Domains

Fangzhao Wu, Zhigang Yuan, and Yongfeng Huang, *Senior Member, IEEE*

Abstract—We propose a collaborative multi-domain sentiment classification approach to train sentiment classifiers for multiple domains simultaneously. In our approach, the sentiment information in different domains is shared to train more accurate and robust sentiment classifiers for each domain when labeled data is scarce. Specifically, we decompose the sentiment classifier of each domain into two components, a global one and a domain-specific one. The global model can capture the general sentiment knowledge and is shared by various domains. The domain-specific model can capture the specific sentiment expressions in each domain. In addition, we extract domain-specific sentiment knowledge from both labeled and unlabeled samples in each domain and use it to enhance the learning of domain-specific sentiment classifiers. Besides, we incorporate the similarities between domains into our approach as regularization over the domain-specific sentiment classifiers to encourage the sharing of sentiment information between similar domains. Two kinds of domain similarity measures are explored, one based on textual content and the other one based on sentiment expressions. Moreover, we introduce two efficient algorithms to solve the model of our approach. Experimental results on benchmark datasets show that our approach can effectively improve the performance of multi-domain sentiment classification and significantly outperform baseline methods.

Index Terms—Sentiment classification, multiple domains, multi-task learning

1 Introduction

With the development of Web 2.0 websites, user generated content (UGC), such as product reviews, blogs, microblogs and so on, has been growing explosively. Mining the sentiment information in the massive user generated content can help sense the public's opinions towards various topics, such as products, brands, disasters, events, celebrities and so on, and is useful in many applications [1], [2]. For example, researchers have found that analyzing the sentiments in tweets has the potential to predict variation of stock market prices and presidential election results [3]. Classifying the sentiments of massive microblog messages is also helpful to substitute or supplement traditional polling, which is expensive and time-consuming [4]. Product review sentiment analysis can help companies improve their products and services, and help customers make more informed decisions [5], [6]. Analyzing the sentiments of user generated content is also proven useful for user interest mining, personalized recommendation, social advertising, customer relation management, and crisis management [7]. Thus, sentiment classification is a hot research topic in both industrial and academic fields [1], [2], [8], [9].

In many mainstream sentiment analysis methods, sentiment classification is regarded as a text classification problem [1], [2]. Supervised machine learning techniques, such as SVM, Logistic Regression and CNN, are frequently applied to train sentiment classifiers on labeled datasets and predict the sentiments of unseen texts [10], [11]. These methods have been used to analyze the sentiments of product reviews [5], microblogs [11], [12] and so on. However, sentiment classification is widely recognized as a domain-dependent problem [2], [13]. This is because in different domains there are different sentiment expressions, and the same word may convey different sentiments in different domains [2], [14]. For example, in the domain of electronic product reviews the word "easy" is usually positive, e.g., "this digital camera is easy to use." However, in the domain of movie reviews, "easy" is frequently used as a negative word. For instance, "the ending of this movie is easy to guess." Thus, the sentiment classifier trained in one domain may fail to capture the specific sentiment expressions of another domain, and its performance in a different domain is usually unsatisfactory [13].

An intuitive solution to this problem is to train a domain-specific sentiment classifier for each domain using the labeled samples of this domain [10]. However, the labeled data in many domains is usually scarce. In addition, since there are massive domains involved in online user generated content, it is very costly and time-consuming to annotate enough samples for them. Without sufficient labeled data, it is quite difficult to train an accurate and robust domain-specific sentiment classifier for each domain independently. The motivation of our work is that although each domain has its specific sentiment expressions, different domains also share many common sentiment words. For example, general sentiment words such as "best", "perfect", and "worst" convey consistent sentiment polarities in various domains. Thus, training sentiment classifiers for multiple domains simultaneously and exploiting the common sentiment knowledge shared among them can alleviate the problem of scarce labeled data [15], [16], [17].

• *The authors are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: {wfz12, yuanzg14}@mails.tsinghua.edu.cn, yfhuang@tsinghua.edu.cn.*

Manuscript received 29 May 2016; revised 11 Jan. 2017; accepted 10 Feb. 2017. Date of publication 15 Feb. 2017; date of current version 1 June 2017. Recommended for acceptance by B. Poblete.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2017.2669975

Motivated by above observations, in this paper we propose to train sentiment classifiers for multiple domains simultaneously in a collaborative way. In our approach, the sentiment classifier of each domain is decomposed into two components, i.e., a global one and a domain-specific one. The domain-specific sentiment classifier is trained using labeled samples of one domain and can capture the domain-specific sentiment expressions. The global sentiment classifier is shared by all domains and is trained on the labeled samples from various domains to have better generalization ability. It can capture the general sentiment knowledge consistent in different domains. In addition, we extract prior general sentiment knowledge from general-purpose sentiment lexicons and incorporate it into our approach to guide the learning of the global sentiment classifier. Besides, we propose to extract domain-specific sentiment knowledge for each domain from both limited labeled samples and massive unlabeled samples. The domain-specific sentiment knowledge is used to enhance the learning of domain-specific sentiment classifiers in our approach. Moreover, since different pairs of domains have different sentiment relatedness, we propose to measure the similarities between domains and incorporate them into our approach to encourage the sharing of sentiment information between similar domains. Two kinds of domain similarity measures are explored, one based on the textual content, and the other one based on the sentiment word distribution. The model of our approach is formulated as a convex optimization problem. In order to solve it efficiently, we introduce an accelerated algorithm based on FISTA [18]. In addition, we propose a parallel algorithm based on ADMM [19] to further improve its efficiency when domains to be analyzed are massive. Extensive experiments were conducted on benchmark sentiment datasets. Experimental results show our approach can improve sentiment classification performance effectively and outperform state-of-the-art methods significantly.

The major contributions of this paper are as follows:

- We propose a collaborative multi-domain sentiment classification approach (*CMSC*) based on multi-task learning to train sentiment classifiers for multiple domains simultaneously. It can exploit the sentiment relatedness between different domains and effectively alleviate the problem of scarce labeled data.
- We propose to extract domain-specific sentiment knowledge for each domain by propagating the sentiment scores inferred from limited labeled samples along contextual similarities mined from massive unlabeled samples.
- We propose to incorporate the similarities between domains into the collaborative learning process. In addition, we propose a novel domain similarity measure based on the sentiment expression distributions.
- We introduce an accelerated algorithm based on FISTA [18] to solve our model effectively, and propose a parallel algorithm based on ADMM [19] to further improve its efficiency.
- We evaluate our approach by conducting extensive experiments on the benchmark Amazon product review datasets. The experimental results show our approach can improve the sentiment classification accuracy by 2.74 percent in average compared with the best baseline method.

This paper is an extended and improved version of our previous work in [20]. In this version, we have made many important improvements in both algorithm and experiment. First, besides the single-node version algorithm for solving the model of our approach, in this paper we propose a parallel version algorithm, which is more efficient when there are a large number of domains to be analyzed. Second, in this paper we propose to extract domain-specific sentiment knowledge by combining limited labeled samples with massive unlabeled samples, which is not considered in previous work. The domain-specific sentiment knowledge contains rich specific sentiment expressions used in each domain and can provide important prior information for learning domain-specific sentiment classifiers. It is also used in our approach to measure the similarities between different domains. Third, a large multi-domain sentiment dataset was added to the experiments to evaluate the performance of our approach more thoroughly. In addition, more experiments were conducted. For example, we conducted experiments to explore the influence of training data size on the performance of our approach. We also conducted experiments to evaluate the time complexity of the proposed parallel algorithm. Besides, more detailed analysis and discussions on the experimental results are presented in this paper. Thus, compared with the previous version work [20], a large amount of new content has been added to this paper.

The rest of this paper is organized as follows. In Section 2, we briefly review several related works. In Section 3, we introduce two important components in our approach, i.e., domain-specific sentiment knowledge extraction and domain similarity measure. In Section 4, we present our collaborative multi-domain sentiment classification approach as well as the optimization algorithms in detail. In Section 5, we report the experimental results on benchmark datasets. In Section 6, we conclude this paper.

2 Related Work

2.1 Multi-Domain Sentiment Classification

Sentiment classification has been widely known as a highly domain-dependent problem [13], [14], [15], [21], [22]. An intuitive method to solve this problem is training a domain-specific sentiment classifier or building a domain-specific sentiment lexicon for each domain [10], [23]. For example, Pang et al. built sentiment classifiers for movie reviews using machine learning techniques such as SVM and Naive Bayes based on the labeled data of this domain [10]. Lu et al. proposed to construct a domain-specific sentiment lexicon by incorporating information from various sources in this domain, such as sentiment labels and linguistic heuristics [23]. However, in many domains, the labeled data is usually in limited size and insufficient to extract accurate and robust sentiment information. In addition, since there are massive domains involved in online user generated content, it is expensive and time-consuming to manually annotate enough samples for each domain.

A popular method to reduce the effort of manual annotation is using transfer learning to adapt the sentiment classifier from a source domain with sufficient labeled data to a target domain with scarce or no labeled data [24]. Many cross domain sentiment classification methods belong to this kind [13], [14], [22], [25]. For example, Blitzer et al. proposed a sentiment domain adaption method based on structural correspondence learning (*SCL*) algorithm [13]. The core idea of *SCL* is finding correspondence among features from

different domains by computing their associations with pivot features. Pan et al. proposed a spectral feature alignment (*SFA*) algorithm for cross domain sentiment classification to reduce the gap of sentiment expressions from different domains [22]. He et al. proposed to extract polarity-bearing topics based on a modified joint sentiment-topic (*JST*) model using data from both source and target domains [25]. These topics are used to augment the feature representations of texts from both domains. Then a sentiment classifier is trained on labeled data in source domain and applied to unseen data in target domain. The assumption behind these cross domain sentiment classification methods is that there is sufficient labeled data in source domain while the labeled data in target domain is scarce or non-existent [16]. The goal of these methods is to adapt the sentiment knowledge extracted from the labeled data of source domain to target domain. However, in this paper we assume that the labeled data in each domain is insufficient, and our goal is to train an accurate and robust sentiment classifier for each domain in a collaborative way by exploiting the sentiment relatedness among these domains.

Another line of research in multi-domain sentiment classification is sentiment classifier combination [15], [21]. For example, Li et al. proposed to combine the classification results of sentiment classifiers from different domains to make final predictions [21]. These methods integrate the sentiment knowledge from different domains at the classification stage, while in our approach the sentiment knowledge from different domains is shared at the learning stage in order to help train sentiment classifiers for each domain more accurately when labeled data is insufficient.

2.2 Multi-Task Learning

The approach proposed in this paper is based on multi-task learning [26], [27]. The aim of multi-task learning is to improve the generalization ability and prediction performance by learning multiple related tasks simultaneously and leveraging the common knowledge shared by these tasks appropriately [27]. The main difference between different multi-task learning methods lies in how they model and incorporate the task relatedness [27]. For example, Evgeniou and Pontil [26] proposed a regularized multi-task learning method. In their method, the classification models of related tasks are constrained to be similar with their average model. Thus, in this method the relatedness between various tasks is introduced by the average model and the direct relations between these tasks are not taken into consideration. Liu et al. proposed a multi-task feature learning method [28]. In their method, the classification models of related tasks are assumed to share the same sparse feature space, which is selected by group Lasso [29]. However, this assumption may not hold in multi-domain sentiment classification scenario, since different features are used to express sentiments in different domains. In trace-norm regularized multi-task learning methods [30], [31], the models from multiple related tasks are assumed to share a low-dimensional subspace. In clustered multi-task learning methods [32], [33], the group structure of models from various tasks is explored. The models of tasks from the same cluster are constrained to be more similar with each other than those from different clusters.

Different from above multi-task learning methods, in our collaborative multi-domain sentiment classification approach, the task relatedness is modeled in two aspects. First, the sentiment classification models of multiple domains share the same global component. The classification classifier of each domain can contribute to this global component and benefit from it during the learning stage. Second, each pair of domain-specific sentiment classification models is linked via their domain similarity and learned collaboratively. Similar domains are encouraged to share more sentiment information with each other than dissimilar domains. In these ways, the shared sentiment information among different domains is fully exploited and the problem of scarce labeled data is effectively alleviated.

3 Domain-Specific Sentiment Knowledge and Domain Similarity

3.1 Domain-Specific Sentiment Knowledge

Two kinds of data are combined to extract domain-specific sentiment knowledge for each domain. The first kind of data is the labeled samples, which are associated with sentiment labels and can be used to infer domain-specific sentiment expressions directly. A common observation in sentiment analysis field is that the words occurring more frequently in positive samples than negative samples usually tend to convey positive sentiment orientations, and vice versa. Thus, we can propagate the sentiment labels from documents/sentences to words to extract the domain-specific sentiment expressions. In this paper, we propose to extract the initial sentiment scores of words based on their distribution differences in positive and negative samples. More specifically, denote $\mathbf{s}^m \in \mathbb{R}^{D\times 1}$ as the sentiment word distribution of domain m, where s_w^m is the sentiment score of word w in this domain. Then s_w^m is computed using word w's association with positive sentiment label in domain m minus its association with negative sentiment label. In this paper we use *pointwise mutual information* (PMI) [35] to measure the associations between words and sentiment labels, and the sentiment score of word w in domain m is formulated as

$$\begin{aligned} s_w^m &= \mathrm{PMI}(w, posLabel_m) - \mathrm{PMI}(w, negLabel_m) \\ &= \log \frac{n(w, posLabel_m) N_m}{n(w) n(posLabel_m)} - \log \frac{n(w, negLabel_m) N_m}{n(w) n(negLabel_m)} \\ &= \log \frac{n(w, posLabel_m)\ n(negLabel_m)}{n(w, negLabel_m)\ n(posLabel_m)}, \end{aligned} \tag{1}$$

where $n(posLabel_m)$ and $n(negLabel_m)$ are the numbers of positive and negative samples in domain m respectively, and $n(w, posLabel_m)$ and $n(w, negLabel_m)$ are the frequencies of word w occurring in positive and negative samples of domain m. N_m is the number of all labeled samples in domain m. According to Eq. (1), if a sentiment word w has a higher probability to occur in positive samples than negative samples, then it will have a positive sentiment score, which indicates it tends to convey positive sentiment information in this domain.

Although labeled samples can provide direct domain-specific sentiment indications, they are usually in small size due to the high cost of manual annotation. The accuracy and coverage of the domain-specific sentiment word distribution extracted from labeled samples are usually limited. Thus, we propose to incorporate another kind of data, i.e., unlabeled samples, to refine the sentiment word distribution extracted from labeled samples. Compared with labeled samples, unlabeled samples are usually

much cheaper and easier to collect on a large scale. Although unlabeled samples are not associated with sentiment labels directly, they still contain rich domain-specific sentiment information. For example, an unlabeled review in *Kitchen* domain is "the food processor is quick and easy for making baby food." Since "quick" and "easy" are used to describe the same target in the same sentence, they probably convey the same sentiment. This is because people usually hold consistent opinions towards the same target in a short period, which is validated by social science theories such as *sentiment consistency* [36]. If we can find more cases that these two words co-occur in the same sentence, then we can infer that they tend to convey similar sentiments in this domain. Thus, in this paper we propose to extract domain-specific sentiment relations among words from the unlabeled samples based on their co-occurrence patterns.

Denote $\mathcal{S}_m$ as the set of all sentences in the unlabeled samples of domain m. If a sentence contains adversative conjunctions such as "but" and "however", we break it into clauses to make sure the sentiment expressed in each sentence of $\mathcal{S}_m$ is consistent. Then we count the frequencies of each word and each pair of words. Denote N_w^m as the frequency of word w, and N_{w_1,w_2}^m as the frequency of word pair $\{w_1, w_2\}$ in domain m. Then the contextual similarity score between words w_1 and w_2 is formulated as follows:

$$C_{w_1,w_2}^m = \log \frac{N_{w_1,w_2}^m \cdot |\mathcal{S}_m|}{N_{w_1}^m \cdot N_{w_2}^m}, \tag{2}$$

where C_{w_1,w_2}^m represents the contextual similarity score between words w_1 and w_2 in domain m, and $|\mathcal{S}_m|$ is the size of $\mathcal{S}_m$, i.e., the number of all sentences in the unlabeled samples of domain m. According to Eq. (2), if a pair of words has a higher probability to co-occur with each other in a specific domain, then they tend to share a higher contextual similarity score, which indicates that they probably convey similar sentiments in this domain. Note that the contextual similarity score defined in Eq. (2) can be negative. In this paper, we only keep the positive similarities and filter out all the negative ones.

Based on the contextual similarities among words extracted from massive unlabeled samples, we can build a domain-specific contextual similarity graph, where nodes represent words and edges represent the contextual similarities between words. In this paper, we propose to use the domain-specific contextual similarity graph to refine the initial sentiment word distribution extracted from limited labeled samples to improve its accuracy and coverage. Denote $\mathbf{s}^m \in \mathbb{R}^{D\times 1}$ as the initial sentiment word distribution of domain m, and $\mathbf{C}^m \in \mathbb{R}^{D\times D}$ as the domain-specific contextual similarity graph. Denote $\mathbf{p}^m \in \mathbb{R}^{D\times 1}$ as the final sentiment word distribution after refinement, which is computed according to

$$\begin{aligned} \arg\min_{\mathbf{p}^m} \quad & \sum_{i=1}^{D}(p_i^m - s_i^m)^2 + \frac{\theta}{2}\sum_{i=1}^{D}\sum_{j\neq i} C_{i,j}^m (p_i^m - p_j^m)^2 \\ & = \|\mathbf{p}^m - \mathbf{s}^m\|_2^2 + \theta \cdot (\mathbf{p}^m)^T \mathbf{L}^m \mathbf{p}^m, \end{aligned} \tag{3}$$

where s_i^m is the initial sentiment score of the i_{th} word in domain m, and $C_{i,j}^m$ is the contextual similarity score between the i_{th} and j_{th} words. $\mathbf{L}^m \in \mathbb{R}^{D\times D}$ is the Laplacian matrix of $\mathbf{C}^m$, and $\mathbf{L}^m = \mathbf{D}^m - \mathbf{C}^m$, where $\mathbf{D}^m$ is a diagonal matrix and $D_{i,i}^m = \sum_{j=1}^{D} C_{i,j}^m$. The quadratic graph regularization in Eq. (3) is inspired by label propagation, and the inherent connection between label propagation and quadratic cost criterion was pointed by Bengio et al. in [37]. Thus, through Eq. (3) we propagate the initial sentiment scores extracted from limited labeled samples to all sentiment expressions along the domain-specific contextual similarities, which is helpful in two aspects. First, it can propagate the sentiment scores from the popular sentiment words to the expressions not covered by labeled samples. In this way, the coverage of the domain-specific sentiment word distribution can be improved. Second, the sentiment scores of infrequent words can be refined by the sentiment scores of popular sentiment words in this domain. Thus, the accuracy of the domain-specific sentiment word distribution can be improved at the same time.

Since $\mathbf{L}^m$ in Eq. (3) is positive-semidefinite, the optimization problem in Eq. (3) is convex. We can derive that its analytical solution is $\mathbf{p}^m = (\mathbf{I} + \theta\mathbf{L}^m)^{-1}\mathbf{s}^m$, where $\mathbf{I} \in \mathbb{R}^{D\times D}$ is an identity matrix. However, since matrix inversion is time-consuming especially when dimension D is high, we propose to solve Eq. (3) using gradient descent method and use sparse matrix to store $\mathbf{L}^m$.

3.2 Domain Similarity

3.2.1 Textual Content Based Domain Similarity

The textual content based domain similarity is motivated by the observation that although different topics and opinion targets are discussed in different domains, similar domains may share many common terms. For example, in both *Smart Phone* and *Digital Camera* domains, terms like "screen", "battery", and "image" are frequently used. In contrast, the probability of two far different domains such as *Smart Phone* and *Book* sharing many common terms is low. Thus, we propose to measure the similarity between domains based on their textual content.

Inspired by the work in [38], here we select Jensen-Shannon divergence to measure the similarity of two domains based on their textual term distributions. Denote $\mathbf{d}^m \in \mathbb{R}^{D\times 1}$ and $\mathbf{d}^n \in \mathbb{R}^{D\times 1}$ as the term distribution vectors of domains m and n respectively, where D represents the dictionary size. $d_t^m \in [0, 1]$ stands for the probability of term t occurring in domain m. Then the textual content based domain similarity between domains m and n is formulated as

$$\begin{aligned} ContentSim(m,n) =& 1 - D_{JS}(\mathbf{d}^m||\mathbf{d}^n) \\ =& 1 - \frac{1}{2}(D_{KL}(\mathbf{d}^m||\overline{\mathbf{d}}) + D_{KL}(\mathbf{d}^n||\overline{\mathbf{d}})), \end{aligned} \tag{4}$$

where $\overline{\mathbf{d}} = \frac{1}{2}(\mathbf{d}^m + \mathbf{d}^n)$ is the average distribution, $D_{JS}(\cdot)$ represents Jensen-Shannon divergence, and $D_{KL}(\cdot)$ is the Kullback-Leibler divergence which is defined as:

$$D_{KL}(\mathbf{p}||\mathbf{q}) = \sum_{t=1}^{D} p(t)\log_2 \frac{p(t)}{q(t)}. \tag{5}$$

Since the base of logarithm used in Eq. (5) is 2, $D_{JS}(\mathbf{d}^m||\mathbf{d}^n) \in [0, 1]$. Thus, the range of the textual content based domain similarity defined in Eq. (4) is also $[0, 1]$.

3.2.2 Sentiment Expression Based Domain Similarity

The textual content based domain similarity introduced in previous section can measure whether two domains have

similar word usage patterns. However, high similarity in textual content does not necessarily mean that sentiment words are used in similar ways in these domains. For example, both *CPU* and *Battery* belong to electronic hardware. In *CPU* domain, the word "fast" is usually positive. For instance, "Intel Core i7 is very fast." However, in *Battery* domain, the word "fast" is frequently used as a negative word (e.g., "This battery runs out too fast"). Thus, measuring domain similarity based on sentiment expressions may be more suitable for multi-domain sentiment classification task.

Denote $\mathbf{p}^m$ and $\mathbf{p}^n$ as the sentiment word distributions of domains m and n respectively, which are extracted from both labeled and unlabeled samples according to previous section. Then the sentiment expression based domain similarity between domains m and n is defined as the cosine similarity of their sentiment word distributions

$$SentiSim(m,n) = \frac{\mathbf{p}^m \cdot \mathbf{p}^n}{\|\mathbf{p}^m\|_2 \cdot \|\mathbf{p}^n\|_2}. \tag{6}$$

Note that $SentiSim(m,n)$ defined in Eq. (6) can be negative in theory, although the probability is very small. In this paper, we constrain that domain similarities should be non-negative. Thus, if the $SentiSim$ score between a pair of domains is negative, then we set it to zero.

4 Collaborative Multi-Domain Sentiment Classification

4.1 Notations

First, we introduce several notations that will be used in following discussions. Assume there are M domains to be analyzed. Denote $\{\mathbf{X}^m \in \mathbb{R}^{N_m \times D}, \mathbf{y}^m \in \mathbb{R}^{N_m \times 1}\}$ as the labeled samples in domain m, where N_m is the number of labeled samples and D is the size of feature set. $\mathbf{x}_i^m \in \mathbb{R}^{D \times 1}$ is the transpose of the ith row of $\mathbf{X}^m$, standing for the feature vector of the ith labeled sample in domain m, and y_i^m is its sentiment label. In this paper we focus on binary sentiment classification, i.e., classifying a piece of text into positive or negative, and $y_i^m \in \{+1, -1\}$. Denote the global sentiment classifier shared by all domains as $\mathbf{w} \in \mathbb{R}^{D \times 1}$, and the domain-specific sentiment classifier of domain m as $\mathbf{w}_m \in \mathbb{R}^{D \times 1}$. We use matrix $\mathbf{W} \in \mathbb{R}^{D \times M}$ to represent all domain-specific sentiment classifiers, where the mth column of $\mathbf{W}$ stands for the domain-specific sentiment classification model of domain m, i.e., $\mathbf{W}_{\cdot,m} = \mathbf{w}_m$. Denote the loss of classifying $\mathbf{x}_i^m$ into label y_i^m under parameters $\mathbf{w}$ and $\mathbf{W}$ as $f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m})$, where f is classification loss function, which can be squared loss, log loss, hinge loss and so on.

In this paper, we propose to extract prior general sentiment knowledge from general-purpose sentiment lexicons to enhance the learning of the global sentiment classification model $\mathbf{w}$. Denote $\mathbf{p} \in \mathbb{R}^{D \times 1}$ as the general sentiment knowledge extracted from an existing sentiment lexicon. If word i is included in the sentiment lexicon and labeled as positive (or negative), then $p_i = 1$ (or -1). Otherwise, $p_i = 0$. In addition, denote $\mathbf{p}^m \in \mathbb{R}^{D \times 1}$ as the domain-specific sentiment knowledge extracted from both labeled and unlabeled data of domain m, and p_i^m is the prior sentiment score of word i. Besides, denote $\mathbf{S} \in \mathbb{R}^{M \times M}$ as the domain similarity matrix. $S_{m,n} \in [0,1]$ is the similarity between domains m and n. The diagonal elements of $\mathbf{S}$ are set to zeros.

4.2 Model

Given multiple domains to be analyzed, a small number of labeled samples in these domains, the domain similarities between them, the general sentiment knowledge extracted from general-purpose sentiment lexicons, and the domain-specific sentiment knowledge of each domain extracted from both labeled and unlabeled samples, the goal of our approach is to train accurate domain-specific sentiment classifiers for multiple domains in a collaborative way. The model of our proposed approach is formulated as an optimization problem as follows:

$$\begin{aligned}\arg\min_{\mathbf{w},\mathbf{W}} \ \mathcal{L}(\mathbf{w},\mathbf{W}) &= \sum_{m=1}^{M}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m}) - \alpha_1 \mathbf{w}^T \mathbf{p} \\ &- \alpha_2 \sum_{m=1}^{M} (\mathbf{w} + \mathbf{W}_{\cdot,m})^T \mathbf{p}^m + \beta \sum_{m=1}^{M}\sum_{n \neq m} S_{m,n} \|\mathbf{W}_{\cdot,m} - \mathbf{W}_{\cdot,n}\|_2^2 \\ &+ \lambda_1(\|\mathbf{w}\|_2^2 + \|\mathbf{W}\|_F^2) + \lambda_2(\|\mathbf{w}\|_1 + \|\mathbf{W}\|_{1,1}),\end{aligned} \tag{7}$$

where α_1, α_2, and β are non-negative regularization coefficients for general sentiment knowledge, domain-specific sentiment knowledge, and domain similarity knowledge respectively. λ_1 and λ_2 are positive coefficients of the ℓ_2- and ℓ_1-norm regularization terms. $\|\mathbf{W}\|_F$ and $\|\mathbf{W}\|_{1,1}$ are the Frobenius norm and $\ell_{1,1}$-norm of matrix $\mathbf{W}$ respectively. $\|\mathbf{W}\|_F = \sqrt{\sum_{i=1}^{D}\sum_{j=1}^{M} W_{i,j}^2}$ and $\|\mathbf{W}\|_{1,1} = \sum_{i=1}^{D}\sum_{j=1}^{M} |W_{i,j}|$. Inspired by the idea of Lasso [39], we introduce the ℓ_1-norm of the global sentiment classification model and the $\ell_{1,1}$-norm of the domain-specific sentiment classification models to control the sparsity of these models. Since not all words convey sentiment information, introducing the ℓ_1-norm of model parameters can be regarded as conducting feature selection for sentiment words. We also combine the ℓ_1-norm regularization terms with the ℓ_2-norm regularization terms motivated by elastic net regularization [40], which can improve model stability in high-dimensional problems.

According to Eq. (7), the final sentiment classifier of each domain is a linear combination of the global sentiment classification model and the domain-specific model of this domain. The global sentiment classification model is shared by all domains, and is trained on the labeled samples from all domains. The domain-specific sentiment classification model is trained using the labeled data within one domain. By splitting the sentiment classifier of each domain into a global component and a domain-specific component, we can better model the general sentiment knowledge shared by various domains and at the same time capture the domain-specific sentiment knowledge more accurately. Following many previous works in sentiment analysis area [10], [11], here we select linear classification model in our approach, which has good sentiment classification performance and excellent interpretability. Various classification loss functions can be selected for f in our model. For example, f can be squared loss (i.e., $f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m}) = ((\mathbf{w} + \mathbf{W}_{\cdot,m})^T \cdot \mathbf{x}_i^m - y_i^m)^2$), hinge loss (i.e., $f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m}) = [1 - y_i^m(\mathbf{w} + \mathbf{W}_{\cdot,m})^T \cdot \mathbf{x}_i^m]_+$), and log loss (i.e., $f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m}) = \log(1 + \exp(-y_i^m (\mathbf{w} + \mathbf{W}_{\cdot,m})^T \cdot \mathbf{x}_i^m))$). Minimizing the term $\sum_{m=1}^{M}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m})$ in Eq. (7) means that we hope the sentiment classifiers learned by our approach can classify the labeled samples in each domain as accurately as possible. In this way, we

incorporate the sentiment information in labeled samples into the sentiment classifier learning.

In Eq. (7), by the term $-\alpha_1 \mathbf{w}^T\mathbf{p}$ we constrain that the global sentiment classification model learned by our approach is consistent with the prior general sentiment knowledge extracted from sentiment lexicons. For example, since "excellent" is a positive sentiment word in many existing sentiment lexicons such as *SentiWordNet* [41], we expect that the sentiment score of "excellent" in our global sentiment classification model is also positive. If its sentiment score is negative in our model, then a penalty will be added to the objective function. In this way, the prior general sentiment knowledge extracted from general-purpose sentiment lexicons can be used to guide the learning of the global sentiment classification model in our approach. Similarly, through the term $-\alpha_2 \sum_{m=1}^{M} (\mathbf{w} + \mathbf{W}_{\cdot,m})^T \mathbf{p}^m$ we incorporate the domain-specific sentiment knowledge into sentiment classifier training. If a word w has a positive (or negative) prior sentiment score in the domain-specific sentiment knowledge of domain m, then we hope its sentiment score in the final sentiment classifier of this domain is also positive (or negative).

From Eq. (7), the domain similarity knowledge is incorporated into our model as graph regularization of the domain-specific sentiment classification models. Minimizing the term $\sum_{m=1}^{M}\sum_{n\neq m} S_{m,n}\|\mathbf{W}_{\cdot,m} - \mathbf{W}_{\cdot,n}\|_2^2$ means that if two domains share a high domain similarity with each other, then the sentiment classifiers of these two domains should be more similar. In this way, we encourage similar domains to share more sentiment information with each other than dissimilar domains.

4.3 Optimization Method

Since the objective function in our model (Eq. (7)) is convex, learning the optimal sentiment classifiers in our approach is equivalent to solving a convex optimization problem [42]. However, even if the loss function f is smooth, the optimization problem in Eq. (7) is still nonsmooth due to the ℓ_1-norm regularization of the global sentiment classification model and the $\ell_{1,1}$-norm regularization of the domain-specific sentiment classification models. In addition, the learning processes of sentiment classifiers in different domains are coupled together in our approach in order to exploit the sentiment relatedness among these domains. Thus, it is challenging to solve the optimization problem in our approach efficiently. In this paper, we introduce an accelerated algorithm based on FISTA [18] to solve the model of our approach. In addition, we propose a parallel algorithm based on ADMM [19] to train sentiment classifiers for multiple domains in a parallel way, which can further improve the efficiency of our approach when domains to be analyzed are massive. Next we will introduce them in detail.

4.3.1 An Accelerated Algorithm

In this section, we introduce the FISTA based accelerated algorithm for our approach which can be conducted on a single computing node. As mentioned before, the optimization problem in our approach is nonsmooth. Although we can use subgradient descent method to solve it, the convergence rate of subgradient method is $O(1/\sqrt{k})$ and is far from satisfactory, where k is the number of iterations. Thus, we propose to use the accelerated algorithm based on FISTA [18] to solve the optimization problem in Eq. (7) when f is smooth (such as squared loss and log loss). This algorithm has the same computational complexity as gradient method and subgradient method in each iteration, and at the same time has a convergence rate of $O(1/k^2)$, much faster than that of gradient method ($O(1/k)$) and subgradient method ($O(1/\sqrt{k})$).

Different from gradient method and subgradient method where in each iteration the current solution is computed using the last solution, in FISTA the current solution is estimated using the last two solutions and the "momentum" between them is exploited to accelerate the optimization process [18]. In each iteration of FISTA, two kinds of points are sequentially updated. The first kind of point (denoted as *search point*) is a linear combination of last two solutions, which is defined as

$$\begin{aligned}\mathbf{v}^{k+1} &= \mathbf{w}^k + a_k(\mathbf{w}^k - \mathbf{w}^{k-1}),\\ \mathbf{V}^{k+1} &= \mathbf{W}^k + a_k(\mathbf{W}^k - \mathbf{W}^{k-1}).\end{aligned} \tag{8}$$

where a_k is a positive linear combination coefficient. The second kind of point is the gradient update of the search point (denoted as *approximate point*). This point is updated using following steps. First, denote

$$\begin{aligned}g(\mathbf{w},\mathbf{W}) = &\sum_{m=1}^{M}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{w} + \mathbf{W}_{\cdot,m}) - \alpha_1 \mathbf{w}^T\mathbf{p}\\ &- \alpha_2 \sum_{m=1}^{M} (\mathbf{w} + \mathbf{W}_{\cdot,m})^T \mathbf{p}^m + \beta \sum_{m=1}^{M}\sum_{n\neq m} S_{m,n}\\ &\|\mathbf{W}_{\cdot,m} - \mathbf{W}_{\cdot,n}\|_2^2 + \lambda_1(\|\mathbf{w}\|_2^2 + \|\mathbf{W}\|_F^2).\end{aligned} \tag{9}$$

Then the approximate points in the k_{th} iteration are updated as follows:

$$\begin{aligned}\mathbf{w}^{k+1} &= S_{\lambda_2/L_k}\left(\mathbf{v}^{k+1} - \frac{1}{L_k}\frac{\partial g(\mathbf{w},\mathbf{W})}{\partial \mathbf{w}}|_{\mathbf{w}=\mathbf{v}^{k+1},\mathbf{W}=\mathbf{V}^{k+1}}\right),\\ \mathbf{W}_{\cdot,m}^{k+1} &= S_{\lambda_2/L_k}\left(\mathbf{V}_{\cdot,m}^{k+1} - \frac{1}{L_k}\frac{\partial g(\mathbf{w},\mathbf{W})}{\partial \mathbf{W}_{\cdot,m}}|_{\mathbf{w}=\mathbf{v}^{k+1},\mathbf{W}=\mathbf{V}^{k+1}}\right),\end{aligned}$$

where S is the soft thresholding operator and is defined as $S_\kappa(x) = [x-\kappa]_+ - [-x-\kappa]_+$ [43]. $\frac{1}{L_k}$ is the step size at the k_{th} iteration and its value is selected to satisfy following inequation:

$$\begin{aligned}g(\mathbf{w}^{k+1},\mathbf{W}^{k+1}) \leq\ & g(\mathbf{v}^{k+1},\mathbf{V}^{k+1}) + \frac{L_k}{2}\|\mathbf{w}^{k+1} - \mathbf{v}^{k+1}\|_2^2\\ &+ \frac{L_k}{2}\|\mathbf{W}^{k+1} - \mathbf{V}^{k+1}\|_F^2\\ &+ (\mathbf{w}^{k+1} - \mathbf{v}^{k+1})^T \frac{\partial g(\mathbf{w},\mathbf{W})}{\partial \mathbf{w}}|_{\mathbf{w}=\mathbf{v}^{k+1},\mathbf{W}=\mathbf{V}^{k+1}}\\ &+ \sum_{m=1}^{M}(\mathbf{W}_{\cdot,m}^{k+1} - \mathbf{V}_{\cdot,m}^{k+1})^T \frac{\partial g(\mathbf{w},\mathbf{W})}{\partial \mathbf{W}_{\cdot,m}}|_{\mathbf{w}=\mathbf{v}^{k+1},\mathbf{W}=\mathbf{V}^{k+1}}.\end{aligned} \tag{10}$$

Note that if the loss function f used in our approach is nonsmooth, for example, f is hinge loss, then above algorithm cannot be applied to find the optimal solution. In this case, we use subgradient descent method to solve our approach.

4.3.2 A Parallel Algorithm

When the domains to be analyzed are massive, it is inefficient to train sentiment classifiers for them on a single computing node due to the limit of memory and computational ability. Motivated by [19] and [44], here we propose a

parallel algorithm based on ADMM [19] to solve our approach more efficiently.

Assume there are G parallel nodes, such as computers and CPU cores, and we partition the domains to be analyzed into G groups. The domains in the same group are processed at the same node, and different groups are processed at different nodes. Denote $\mathcal{M}_g$ as the set of domains in group g. We keep a copy of $\mathbf{w}$ in each group and denote it as $\mathbf{v}_g$ in group g. In addition, we also keep a copy of $\mathbf{W}_{\cdot,m}$ and $\mathbf{W}_{\cdot,n}$ for each pair of domains m and n, and denote them as $\mathbf{v}_{m,n}$ and $\mathbf{v}_{n,m}$. Then the optimization problem in the model of our approach (Eq. (7)) is equivalent to

$$\begin{aligned}\min\ & \sum_{g=1}^{G}\sum_{m\in\mathcal{M}_g}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{v}_g + \mathbf{W}_{\cdot,m}) - \alpha_2\sum_{m=1}^{M}\mathbf{W}_{\cdot,m}^T\mathbf{p}^m \\ & - \mathbf{w}^T\left(\alpha_1\mathbf{p} + \alpha_2\sum_{m=1}^{M}\mathbf{p}^m\right) + \lambda_1(\|\mathbf{w}\|_2^2 + \|\mathbf{W}\|_F^2) \\ & + \beta\sum_{m=1}^{M}\sum_{n\neq m} S_{m,n}\|\mathbf{v}_{m,n} - \mathbf{v}_{n,m}\|_2^2 + \lambda_2(\|\mathbf{w}\|_1 + \|\mathbf{W}\|_{1,1}), \\ \text{s.t.}:\ & \mathbf{v}_g = \mathbf{w}, g = 1,\ldots,G \\ & \mathbf{v}_{m,n} = \mathbf{W}_{\cdot,m}, n = 1,\ldots,M.\end{aligned}$$

In ADMM, above optimization problem is further transformed into an augmented Lagrangian form as follows:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\omega},\boldsymbol{\nu},\boldsymbol{\mu}) = & \sum_{g=1}^{G}\sum_{m\in\mathcal{M}_g}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{v}_g + \mathbf{W}_{\cdot,m}) \\ & - \alpha_2\sum_{m=1}^{M}\mathbf{W}_{\cdot,m}^T\mathbf{p}^m - \mathbf{w}^T\left(\alpha_1\mathbf{p} + \alpha_2\sum_{m=1}^{M}\mathbf{p}^m\right) \\ & + \beta\sum_{m=1}^{M}\sum_{n\neq m} S_{m,n}\|\mathbf{v}_{m,n} - \mathbf{v}_{n,m}\|_2^2 + \lambda_1(\|\mathbf{w}\|_2^2 + \|\mathbf{W}\|_F^2) \\ & + \lambda_2(\|\mathbf{w}\|_1 + \|\mathbf{W}\|_{1,1}) + \frac{\rho}{2}\sum_{g=1}^{G}(\|\mathbf{w} - \mathbf{v}_g + \mathbf{u}_g\|_2^2 - \|\mathbf{u}_g\|_2^2) \\ & + \frac{\rho}{2}\sum_{m=1}^{M}\sum_{n\neq m}(\|\mathbf{W}_{\cdot,m} - \mathbf{v}_{m,n} + \mathbf{u}_{m,n}\|_2^2 - \|\mathbf{u}_{m,n}\|_2^2),\end{aligned} \tag{11}$$

where ρ is a positive penalty coefficient, $\mathbf{u}_g \in \mathbb{R}^{D\times 1}$ and $\mathbf{u}_{m,n} \in \mathbb{R}^{D\times 1}$ are scaled dual variables. $\boldsymbol{\omega}$, $\boldsymbol{\nu}$, and $\boldsymbol{\mu}$ are variable sets, introduced to represent three kinds of variables. Among them, $\boldsymbol{\omega} = \{\mathbf{w}, \mathbf{W}_{\cdot,m}, m \in [1,M]\}$, $\boldsymbol{\nu} = \{\mathbf{v}_g, \mathbf{v}_{m,n}, g \in [1,G], m,n \in [1,M]\}$ and $\boldsymbol{\mu} = \{\mathbf{u}_g, \mathbf{u}_{m,n}, g \in [1,G], m,n \in [1,M]\}$. In ADMM, these three kinds of variables are updated sequentially in each iteration, which is different from traditional multiplier methods where all variables are updated simultaneously. Specifically, in the k_{th} iteration $\boldsymbol{\omega}$, $\boldsymbol{\nu}$, and $\boldsymbol{\mu}$ are updated as follows:

$$\boldsymbol{\omega}^{k+1} = \arg\min_{\boldsymbol{\omega}}\ \mathcal{L}(\boldsymbol{\omega}, \boldsymbol{\nu}^k, \boldsymbol{\mu}^k), \tag{12}$$

$$\boldsymbol{\nu}^{k+1} = \arg\min_{\boldsymbol{\nu}}\ \mathcal{L}(\boldsymbol{\omega}^{k+1}, \boldsymbol{\nu}, \boldsymbol{\mu}^k), \tag{13}$$

$$\boldsymbol{\mu}^{k+1} = \arg\max_{\boldsymbol{\mu}}\ \mathcal{L}(\boldsymbol{\omega}^{k+1}, \boldsymbol{\nu}^{k+1}, \boldsymbol{\mu}). \tag{14}$$

We will introduce these steps one by one in detail.

Updating $\boldsymbol{\omega}^{k+1}$. According to Eqs. (11) and (12), the update processes of $\mathbf{w}$ and $\mathbf{W}_{\cdot,m}$ are separable. In addition, updating $\mathbf{w}$ can be conducted at a single node as follows:

$$\begin{aligned}\mathbf{w}^{k+1} = \arg\min_{\mathbf{w}}\ & \frac{\rho}{2}\sum_{g=1}^{G}\|\mathbf{w} - \mathbf{v}_g^k + \mathbf{u}_g^k\|_2^2 + \lambda_1\|\mathbf{w}\|_2^2 \\ & + \lambda_2\|\mathbf{w}\|_1 - \mathbf{w}^T\left(\alpha_1\mathbf{p} + \alpha_2\sum_{m=1}^{M}\mathbf{p}^m\right).\end{aligned}$$

Above optimization problem is convex but nonsmooth. Thanks to the proximal algorithm [43], we can derive an analytical solution to it as follows:

$$\mathbf{w}^{k+1} = S_{\frac{\lambda_2}{\rho G + 2\lambda_1}}\left(\textstyle\sum_{g=1}^{G}(\mathbf{v}_g^k - \mathbf{u}_g^k) + \alpha_1\mathbf{p} + \alpha_2\sum_{m=1}^{M}\mathbf{p}^m\right),$$

where $S(\cdot)$ is the soft thresholding operator [43].

$\mathbf{W}_{\cdot,m}$ is updated as follows:

$$\begin{aligned}\mathbf{W}_{\cdot,m}^{k+1} = \arg\min_{\mathbf{W}_{\cdot,m}}\ & \sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{v}_g + \mathbf{W}_{\cdot,m}) - \alpha_2\mathbf{W}_{\cdot,m}^T\mathbf{p}^m \\ & + \frac{\rho}{2}\sum_{n\neq m}\|\mathbf{W}_{\cdot,m} - \mathbf{v}_{m,n}^k + \mathbf{u}_{m,n}^k\|_2^2 \\ & + \lambda_1\|\mathbf{W}_{\cdot,m}\|_2^2 + \lambda_2\|\mathbf{W}_{\cdot,m}\|_1,\end{aligned} \tag{15}$$

where g is the group which domain m belongs to. According to Eq. (15), the updating of $\mathbf{W}_{\cdot,m}, m = 1,\ldots,M$ is separable across different domains and can be computed independently. Thus we update $\mathbf{W}_{\cdot,m}$ in parallel at different nodes. However, since the optimization problem in Eq. (15) is convex but nonsmooth, and there is no analytical solution to it, it may take many iterations to find the optimal solution. Thus, we propose to solve it using FISTA [18] algorithm when the loss function f is smooth. The detailed algorithm can be derived according to the steps in previous section. If f is not smooth, subgradient descent method is applied to solve Eq. (15).

Updating $\boldsymbol{\nu}^{k+1}$. According to Eqs. (11) and (13), the update processes of $\mathbf{v}_g$ and $\mathbf{v}_{m,n}$ are also separable and can be conducted independently. In addition, the updating of $\mathbf{v}_g$, $g = 1,\ldots,G$ is separable across different domain groups and can be conducted at different nodes. At the node which domain group $\mathcal{M}_g$ belongs to, $\mathbf{v}_g$ is updated as follows:

$$\begin{aligned}\mathbf{v}_g^{k+1} = \arg\min_{\mathbf{v}_g}\ & \sum_{m\in\mathcal{M}_g}\sum_{i=1}^{N_m} f(\mathbf{x}_i^m, y_i^m, \mathbf{v}_g + \mathbf{W}_{\cdot,m}^{k+1}) \\ & + \frac{\rho}{2}\|\mathbf{w}^{k+1} - \mathbf{v}_g + \mathbf{u}_g^k\|_2^2.\end{aligned} \tag{16}$$

According to Eq. (16), updating $\mathbf{v}_g$ also needs to solve a convex optimization problem. Similar with updating $\mathbf{W}_{\cdot,m}$, we propose to use FISTA algorithm to update $\mathbf{v}_g$ when loss function f is smooth and apply subgradient descent method when f is nonsmooth.

From Eq. (11), the updating of $\mathbf{v}_{m,n}$ is separable across different pairs of domains, and can be solved in a parallel way. $\mathbf{v}_{m,n}^{k+1}$ and $\mathbf{v}_{n,m}^{k+1}$ need to be updated jointly, and the detailed updating formulation is as follows:

$$\{\mathbf{v}_{m,n}^{k+1}, \mathbf{v}_{n,m}^{k+1}\} = \arg\min_{\mathbf{v}_{m,n},\mathbf{v}_{n,m}} 2\beta S_{m,n}\|\mathbf{v}_{m,n} - \mathbf{v}_{n,m}\|_2^2 + \frac{\rho}{2}\|\mathbf{W}_{\cdot,m}^{k+1} - \mathbf{v}_{m,n} + \mathbf{u}_{m,n}^k\|_2^2 + \frac{\rho}{2}\|\mathbf{W}_{\cdot,n}^{k+1} - \mathbf{v}_{n,m} + \mathbf{u}_{n,m}^k\|_2^2.$$

It is a convex optimization problem and its analytical solution is

$$\begin{aligned}\mathbf{v}_{m,n}^{k+1} &= \theta(\mathbf{W}_{\cdot,m}^{k+1} + \mathbf{u}_{m,n}^k) + (1-\theta)(\mathbf{W}_{\cdot,n}^{k+1} + \mathbf{u}_{n,m}^k),\\ \mathbf{v}_{n,m}^{k+1} &= (1-\theta)(\mathbf{W}_{\cdot,m}^{k+1} + \mathbf{u}_{m,n}^k) + \theta(\mathbf{W}_{\cdot,n}^{k+1} + \mathbf{u}_{n,m}^k),\end{aligned} \tag{17}$$

where $\theta = (4\beta S_{m,n} + \rho)/(8\beta S_{m,n} + \rho)$.

Updating $\boldsymbol{\mu}^{k+1}$. From Eqs. (11) and (14) we can see that the update processes of $\mathbf{u}_m$ and $\mathbf{u}_{m,n}$ are also separable. In addition, the updating of $\mathbf{u}_g$ is separable across different domain groups and can be computed as follows:

$$\mathbf{u}_g^{k+1} = \mathbf{w}^{k+1} - \mathbf{v}_g^{k+1} + \mathbf{u}_g^k. \tag{18}$$

Besides, the updating of $\mathbf{u}_{m,n}$ is separable across different domain pairs and can be conducted as follows:

$$\mathbf{u}_{m,n}^{k+1} = \mathbf{W}_{\cdot,m}^{k+1} - \mathbf{v}_{m,n}^{k+1} + \mathbf{u}_{m,n}^k. \tag{19}$$

4.4 Complexity Analysis

In this section we briefly analyze the time complexity of the two algorithms for our approach. First we study the FISTA-based accelerated algorithm that is conducted at a single node. When the loss function f is smooth and FISTA-based algorithm is applied to solve our approach, the convergence rate of the whole algorithm is $O(1/\sqrt{\epsilon})$, where ϵ is the desired accuracy [18]. In each iteration of this algorithm, the main time complexity lies in updating $\mathbf{v}$ and $\mathbf{V}$, computing the partial derivative of g with respect to $\mathbf{w}$ and $\mathbf{W}$, and updating $\mathbf{w}$ and $\mathbf{W}$. The time complexities of updating $\mathbf{v}$ and $\mathbf{V}$ are $O(D)$ and $O(MD)$ respectively. Computing the partial derivative of g with respect to $\mathbf{w}$ and $\mathbf{W}$ takes $O(ND)$ and $O(ND + M^2D)$ floating-point operations. The time complexities of updating $\mathbf{w}$ and $\mathbf{W}$ are $O(D)$ and $O(MD)$ respectively. Thus, the total time complexity of our approach with smooth loss function f is $O(\frac{(N+M^2)D}{\sqrt{\epsilon}})$. When f is nonsmooth and subgradient descent method is applied to solve our approach, then the convergence rate will become $O(1/\epsilon^2)$. In each iteration, the main time complexity lies in computing subgradients and updating model parameters, which take $O(ND + M^2D)$ and $O((1+M)D)$ floating-point operations respectively. Thus, the complexity of our approach in this case is $O(\frac{(N+M^2)D}{\epsilon^2})$.

Next we briefly analyze the time complexity of our ADMM-based parallel algorithm. The convergence rate of this algorithm is $O(1/\epsilon)$ for convex optimization problems [19]. In each iteration, the time complexity mainly lies in updating $\mathbf{w}$, $\mathbf{W}_{\cdot,m}$, $\mathbf{v}_g$, $\mathbf{v}_{m,n}$, $\mathbf{u}_g$ and $\mathbf{u}_{m,n}$. Updating $\mathbf{w}$ needs $O(GD)$ floating-point operations. The time complexities of updating $\mathbf{W}_{\cdot,m}$ and $\mathbf{v}_g$ are $O(\frac{(N+M^2)D}{G\sqrt{\epsilon}})$ and $O(\frac{ND}{G\sqrt{\epsilon}})$ respectively when f is smooth. When f is nonsmooth, their time complexities become $O(\frac{(N+M^2)D}{G\epsilon^2})$ and $O(\frac{ND}{G\epsilon^2})$. In addition, the time complexity of updating $\mathbf{v}_{m,n}$ is $O(\frac{M^2D}{G})$. Besides, the time complexities of updating $\mathbf{u}_g$ and $\mathbf{u}_{m,n}$ are $O(D)$ and $O(\frac{M^2D}{G})$ respectively. In summary, if f is smooth, then the overall time complexity of our parallel algorithm is $O(\frac{D}{\epsilon}(G + \frac{N+M^2}{G\sqrt{\epsilon}}))$. And if f is nonsmooth, then the overall time complexity becomes $O(\frac{D}{\epsilon}(G + \frac{N+M^2}{G\epsilon^2}))$.

TABLE 1
The Statistics of the Multi-Domain Sentiment Datasets

	Amazon-4				*Amazon-21*
	Book	*DVD*	*Electronics*	*Kitchen*	
Positive	1,000	1,000	1,000	1,000	24,303
Negative	1,000	1,000	1,000	1,000	24,360
Unlabeled	973,194	122,438	21,009	17,856	267,341

5 Experiments

5.1 Datasets and Experimental Settings

Two benchmark multi-domain sentiment datasets were used in our experiments. The first one is the famous Amazon product review sentiment dataset[1] (denoted as *Amazon-4*), which was collected by Blitzer et al. [13] and includes four domains, i.e., *Book*, *DVD*, *Electronics* and *Kitchen*. It is widely used in multi-domain and cross-domain sentiment analysis fields. In each domain, 1,000 positive and 1,000 negative reviews are included. The second dataset was also crawled by Blitzer et al. from Amazon. Besides the 4 domains mentioned above, it also contains another 21 domains, such as *Beauty*, *Software*, *Music* and so on. We used these 21 domains to construct another sentiment dataset and denoted it as *Amazon-21*. In addition, besides the labeled samples, a large number of unlabeled samples are included in these datasets. The detailed statistics of these multi-domain sentiment datasets are summarized in Table 1.

Several preprocessing steps were taken before experiments. Words were converted to lower cases and stopwords were removed. Following [13], unigrams and bigrams were selected as features. The sentiment lexicon used to extract prior general sentiment knowledge is SentiWordNet[2] [41]. Unless otherwise specified, in each domain of the *Amazon-4* dataset, 400 labeled samples were randomly selected as training data. And in each domain of the *Amazon-21* dataset, we randomly selected 10 percent of the labeled samples for training. Other labeled samples in these datasets were used as test data. The parameters of our approach as well as those of baseline methods were tuned using cross-validation. Each experiment was repeated ten times independently and the average results were reported. Classification accuracy is selected as performance metric following many previous works in sentiment analysis area [13], [22].

5.2 Comparison of Domain Similarity Measures

In this section we conducted experiments to figure out which one of the two domain similarity measures introduced in Section 3 is more suitable for multi-domain sentiment classification task. The experimental results on *Amazon-4* dataset are shown in Fig. 1, and the results on *Amazon-21* dataset show similar patterns. Hinge loss was used in our approach in these experiments.

1. http://www.cs.jhu.edu/~mdredze/datasets/sentiment/
2. http://sentiwordnet.isti.cnr.it/

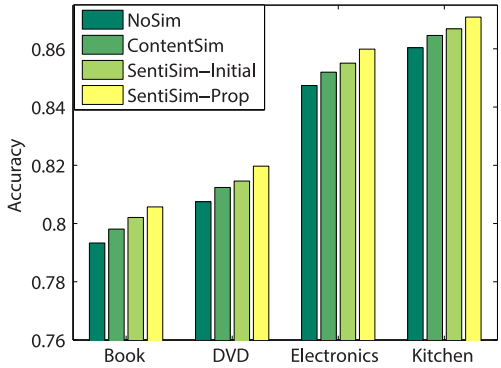


Fig. 1. The performance of our approach with different kinds of domain similarity. *NoSim*, *ContentSim*, and *SentiSim* represent the performance of our approach with no domain similarity, with textual content based domain similarity, and with sentiment expression based domain similarity respectively. The difference between *SentiSim-Initial* and *SentiSim-Prop* is that the former is based on the initial sentiment scores extracted from labeled samples, and the latter is based on the sentiment scores after propagation.

From Fig. 1, we can see that the performance of our collaborative multi-domain sentiment classification approach with sentiment expression based domain similarity is better than that with textual content based domain similarity. This result indicates that the domain similarity based on sentiment expressions can better measure the sentiment relatedness between different domains than that based on textual content in multi-domain sentiment classification task. In addition, after propagating the initial sentiment scores extracted from labeled samples along the contextual similarities mined from unlabeled data, the sentiment expression based domain similarity performs better than that without propagation. Thus, the sentiment expressions mined from scarce labeled samples can be effectively extended and refined by incorporating the useful sentiment information in massive unlabeled samples. In all following experiments, the domain similarity based on the refined sentiment expression distribution was used in our approach to measure the similarities between different domains.

5.3 Performance Evaluation

In this section we conducted experiments to evaluate the performance of our approach by comparing it with several state-of-the-art methods. The methods to be compared are:

- *LS-single*, *SVM-single*, *LR-single*: Three popular supervised sentiment classification methods, i.e., Least Squares method, Support Vector Machine and Logistic Regression, which are trained and tested in each single domain.
- *LS-all*, *SVM-all*, *LR-all*: Least Squares method, Support Vector Machine and Logistic Regression, trained on labeled samples of all domains and tested in each domain.
- *ClassifierCom*: The sentiment classifier combination method for multi-domain sentiment classification [21]. In each domain, an SVM classifier is trained on the labeled samples of this domain. Then the classifiers from multiple domains are used to make predictions for each domain. The final prediction is made by combining all prediction results of these classifiers.
- *RMTL*: The regularized multi-task learning method proposed in [26]. The model of each task is constrained to be similar with their average model.
- *MTFL21R*: Multi-task feature learning with $\ell_{2,1}$-norm regularization [28]. In this method, the models from different tasks are assumed to share the same sparse feature space selected by group Lasso.
- *MTLGraph*: Multi-task learning with graph regularization [27]. The domain similarity graph is built using the sentiment expression based domain similarities.
- *SSMTL*: Semi-supervised multi-task learning [45], which can exploit the unlabeled data in multi-task learning via parameterized neighborhood-based classification.
- *ParaVec*: The Paragraph Vector method [46]. In each domain, both labeled and unlabeled samples are used to learn document vectors[3] and logistic regression is applied to learn domain-specific sentiment classifier on these vectors.
- *CMSC-LS*, *CMSC-SVM*, *CMSC-Log*: Our proposed collaborative multi-domain sentiment classification approaches with squared loss, hinge loss, and log loss respectively.[4]

The experimental results of these methods on different datasets are summarized in Table 2.

From Table 2 we can see that our approach performs better than all baseline methods on both datasets. We further conducted two-sample one-tail t-tests to compare the results of our approach with those of baseline methods. The hypothesis testing results show that our approach can outperform baseline methods at the significance level of 0.01. Our approach performs much better than single domain sentiment classification methods, such as *LS-single*, *SVM-single* and *LR-single*. This result validates that leveraging the sentiment information in other domains can help train more accurate and robust domain-specific sentiment classifiers when labeled data is scarce. The sentiment classification methods that combine labeled samples from all domains, such as *LS-all*, *SVM-all* and *LR-all*, perform better than the single domain sentiment classification methods. However, our approach also outperforms them. This is because in these methods only one global sentiment classifier is trained and applied to different domains. Thus these methods fail to capture the specific sentiment expressions in each domain. Since different domains have different sentiment expressions, and sometimes the same expression may convey different sentiments in different domains, the global sentiment classifiers trained in these methods cannot be suitable for all domains. Our approach can tackle this problem effectively by learning a domain-specific sentiment classifier for each domain. Our approach also performs better than *ClassifierCom* method by a large margin. The main difference between *ClassifierCom* method and our approach is that in *ClassifierCom* method the sentiment knowledge from different domains is combined at the classification stage, while in our approach the sentiment knowledge from different domains is shared at the model

3. Following [46], for each document we learn two vectors based on PV-DM and PV-DBOW models respectively, each with a dimension of 300. The final vector representation of a document is the concatenation of these two vectors. The window size is set to 11.

4. The source code of our approach is available at https://github.com/EricFangzhaoWu/CMSC

TABLE 2
The Accuracies and Standard Deviations of Different Methods

Method	*Amazon-4*				*Amazon-21*
	Book	*DVD*	*Electronics*	*Kitchen*	
LS-single	0.7387 ± 0.0094	0.7436 ± 0.0097	0.7889 ± 0.0114	0.8089 ± 0.0102	0.7248 ± 0.0061
SVM-single	0.7428 ± 0.0141	0.7590 ± 0.0104	0.7932 ± 0.0113	0.8147 ± 0.0096	0.7235 ± 0.0087
LR-single	0.7449 ± 0.0129	0.7571 ± 0.0114	0.8013 ± 0.0067	0.8196 ± 0.0085	0.7327 ± 0.0076
LS-all	0.7610 ± 0.0095	0.7749 ± 0.0104	0.8211 ± 0.0085	0.8347 ± 0.0084	0.8421 ± 0.0047
SVM-all	0.7541 ± 0.0080	0.7708 ± 0.0092	0.8108 ± 0.0101	0.8269 ± 0.0097	0.8403 ± 0.0059
LR-all	0.7634 ± 0.0090	0.7795 ± 0.0078	0.8255 ± 0.0095	0.8389 ± 0.0066	0.8413 ± 0.0029
ClassifierCom	0.7543 ± 0.0174	0.7681 ± 0.0139	0.8030 ± 0.0133	0.8249 ± 0.0146	0.8199 ± 0.0068
RMTL	0.7746 ± 0.0087	0.7842 ± 0.0109	0.8298 ± 0.0083	0.8423 ± 0.0090	0.8468 ± 0.0064
MTFL21R	0.7555 ± 0.0094	0.7687 ± 0.0090	0.7990 ± 0.0089	0.8253 ± 0.0075	0.7686 ± 0.0078
MTLGraph	0.7528 ± 0.0087	0.7669 ± 0.0091	0.8114 ± 0.0062	0.8244 ± 0.0060	0.7600 ± 0.0094
SSMTL	0.7690 ± 0.0100	0.7719 ± 0.0141	0.8142 ± 0.0132	0.8312 ± 0.0062	0.8320 ± 0.0058
ParaVec	0.7716 ± 0.0120	0.7497 ± 0.0098	0.7660 ± 0.0076	0.7807 ± 0.0084	0.7797 ± 0.0038
CMSC-LS	0.7880 ± 0.0095	0.8014 ± 0.0107	0.8379 ± 0.0097	0.8591 ± 0.0070	0.8632 ± 0.0052
CMSC-SVM	0.7995 ± 0.0090	0.8179 ± 0.0087	**0.8585 ± 0.0065**	**0.8713 ± 0.0087**	0.8638 ± 0.0033
CMSC-Log	**0.8116 ± 0.0091**	**0.8208 ± 0.0086**	0.8554 ± 0.0077	0.8712 ± 0.0059	**0.8696 ± 0.0018**

Due to space limit, we only report the average results on Amazon-21 *dataset.*

learning stage. The superior performance of our approach compared with *ClassifierCom* method indicates that fusing the sentiment knowledge of multiple domains at the learning stage is more appropriate than fusing it at the classification stage. Our approach also outperforms the three state-of-the-art multi-task learning methods, i.e., *RMTL*, *MTFL21R* and *MTLGraph*. This result implies that our approach can better exploit the sentiment relatedness between multiple domains. In *RMTL*, the models of different tasks are constrained to be close to their average model. However, the pairwise relations between different domains are not considered in *RMTL* and the performance is not optimal, since sharing sentiment information between similar domains is more effective than dissimilar domains [13]. In contrast with *RMTL*, *MTLGraph* models the pairwise relations between different domains, but omits the general sentiment knowledge shared by all domains. Since general sentiment words such as "perfect" and "worst" convey consistent sentiments in different domains, the general sentiment knowledge can effectively contribute to the learning of sentiment classifiers in each domain. *MTFL21R* assumes that different tasks share the same sparse feature space. However, this assumption may not hold in multi-domain sentiment classification task since different domains usually have different sentiment expressions. Our approach can overcome above drawbacks by decomposing the sentiment classifier of each domain into two components, a global one and a domain-specific one. Through the global model, our approach can exploit the common sentiment knowledge shared by different domains and reduce the requirement for labeled data in each domain. At the same time, our approach can capture the specific sentiment expressions in each domain using the domain-specific models. Besides, by incorporating the domain similarities between different domains as graph regularization, our approach can encourage the sharing of sentiment information between similar domains. The experimental results validate that our approach is more suitable for multi-domain sentiment classification than these state-of-the-art multi-task learning methods. We also compared our approach with two famous methods which can exploit unlabeled data for text classification. The first one is *SSMTL* [45], which is a semi-supervised multi-task learning method and incorporates unlabeled data via parameterized neighborhood-based classification. The second one is Paragraph Vector method (*ParaVec*) [46], which can learn non-linear representations for documents from unlabeled data using neural network techniques. The experimental results show that our approach can outperform both of them consistently, which indicates our approach is a more appropriate way to exploit unlabeled data for multi-domain sentiment classification.

We also conducted experiments to explore the contributions of the additional sentiment knowledge besides the labeled samples to the performance improvement of our approach, such as the prior general sentiment knowledge extracted from general-purpose sentiment lexicons, the similarities between domains, and the domain-specific sentiment knowledge extracted from both labeled and unlabeled data. We evaluated the usefulness of each kind of sentiment knowledge by setting the parameters of others to zeros. For example, when we evaluate the usefulness of the prior general sentiment knowledge, we set α_2 and β to zeros. The experimental results on *Amazon-4* dataset are shown in Fig. 2.

From Fig. 2 we have two observations. First, each kind of sentiment knowledge explored in this paper can help improve the performance of our collaborative multi-domain sentiment classification approach. The experimental results validate that incorporating the prior general sentiment knowledge extracted from general-purpose sentiment lexicons can effectively help guide the learning of the global sentiment classification model. Besides, incorporating the domain-specific sentiment knowledge extracted from both labeled and unlabeled samples can enhance the learning of domain-specific sentiment models significantly. In addition, the experiments also show that incorporating the similarities between domains can help distinguish the different sentiment relatedness between different pairs of domains and encourage the sharing of sentiment information between similar domains. Second, after incorporating all the three kinds of sentiment knowledge, the performance of our approach can be further improved. This result indicates that different kinds of sentiment knowledge can collaborate with each other in the framework of our approach.

We further conducted several experiments to explore the performance of our approach with prior general sentiment

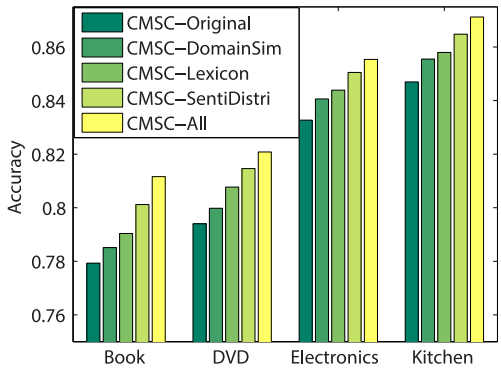


Fig. 2. The performance of our approach with no additional sentiment knowledge (*CMSC-Original*), with domain similarity knowledge (*CMSC-DomainSim*), with sentiment lexicon knowledge (*CMSC-Lexicon*), with domain-specific sentiment word distribution knowledge (*CMSC-SentiDistri*), and with all three kinds of sentiment knowledge (*CMSC-All*). The loss function used in our approach is log loss.

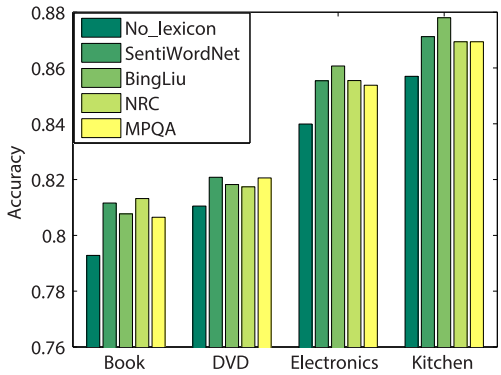


Fig. 3. The performance of our approach without prior general sentiment knowledge and with prior general sentiment knowledge extracted from different general-purpose sentiment lexicons, such as SentiWordNet, Bing Liu's lexicon, NRC emotion lexicon, and MPQA.

knowledge extracted from different sentiment lexicons, such as SentiWordNet [41], Bing Liu's lexicon [5], NRC emotion lexicon [47] and MPQA [34]. Experimental results are reported in Fig. 3.

These results show our approach can consistently improve the performance of multi-domain sentiment classification with different kinds of sentiment lexicons. It further validates the usefulness of the prior general sentiment knowledge and indicates that our approach is robust to the selection of general-purpose sentiment lexicons.

5.4 Influence of Training Data Size

In this section, we conducted experiments to explore the influence of training data size on the performance of our approach. We want to verify whether our approach can alleviate the requirement for labeled samples by training sentiment classifiers for multiple domains collaboratively. In our experiments, we varied the number of training samples in each domain from 100 to 1,000, with a step size of 100. The loss function used in our approach is log loss. The experimental results on *Amazon-4* dataset are shown in Fig. 4. We also introduce two baseline methods *LR-all* and *LR-single* for comparison, which represent *Logistic Regression* sentiment classifiers trained on labeled samples of all domains and on labeled data of each single domain respectively.

According to Fig. 4, as the number of training samples in each domain increases, the performance of our approach as well as baseline methods improves. This is intuitive because more labeled data can bring more sentiment information to sentiment classifier training. In addition, our approach can outperform baseline methods consistently no matter what the training data size is. It validates the effectiveness of our approach in improving the performance of multi-domain sentiment classification. An interesting observation from Fig. 4 is that when training data is scarce, *LR-all* performs better than *LR-single*, but when the size of training data increases, the performance of *LR-single* approaches *LR-all* quickly. This is because in *LR-all* only a single sentiment classifier is trained and applied to multiple domains. Thus it cannot capture the domain-specific sentiment expressions in each domain. Our approach can overcome this problem by training a domain-specific sentiment classifier for each domain. Thus our approach significantly outperforms *LR-all*. Although *LR-single* also builds domain-specific sentiment classifiers for each domain by training on the labeled samples of this domain, our approach outperforms it significantly. In addition, the performance improvement of our approach over *LR-single* method is more significant when the labeled data in each domain is scarce. From Fig. 4, the performance of our approach with 200 training samples is approximately the same as *LR-single* method with 900 training samples. These results validate that by training sentiment classifiers for multiple domains collaboratively and exploiting the sentiment relatedness between these domains, our approach can effectively reduce the requirement for labeled samples and significantly improve the sentiment classification performance when training data is scarce.

5.5 Time Efficiency

We conducted several experiments to explore the time complexity of our approach. The algorithms were implemented using Matlab 2014a. All experiments were conducted on a desktop computer with Intel Core i7 CPU (3.4 GHz) and 16 GB RAM. The single-node version FISTA-based accelerated algorithm was conducted on a single core of this machine, and the ADMM-based parallel algorithm was distributed across the 4 cores of this machine. The experiments were conducted on the *Amazon-21* dataset. In each experiment we randomly selected r of the labeled samples in each domain for training. We varied the ratio r from 5 to 50 percent with a

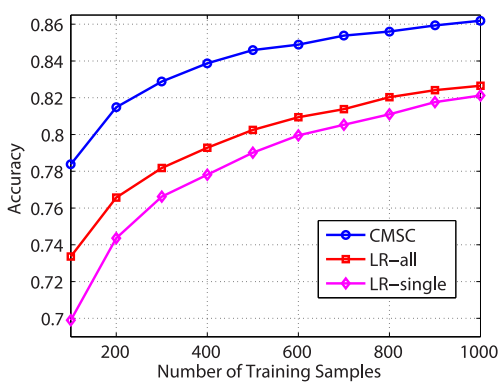


Fig. 4. The average performance of our approach and baseline methods on the four domains of *Amazon-4* dataset with different numbers of training samples. *CMSC* represents our collaborative multi-domain sentiment classification approach. *LR-all* and *LR-single* are *Logistic Regression* sentiment classifiers trained on all labeled samples and single-domain labeled samples, respectively.

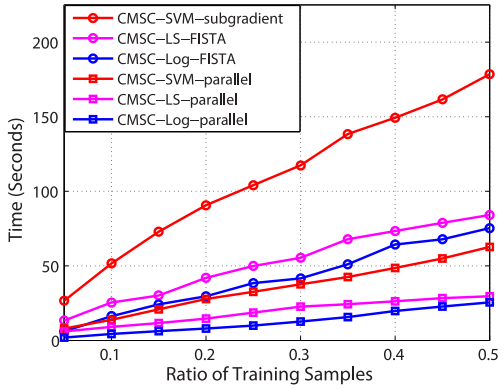


Fig. 5. The time complexity of our approach with different optimization algorithms and loss functions.

step size of 5 percent. Each experiment was repeated 10 times independently and the average results were reported in Fig. 5.

From Fig. 5 we can see that the running time of our approach with different kinds of loss functions is approximately linear with respect to size of the training data. This result validates our analysis of the time complexity in Section 4.4. Besides, our approach with log loss (*CMSC-Log*) and squared loss (*CMSC-LS*) runs much faster than that with hinge loss (*CMSC-SVM*). It validates the usefulness of the accelerated algorithm based on FISTA in improving the efficiency of our approach. In addition, the running time of the parallel algorithm is significantly less than that of single-node version optimization algorithm. It validates the effectiveness of our parallel algorithm in speeding up the learning process by training sentiment classifiers for multiple domains in parallel at different computing nodes.

5.6 Parameter Analysis

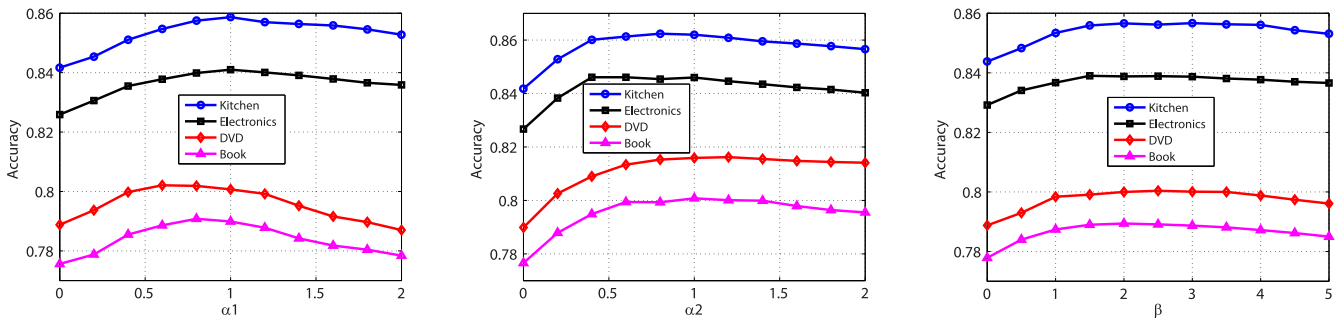
Next we explore the influence of parameter settings on the performance of our approach. The most important parameters are α_1, α_2, and β, which control the relative importance of prior general sentiment knowledge, domain-specific sentiment knowledge, and domain similarity knowledge respectively. We evaluated the influence of one parameter by fixing the others to zero. Experimental results on *Amazon-4* dataset are summarized in Fig. 6.

From Fig. 6 we can see that the influences of these parameters on the performance of our approach show similar patterns. As the parameters increase from a small value, the performance of our approach first increases and then decreases. This is because when these parameters are too small, the useful sentiment knowledge in general-purpose sentiment lexicons, domain-specific sentiment word distributions, and domain similarities is not fully exploited. Thus the performance of our approach is not optimal and improves quickly as the parameter values increase from 0. However, when these parameters become too large, the information in these three kinds of sentiment knowledge is overemphasized and the labeled samples are not fully respected. Therefore, the performance of our approach starts to decrease. Another observation from Fig. 6 is that as long as these parameters are in a moderate range, our approach can consistently improve sentiment classification performance. In practical applications, we recommend to tune the five parameters of our approach, i.e., λ_1, λ_2, α_1, α_2, and β, in a sequential manner as follows. First, we tune the value of λ_1 via cross-validation to control model complexity by fixing the other parameters to zero. Second, we fix λ_1 and tune λ_2 to control model sparsity. Then we tune the values of α_1, α_2, and β one by one in the same manner. Although this parameter tuning method is simple, we find that it works quite well for our approach in the experiments.

5.7 Case Study

In this section, we conducted several case studies to have a better understanding of how our approach works. We identified the words with the highest sentiment weights in both global and domain-specific sentiment models learned by our approach. The results on the *Amazon-4* dataset are shown in Table 3.

From Table 3 we can see that the global sentiment classification model learned by our approach can capture the general sentiment knowledge quite well. The top words in the global sentiment classification model such as "excellent", "perfect", "disappointed", and "worst" are all general sentiment words that convey strong sentiment orientation and keep consistent sentiment polarities in different domains. In addition, the top words in the domain-specific sentiment classification models can effectively capture the specific sentiment expressions in each domain. For example, "easy" is a positive word in *Kitchen* domain (e.g., "This fryer is easy to use"), while "return" is a negative word in this domain (e.g., "I will return this product"). An interesting phenomenon from Table 3 is that the same word can convey different sentiment polarities in different domains. For example, "read" is frequently used as a positive word in *Book* domain (e.g., "I've read it twice and will probably read it about ten more times before I die"). However, it is a negative word in *DVD* domain, since people often say



(a) Influence of α_1. (b) Influence of α_2. (c) Influence of β.

Fig. 6. The performance of our collaborative multi-domain sentiment classification approach (*CMSC*) with different parameter settings.

TABLE 3
The Top Sentiment Words in the Global and Domain-Specific Sentiment Classification Models

Model	Polarity	Words with top sentiment weights
General	Positive	excellent, perfect, great, best, fantastic, wonderful, awesome, a_must, amazing, outstanding, the_best, love
	Negative	waste, disappointing, worst, disappointed, poor, terrible, bad, horrible, poorly, the_worst, unfortunately, awful
Book	Positive	loved, enjoyable, wonderful, easy, a_must, enjoyed, favorite, reading_for, fun, read, entertaining, read_this
	Negative	boring, disappointing, little, bad, instead, waste, poorly, i_would, weak, writing, finish, predictable, research
DVD	Positive	enjoy, hope, back, times, loved, a_must, again, season, better_than, i_loved, superman, first, cool, hilarious
	Negative	boring, worst, bad, terrible, the_worst, awful, waste, lame, picture, dull, sucks, not_worth, book, read, easy
Electronics	Positive	perfect, memory, fast, no_problems, expected, good, highly, clear, simple, the_best, pretty, easy, works, little
	Negative	poor, terrible, back, return, not_work, returned, disappointed, died, unacceptable, junk, useless, error, flaw
Kitchen	Positive	easy, works, easy_to, love, great, little, best, so_far, well, love_this, perfectly, durable, nice, great_product
	Negative	disappointed, broken, poor, return, back, returned, horrible, waste, broke, again, junk, dangerous, cracked, rust

"Read the book, don't watch the movie" to express negative opinions in this domain. Thus, it is important to capture and model the domain-specific sentiment knowledge when analyzing the sentiments of texts from different domains. Besides, although different domains have different sentiment expressions, similar domains also share many common sentiment words. For example, in both *Electronics* and *Kitchen* domains, "esay" is a positive word and "returned" is a negative word. It indicates that measuring the similarities between different domains and encouraging the sharing of sentiment information between similar domains are helpful to learn more accurate domain-specific sentiment classifiers when labeled samples are scarce.

6 Conclusion

This paper presents a collaborative multi-domain sentiment classification approach. Our approach can learn accurate sentiment classifiers for multiple domains simultaneously in a collaborative way and handle the problem of insufficient labeled data by exploiting the sentiment relatedness between different domains. In our approach, the sentiment classifier of each domain is decomposed into two components, a global one and a domain-specific one. The global model can capture the general sentiment knowledge shared by different domains and the domain-specific models are used to capture the specific sentiment expressions of each domain. We propose to extract domain-specific sentiment knowledge from both labeled and unlabeled samples, and use it to enhance the learning of the domain-specific sentiment classifiers. Besides, we propose to use the prior general sentiment knowledge in general-purpose sentiment lexicons to guide the learning of the global sentiment classifier. In addition, we propose to incorporate the similarities between different domains into our approach as regularization over the domain-specific sentiment classifiers to encourage the sharing of sentiment information between similar domains. We formulate the model of our approach into a convex optimization problem. Moreover, we introduce an accelerated algorithm to solve the model of our approach efficiently, and propose a parallel algorithm to further improve its efficiency when domains to be analyzed are massive. Experimental results on benchmark datasets show that our approach can effectively improve the performance of multi-domain sentiment classification, and significantly outperform baseline methods.

Acknowledgments

The authors thank the reviewers for their insightful comments and constructive suggestions on improving this work. This research is supported by the National Natural Science Foundation of China (Grant nos. U1536201, U1405254, and 61472092), the National Science and Technology Support Program of China (Grant nos. 2014BAH41B00 and 2015AA020101), and the Initiative Scientific Research Program of Tsinghua University.

References

[1] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inf. Retrieval*, vol. 2, no. 1/2, pp. 1–135, 2008.
[2] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures Human Language Technol.*, vol. 5, no. 1, pp. 1–167, 2012.
[3] J. Bollen, H. Mao, and A. Pepe, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena," in *Proc. Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 17–21.
[4] B. O'Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith, "From tweets to polls: Linking text sentiment to public opinion time series," in *Proc. Int. AAAI Conf. Weblogs Social Media*, 2010, pp. 122–129.
[5] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 168–177.
[6] T. Chen, R. Xu, Y. He, Y. Xia, and X. Wang, "Learning user and product distributed representations using a sequence model for sentiment analysis," *IEEE Comput. Intell. Mag.*, vol. 11, no. 3, pp. 34–44, Aug. 2016.
[7] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu, "OpinionFlow: Visual analysis of opinion diffusion on social media," *IEEE Trans. Vis. Comput. Graph.*, vol. 20, no. 12, pp. 1763–1772, Dec. 2014.
[8] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intell. Syst.*, vol. 31, no. 2, pp. 102–107, Mar./Apr. 2016.
[9] E. Cambria, B. Schuller, Y. Xia, and B. White, "New avenues in knowledge bases for natural language processing," *Knowl.-Based Syst.*, vol. 108, no. C, pp. 1–4, 2016.
[10] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proc. ACL Conf. Empirical Methods Natural Language Process.*, 2002, pp. 79–86.
[11] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," Stanford Univ., Stanford, CA, USA, Project Rep. CS224N, pp. 1–12, 2009.
[12] F. Wu, Y. Song, and Y. Huang, "Microblog sentiment classification with contextual knowledge regularization," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2332–2338.
[13] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguistics*, 2007, vol. 7, pp. 440–447.
[14] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 513–520.

[15] S.-S. Li, C.-R. Huang, and C.-Q. Zong, "Multi-domain sentiment classification with classifier combination," *J. Comput. Sci. Technol.*, vol. 26, no. 1, pp. 25–33, 2011.
[16] L. Li, X. Jin, S. J. Pan, and J.-T. Sun, "Multi-domain active learning for text classification," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 1086–1094.
[17] G. Li, S. C. Hoi, K. Chang, W. Liu, and R. Jain, "Collaborative online multitask learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1866–1876, Aug. 2014.
[18] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, 2009.
[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[20] F. Wu and Y. Huang, "Collaborative multi-domain sentiment classification," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 459–468.
[21] S. Li and C. Zong, "Multi-domain sentiment classification," in *Proc. 46th Annu. Meeting Assoc. Comput. Linguistics*, 2008, pp. 257–260.
[22] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, "Cross-domain sentiment classification via spectral feature alignment," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 751–760.
[23] Y. Lu, M. Castellanos, U. Dayal, and C. Zhai, "Automatic construction of a context-aware sentiment lexicon: An optimization approach," in *Proc. 20th Int. Conf. World Wide Web*, 2011, pp. 347–356.
[24] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[25] Y. He, C. Lin, and H. Alani, "Automatically extracting polarity-bearing topics for cross-domain sentiment classification," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics*, 2011, pp. 123–131.
[26] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 109–117.
[27] J. Zhou, J. Chen, and J. Ye, "MALSAR: Multi-task learning via structural regularization," Arizona State Univ., Tempe, AZ, USA, 2011.
[28] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient l 2, 1-norm minimization," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 339–348.
[29] L. Meier, S. Van De Geer, and P. Bühlmann, "The group Lasso for logistic regression," *J. Roy. Statist. Soc.: Series B (Statist. Methodology)*, vol. 70, no. 1, pp. 53–71, 2008.
[30] S. Ji and J. Ye, "An accelerated gradient method for trace norm minimization," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 457–464.
[31] G. Obozinski, B. Taskar, and M. I. Jordan, "Joint covariate selection and joint subspace selection for multiple classification problems," *Statist. Comput.*, vol. 20, no. 2, pp. 231–252, 2010.
[32] Y. Zhang and D.-Y. Yeung, "A convex formulation for learning task relationships in multi-task learning," in *Proc. 26th Conf. Uncertainty Artif. Intell.*, 2010, pp. 733–742.
[33] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 702–710.
[34] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proc. Conf. Human Language Technol. Empirical Methods Natural Language Process.*, 2005, pp. 347–354.
[35] K. W. Church and P. Hanks, "Word association norms, mutual information, and lexicography," *Comput. Linguistics*, vol. 16, no. 1, pp. 22–29, 1990.
[36] R. P. Abelson, "Whatever became of consistency theory?" *Personality Social Psychology Bulletin*, vol. 9, no. 1, pp. 37–54, 1983.
[37] Y. Bengio, O. Delalleau, and N. Le Roux , "Label propagation and quadratic criterion," *Semi-Supervised Learn.*, pp. 193–216, 2006.
[38] R. Remus, "Domain adaptation using domain similarity-and domain complexity-based instance selection for cross-domain sentiment analysis," in *Proc. IEEE 12th Int. Conf. Data Mining Workshops*, 2012, pp. 717–723.
[39] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Roy. Statist. Soc. Series B*, vol. 58, pp. 267–288, 1996.
[40] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Statist. Soc.: Series B (Statist. Methodology)*, vol. 67, no. 2, pp. 301–320, 2003.
[41] S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining," in *Proc. 7th Int. Conf. Language Resources Eval.*, 2010, vol. 10, pp. 2200–2204.
[42] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[43] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optimization*, vol. 1, no. 3, pp. 123–231, 2013.
[44] D. Hallac, J. Leskovec, and S. Boyd, "Network Lasso: Clustering and optimization in large graphs," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 387–396.
[45] Q. Liu, X. Liao, H. L. Carin, J. R. Stack, and L. Carin, "Semisupervised multitask learning," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 31, no. 6, pp. 1074–1086, Jun. 2009.
[46] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, vol. 14, pp. 1188–1196.
[47] S. M. Mohammad and P. D. Turney, "Crowdsourcing a word-emotion association lexicon," *Comput. Intell.*, vol. 29, no. 3, pp. 436–465, 2013.

Fangzhao Wu received the BE degree in electronic engineering from Tsinghua University, in 2012. He is currently working toward the PhD degree in the Department of Electronic Engineering, Tsinghua University, Beijing, China. His research interests include text mining and sentiment analysis.

Zhigang Yuan received the BE degree in electronic engineering from Tsinghua University, in 2014. He is currently working toward the PhD degree in the Department of Electronic Engineering, Tsinghua University, Beijing, China. His research interests include language models, representation learning, and sentiment analysis.

Yongfeng Huang received the PhD degree in computer science and engineering from the Huazhong University of Science and Technology, in 2000. He is a professor in the Department of Electronic Engineering, Tsinghua University, Beijing, China. His research interests include cloud computing, data mining, and network security. He is a senior member of the IEEE.