Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Personalized Microblog Sentiment Classification via Multi-Task Learning

Fangzhao Wu and **Yongfeng Huang**
Tsinghua National Laboratory for Information Science and Technology,
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
wufangzhao@gmail.com, yfhuang@tsinghua.edu.cn

Abstract

Microblog sentiment classification is an interesting and important research topic with wide applications. Traditional microblog sentiment classification methods usually use a single model to classify the messages from different users and omit individuality. However, microblogging users frequently embed their personal character, opinion bias and language habits into their messages, and the same word may convey different sentiments in messages posted by different users. In this paper, we propose a personalized approach for microblog sentiment classification. In our approach, each user has a personalized sentiment classifier, which is decomposed into two components, a global one and a user-specific one. Our approach can capture the individual personality and at the same time leverage the common sentiment knowledge shared by all users. The personalized sentiment classifiers of massive users are trained in a collaborative way based on multi-task learning to handle the data sparseness problem. In addition, we incorporate users' social relations into our model to strengthen the learning of the personalized models. Moreover, we propose a distributed optimization algorithm to solve our model in parallel. Experiments on two real-world microblog sentiment datasets validate that our approach can improve microblog sentiment classification accuracy effectively and efficiently.

Introduction

Microblogging services such as Twitter provide ideal places for the public to share their opinions on various topics, such as companies, products, political events, celebrities, daily life and so on. Analyzing the sentiments in massive microblog messages has wide applications, such as user modeling, personalized recommendation, crisis management, political campaign improvement, stock market price prediction, and so on (Tumasjan et al. 2010; Bollen, Mao, and Zeng 2011; Ren and Wu 2013; Wu et al. 2014), Thus, microblog sentiment analysis has become a hot research topic in both industrial and academic fields.

Many approaches have been proposed for microblog sentiment analysis (Liu, Li, and Guo 2012; Liu et al. 2013; Wu, Song, and Huang 2015). However, in all these methods the personality of microblogging users is not taken into consideration, and the messages from different users are classified using a single sentiment classifier. When posting microblog messages, users frequently embed their individual character, opinion bias and language habits into their messages. Thus, the same word or sentence may convey different sentiments in messages posted by different users. For example, a tweet may be "The price comes down! #apple." If the author is a student and wants to buy some Apple products, then it may show a positive sentiment. However, if the author is an investor and holds a large number of Apple stocks, then this tweet probably conveys a negative sentiment. A single sentiment classifier fails to capture the individuality and opinion bias of different users, and the classification performance is usually unsatisfactory.

A simple solution to this problem is to train an individual sentiment classifier for each user using their personal data. However, the number of messages posted by one user is usually limited. Thus it is very difficult to train an accurate sentiment classifier for each single user independently. Although users tend to use some personalized sentiment expressions, they also share a lot of common sentiment knowledge. For example, many global sentiment words such as "love" and "hate" are used in the same way by different users. Thus, it is beneficial to train the personalized sentiment classifiers for massive users simultaneously and exploit the common sentiment knowledge shared by them, which can help alleviate the data sparseness problem. In addition, friends usually hold similar opinion bias towards the same targets, which is formulated as "homophyly" in social science (McPherson, Smith-Lovin, and Cook 2001) and also holds in social media (Hu et al. 2013; Kramer, Guillory, and Hancock 2014). Therefore incorporating the social relations of microblogging users may help strengthen the learning of personalized sentiment classifiers.

Motivated by the above observations, in this paper we propose a personalized sentiment classification approach. In our approach, each user has a personalized sentiment classifier, which is decomposed into two components, a global one and a user-specific one. The global classifier is trained using the messages from all the users in a collaborative way, and the user-specific classifier is trained using the data of an individual user. Thus, our approach can capture the individuality of each user and at the same time leverage the global sentiment knowledge shared by all the users to tackle the data sparseness problem. In addition, we regard the social

relations of microblogging users as the graph structure over the user-specific sentiment models and incorporate them into our model as regularization terms. Besides, we propose a distributed algorithm based on ADMM (Boyd et al. 2011) to solve the optimization problem in our approach in a parallel way. Experimental results on two real-world microblog sentiment datasets show that our approach can improve microblog sentiment classification accuracy and outperform baseline methods consistently and significantly.

Related Work

Microblog sentiment classification is a hot research topic in recent years and many approaches have been proposed (Go, Bhayani, and Huang 2009; Hu et al. 2013; Kiritchenko, Zhu, and Mohammad 2014; Wu, Song, and Huang 2015). For example, Go et al. (2009) proposed to use emoticons in tweets as noisy sentiment labels to train supervised sentiment classifiers for Twitter sentiment analysis, in order to reduce manual labeling effort. Kiritchenko et al. (2014) built two Twitter-specific sentiment lexicons based on the words' associations with emoticons and hasgtags containing sentiment words. Then they extracted sentiment features, such as the number of positive and negative words, for each tweet, and combined these features with other textual features for training and classification. Wu et al. (2015) proposed to extract contextual knowladge from massive unlabeled messages and incorpoarate it to enhance the training of microblog sentiment classifiers. However, all these methods omit the user personality.

Recently, personalized classification in social network and social media starts to attract increasing attentions (Li et al. 2010; Song et al. 2013; Li et al. 2014; Song et al. 2015). For example, Li et al. (2010) proposed a collaborative online learning method to detect sentiments in messages posted by different users, which can be regarded as an online version of the regularized multi-task learning method (Evgeniou and Pontil 2004). However, the useful social relation information is not considered in these methods. Song et al. (2015) proposed a personalized sentiment classification method based on Latent Factor Model. In their method, users and words are represented as distributions over "latent aspects" through matrix factorization. Syntactic parsing tools are utilized to obtain POS tags and dependency relations between words, in order to strengthen the learning of words' representations. However, it is difficult to explain these "latent aspects" intuitively. Besides, their method relies on microblog syntactic parsing results, which are usually inaccurate and unreliable since microblog messages are very causal, noisy and full of informal words. Different from their method, our approach does not rely on syntactic parsing results. In addition, the personalized opinion bias and individuality can be captured by our method explicitly, which is useful for user modeling and personalized recommendation (Tang et al. 2015).

Personalized Sentiment Classification

In this section, we introduce the model of our approach in detail. In addition, we propose a distributed algorithm to solve our model efficiently.

Notations

We first introduce several notations. Assume there are U users in total. Denote $\{\mathbf{x}_j^i, y_j^i | j = 1, ..., N_i\}$ as the microblog messages posted by user i and their sentiment labels, where N_i is the number of messages and D is the number of features. $\mathbf{x}_j^i \in \mathrm{R}^{D\times 1}$ represents the feature vector extracted from the j_{th} message posted by user i. In this paper we focus on binary sentiment classification and $y_j^i \in \{+1, -1\}$. Denote $\mathbf{w} \in \mathrm{R}^{D\times 1}$ as the global sentiment classification model, and $\mathbf{w}_i \in \mathrm{R}^{D\times 1}, i = 1, ..., U$ as the user-specific sentiment model for user i. Denote $f(\cdot)$ as the classification loss function. In addition, we use $\mathcal{F}$ to represent the set of social relations between users. If and only if user i and user j have social relations, $(i, j) \in \mathcal{F}$.

Model

Given the labeled messages posted by different users, their sentiment labels and the social relations between these users, our goal is to train a robust global sentiment classifier to capture the common sentiment knowledge and a user-specific sentiment classifier for each user to capture their individual personality. The model of our personalized microblog sentiment classification approach is as follows:

$$\begin{aligned}\arg\min_{\mathbf{w},\mathbf{w}_i} \sum_{i=1}^{U}\sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{w}+\mathbf{w}_i) + \lambda_1\|\mathbf{w}\|_1 \\ + \lambda_2\sum_{i=1}^{U}\|\mathbf{w}_i\|_1 + \alpha\sum_{(i,j)\in\mathcal{F}}\|\mathbf{w}_i - \mathbf{w}_j\|_2^2,\end{aligned} \tag{1}$$

where λ_1, λ_2 and α are nonnegative regularization coefficients. In our model, the personalized sentiment classifier of each user is linearly decomposed into two components, a global one and a user-specific one. The global component is trained using the data from all the users and the user-specific component is trained using data from individual user. Through this decomposition, the global sentiment model can better capture the global sentiment knowledge without the influence of individual opinion bias, and thus have higher generalization ability. This global sentiment classifier can be used to classify the messages posted by unseen users about whom we have no prior knowledge. In addition, the user-specific sentiment models can better capture the individual personality and opinion bias of each user without the interference caused by global sentiment information. Besides, the social relations between users are modeled as the graph structure over the user-specific sentiment classifiers, and incorporated into our model as regularization terms. $\sum_{(i,j)\in\mathcal{F}}\|\mathbf{w}_i - \mathbf{w}_j\|_2^2$ means that if two users have social relations with each other, then their user-specific sentiment models should be similar, motivated by the observation that friends tend to hold similar opinion bias and share similar individuality. Moreover, we introduce the l_1 norm regularization of the global sentiment model and the user-specific sentiment models to our model motivated by the idea of Lasso (Tibshirani 1996). These regularization terms can help control the model complexity and conduct feature selection for sentiment words at the

same time, since not all the words are sentiment words. The loss function in Eq. (1) can be squared loss (i.e., $f(\mathbf{x}_j^i, y_j^i, \mathbf{w}+\mathbf{w}_i) = ((\mathbf{w}+\mathbf{w}_i)^T\mathbf{x}_j^i - y_j^i)^2$), hinge loss (i.e., $f(\mathbf{x}_j^i, y_j^i, \mathbf{w}+\mathbf{w}_i) = [1 - y_j^i(\mathbf{w}+\mathbf{w}_i)^T\mathbf{x}_j^i]_+$), or log loss (i.e., $f(\mathbf{x}_j^i, y_j^i, \mathbf{w}+\mathbf{w}_i) = \log(1+\exp(-y_j^i(\mathbf{w}+\mathbf{w}_i)^T\mathbf{x}_j^i))$).

Optimization Method

Since there are hundreds of millions of users in microblogging websites such as Twitter and Weibo, the number of users can be very large when training personalized microblog sentiment classifiers. It is not possible for a single computer to finish this task due to the limit of computational ability and memory. Thus, we propose a distributed algorithm based on Alternating Direction Method of Multipliers (ADMM) (Boyd et al. 2011) to solve our model in parallel.

Assume we partition the users and their messages into M groups, and $\mathcal{U}_m$ is the set of users in group m. Each group of data is processed at an independent node, which can be a computer or a CPU core. We keep a copy of $\mathbf{w}$, named $\mathbf{v}_m$, in each group m. We also keep a copy of $\mathbf{w}_i$, named $\mathbf{v}_{i,j}$, at each edge $(i,j) \in \mathcal{F}$. Then the optimization problem in Eq. (1) can be equivalently reformulated as:

$$\begin{aligned}\text{minimize} \sum_{m=1}^{M}\sum_{i\in\mathcal{U}_m}\sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m+\mathbf{w}_i) + \lambda_1\|\mathbf{w}\|_1 \\ + \lambda_2\sum_{i=1}^{U}\|\mathbf{w}_i\|_1 + \alpha\sum_{(i,j)\in\mathcal{F}}\|\mathbf{v}_{i,j}-\mathbf{v}_{j,i}\|_2^2 \\ \text{s.t.} \quad \mathbf{w} = \mathbf{v}_m, m = 1, ..., M \\ \mathbf{w}_i = \mathbf{v}_{i,j}, i = 1, ..., N, (i,j) \in \mathcal{F}.\end{aligned}$$

In order to use ADMM, above optimization problem is further transformed into its augmented Lagrangian form:

$$\begin{aligned}\mathcal{L}(\omega,\nu,\mu) = \sum_{m=1}^{M}\sum_{i\in\mathcal{U}_m}\sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m+\mathbf{w}_i) + \lambda_1\|\mathbf{w}\|_1 \\ + \lambda_2\sum_{i=1}^{U}\|\mathbf{w}_i\|_1 + \alpha\sum_{(i,j)\in\mathcal{F}}\|\mathbf{v}_{i,j}-\mathbf{v}_{j,i}\|_2^2 \\ + \frac{\rho}{2}\sum_{m=1}^{M}(\|\mathbf{w}-\mathbf{v}_m+\mathbf{u}_m\|_2^2 - \|\mathbf{u}_m\|_2^2) \\ + \frac{\rho}{2}\sum_{(i,j)\in\mathcal{F}}(\|\mathbf{w}_i-\mathbf{v}_{i,j}+\mathbf{u}_{i,j}\|_2^2 - \|\mathbf{u}_{i,j}\|_2^2),\end{aligned}$$

where $\mathbf{u}_m \in \mathrm{R}^{D\times 1}$ and $\mathbf{u}_{i,j} \in \mathrm{R}^{D\times 1}$ are scaled dual variables, and ρ is a positive penalty coefficient. $\omega = \{\mathbf{w}, \mathbf{w}_i, i \in [1,N]\}$, $\nu = \{\mathbf{v}_m, \mathbf{v}_{i,j}, m \in [1,M], (i,j) \in \mathcal{F}\}$ and $\mu = \{\mathbf{u}_m, \mathbf{u}_{i,j}, m \in [1,M], (i,j) \in \mathcal{F}\}$. Different from traditional multiplier methods where all the variables are updated simultaneously, in ADMM these variables are updated sequentially in each iteration as follows:

$$\omega^{k+1} = \arg\min_{\omega} \ \mathcal{L}(\omega, \nu^k, \mu^k), \tag{2}$$

$$\nu^{k+1} = \arg\min_{\nu} \ \mathcal{L}(\omega^{k+1}, \nu, \mu^k), \tag{3}$$

$$\mu^{k+1} = \arg\min_{\mu} \ \mathcal{L}(\mu^{k+1}, \nu^{k+1}, \mu). \tag{4}$$

Next, we will discuss each step in detail.

Updating ω^{k+1}. The updating of $\mathbf{w}$ and $\mathbf{w}_i$ is separable. Updating $\mathbf{w}$ can be conducted at a single node:

$$\mathbf{w}^{k+1} = \arg\min_{\mathbf{w}} \ \lambda_1\|\mathbf{w}\|_1 + \frac{\rho}{2}\sum_{m=1}^{M}\|\mathbf{w}-\mathbf{v}_m^k+\mathbf{u}_m^k\|_2^2.$$

Above optimization problem is convex, but not smooth. By applying the proximal algorithm (Parikh and Boyd 2013), we have an analytical solution to it:

$$\mathbf{w}^{k+1} = S_{\lambda_1/\rho}\left(\frac{1}{M}\sum_{m=1}^{M}(\mathbf{v}_m^k - \mathbf{u}_m^k)\right), \tag{5}$$

where $S(\cdot)$ is the soft thresholding operator, which is defined as $S_\kappa(a) = [a-\kappa]_+ - [-a-\kappa]_+$.

Updating $\mathbf{w}_i$ needs to solve:

$$\begin{aligned}\mathbf{w}_i^{k+1} = \arg\min_{\mathbf{w}_i}\sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m^k+\mathbf{w}_i) + \lambda_2\|\mathbf{w}_i\|_1 \\ + \frac{\rho}{2}\sum_{(i,j)\in\mathcal{F}}\|\mathbf{w}_i-\mathbf{v}_{i,j}^k+\mathbf{u}_{i,j}^k\|_2^2.\end{aligned} \tag{6}$$

Updating $\mathbf{w}_i, i = 1, ..., N$ is separable across different users and can be calculated independently. Thus we can update $\mathbf{w}_i, i = 1, ..., N$ in a parallel way. However, since the optimization problem in Eq. (6) is convex but not smooth, and there is no analytical solution to it, the time complexity of updating $\mathbf{w}_i$ may be high if we use traditional optimization methods, such as subgradient descent method, to solve it. Here we introduce an accelerated algorithm based on FISTA (Beck and Teboulle 2009) to update $\mathbf{w}_i$ when the loss function f is smooth, for example, f is squared loss or log loss. This algorithm has the same time complexity with subgradient method at each iteration and much faster convergence rate ($O(1/T^2)$) than subgradient method ($O(1/\sqrt{T})$), where T is the number of iterations.

Next we briefly introduce the accelerated algorithm for updating $\mathbf{w}_i$. First, denote $h(\mathbf{z}) = \sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m^k + \mathbf{z}) + \frac{\rho}{2}\sum_{(i,j)\in\mathcal{F}}\|\mathbf{z}-\mathbf{v}_{i,j}^k+\mathbf{u}_{i,j}^k\|_2^2$ and $g(\mathbf{z}) = \lambda_2\|\mathbf{z}\|_1$. Then Eq. (6) can be equivalently reformulated as:

$$\mathbf{w}_i^{k+1} = \arg\min_{\mathbf{z}} \ h(\mathbf{z}) + g(\mathbf{z}). \tag{7}$$

The core idea of FISTA is exploiting the "momentum" between last two approximate solutions when estimating current solution to accelerate the iteration. It iteratively updates two kinds of points, i.e., search points and approximate points. The search point is a linear combination of last two approximate points:

$$\mathbf{s}_{t+1} = \mathbf{z}_t + a_t(\mathbf{z}_t - \mathbf{z}_{t-1}), \tag{8}$$

where $a_t > 0$ is the positive combination coefficient at the t_{th} iteration.

The approximate point is a gradient update of the latest search point:

$$\mathbf{z}_{t+1} = \mathbf{prox}_{\frac{1}{L_t}g}\left(\mathbf{s}_{t+1} - \frac{1}{L_t}h'(\mathbf{s}_{t+1})\right), \tag{9}$$

where $\frac{1}{L_t}$ is the step size at the t_{th} iteration, which is selected according to following constraint:

$$h(\mathbf{z}_{t+1}) \leq h(\mathbf{s}_{t+1}) + h'(\mathbf{s}_{t+1})(\mathbf{z}_{t+1} - \mathbf{s}_{t+1}) + \frac{L_t}{2}\|\mathbf{z}_{t+1} - \mathbf{s}_{t+1}\|_2^2. \tag{10}$$

The notation **prox** in Eq. (9) represents proximal operator (Parikh and Boyd 2013) which is defined as:

$$\mathbf{prox}_{\lambda g}(\mathbf{s}) = \arg\min_{\mathbf{x}} \; g(\mathbf{x}) + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{s}\|_2^2.$$

Since $g(\mathbf{z}) = \lambda_2\|\mathbf{z}\|_1$, Eq. (9) is equivalent to:

$$\mathbf{z}_{t+1} = S_{\lambda_2/L_t}\left(\mathbf{s}_{t+1} - \frac{1}{L_t}h'(\mathbf{s}_{t+1})\right). \tag{11}$$

The complete accelerated algorithm for updating $\mathbf{w}_i$ is summarized in Algorithm 1.

Algorithm 1 Accelerated algorithm for updating $\mathbf{w}_i$.

Input: $\{\mathbf{x}_j^i, y_j^i | j = 1, ..., N_i\}$, $\mathbf{w}_i^k$, $\mathbf{v}_m^k$, $\mathbf{v}_{i,j}^k$, $\mathbf{u}_{i,j}^k$, $\mathcal{F}$, λ_2, ρ, $\eta > 1$, L_0.
Output: $\mathbf{w}_i^{k+1}$.
Initialize $\mathbf{z}_1 = \mathbf{z}_0 = \mathbf{w}_i^k$, $t = 0$, $L = L_0$, $a_0 = 0$.
while not converge **do**
  $t = t + 1$, $a_t = (1 + \sqrt{1 + 4a_{t-1}^2})/2$.
  $\mathbf{s}_{t+1} = \mathbf{z}_t + a_t(\mathbf{z}_t - \mathbf{z}_{t-1})$.
  $g'(\mathbf{s}_{t+1}) = \sum_{j=1}^{N_i} f'(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m^k + \mathbf{s}_{t+1}) + \rho\sum_{(i,j)\in\mathcal{F}}(\mathbf{s}_{t+1} - \mathbf{v}_{i,j}^k + \mathbf{u}_{i,j}^k)$.
  $\mathbf{z}_{t+1} = S_{\lambda_2/L}\left(\mathbf{s}_{t+1} - \frac{1}{L}g'(\mathbf{s}_{t+1})\right)$.
  while Eq. (10) doesn't hold **do**
    $L = \eta L$.
    $\mathbf{z}_{t+1} = S_{\lambda_2/L}\left(\mathbf{s}_{t+1} - \frac{1}{L}g'(\mathbf{s}_{t+1})\right)$.
  end while
  $\mathbf{z} = \mathbf{z}_{t+1}$.
end while
$\mathbf{w}_i^{k+1} = \mathbf{z}$.

If the classification loss function is not smooth, for example, f is hinge loss, then we use the subgradient descent method to update $\mathbf{w}_i$.

Updating ν^{k+1}. The updating of $\mathbf{v}_m$ and $\mathbf{v}_{i,j}$ is separable and can be solved independently. In addition, updating $\mathbf{v}_m$ is separable across different user groups and can be solved in parallel. At the node of user group $\mathcal{U}_m$, $\mathbf{v}_m$ is updated by:

$$\mathbf{v}_m^{k+1} = \arg\min_{\mathbf{v}_m} \sum_{i\in\mathcal{U}_m}\sum_{j=1}^{N_i} f(\mathbf{x}_j^i, y_j^i, \mathbf{v}_m + \mathbf{w}_i^{k+1}) + \frac{\rho}{2}\|\mathbf{w}^{k+1} - \mathbf{v}_m + \mathbf{u}_m^k\|_2^2. \tag{12}$$

This optimization problem is convex. If the classification loss function f is smooth, then we can also use FISTA algorithm to solve it. The detailed description is omitted here due to space limit. Interested readers can derive this algorithm according to previous discussions. If f is not smooth, then we use the subgradient descent method to solve it.

Updating $\mathbf{v}_{i,j}$ is separable across different edges between users. Thus, they can also be solved in parallel. One thing to note is $\mathbf{v}_{i,j}^{k+1}$ and $\mathbf{v}_{j,i}^{k+1}$ must be updated at the same node:

$$\mathbf{v}_{i,j}^{k+1}, \mathbf{v}_{j,i}^{k+1} = \arg\min_{\mathbf{v}_{i,j},\mathbf{v}_{j,i}} 2\alpha\|\mathbf{v}_{i,j} - \mathbf{v}_{j,i}\|_2^2 + \frac{\rho}{2}\|\mathbf{w}_i^{k+1} - \mathbf{v}_{i,j} + \mathbf{u}_{i,j}^k\|_2^2 + \frac{\rho}{2}\|\mathbf{w}_j^{k+1} - \mathbf{v}_{j,i} + \mathbf{u}_{j,i}^k\|_2^2.$$

We can have analytical solutions to it:

$$\begin{aligned}\mathbf{v}_{i,j}^{k+1} &= \theta(\mathbf{w}_i^{k+1} + \mathbf{u}_{i,j}^k) + (1-\theta)(\mathbf{w}_j^{k+1} + \mathbf{u}_{j,i}^k),\\ \mathbf{v}_{j,i}^{k+1} &= (1-\theta)(\mathbf{w}_i^{k+1} + \mathbf{u}_{i,j}^k) + \theta(\mathbf{w}_j^{k+1} + \mathbf{u}_{j,i}^k),\end{aligned} \tag{13}$$

where $\theta = (4\alpha + \rho)/(8\alpha + \rho)$.

Updating μ^{k+1}. Similar with updating ν^{k+1}, the updating of $\mathbf{u}_m$ and $\mathbf{u}_{i,j}$ is also separable. In addition, updating $\mathbf{u}_m$ is separable across different user groups as follows:

$$\mathbf{u}_m^{k+1} = \mathbf{w}^{k+1} - \mathbf{v}_m^{k+1} + \mathbf{u}_m^k. \tag{14}$$

Updating $\mathbf{u}_{i,j}$ is separable across different edges:

$$\mathbf{u}_{i,j}^{k+1} = \mathbf{w}_i^{k+1} - \mathbf{v}_{i,j}^{k+1} + \mathbf{u}_{i,j}^k. \tag{15}$$

Complexity Analysis

Denote M as the number of parallel nodes, N and F as the total numbers of messages and user relations respectively. Since the our distributed algorithm is based on ADMM, it takes $O(1/\epsilon_1)$ iterations to reach the accuracy of ϵ_1 (Boyd et al. 2011). In each iteration of the distributed algorithm, the time complexity mainly lies in updating $\mathbf{w}$, $\mathbf{w}_i$, $\mathbf{v}_m$, $\mathbf{v}_{i,j}$, $\mathbf{u}_m$ and $\mathbf{u}_{i,j}$. Updating $\mathbf{w}$ needs $O(MD)$ floating-point operations. If the classification loss function f is smooth and Algorithm 1 is used to update $\mathbf{w}_i$, then the time complexity of updating $\mathbf{w}_i$ is $O(\frac{(N+F)D}{M\sqrt{\epsilon_2}})$, where ϵ_2 is the accuracy for updating $\mathbf{w}_i$. Otherwise, the time complexity is $O(\frac{(N+F)D}{M\epsilon_2^2})$. Similarly, if f is smooth, the time complexity of updating $\mathbf{v}_m$ is $O(\frac{ND}{M\sqrt{\epsilon_3}})$, where ϵ_3 is the accuracy. Otherwise, the time complexity of updating $\mathbf{v}_m$ is $O(\frac{ND}{M\epsilon_3^2})$. In addition, the time complexity of updating $\mathbf{v}_{i,j}$ is $O(\frac{FD}{M})$. Besides, the time complexities of updating $\mathbf{u}_m$ and $\mathbf{u}_{i,j}$ are $O(D)$ and $O(\frac{FD}{M})$ respectively. In summary, if f is smooth, then the overall time complexity of our distributed algorithm is $O(\frac{D}{\epsilon_1}((M+1) + \frac{1}{M}((\frac{1}{\sqrt{\epsilon_2}} + \frac{1}{\sqrt{\epsilon_3}})N + (2 + \frac{1}{\sqrt{\epsilon_2}})F)))$. And if f is not smooth, then the overall time complexity becomes $O(\frac{D}{\epsilon_1}((M+1) + \frac{1}{M}((\frac{1}{\epsilon_2^2} + \frac{1}{\epsilon_3^2})N + (2 + \frac{1}{\epsilon_2^2})F)))$. We can see that when the value of M is small, increasing M, i.e., incorporating more parallel nodes, can significantly reduce the overall time complexity.

Experiments

In this section we present the experimental results. Our experiments were conducted on two real-world microblog sentiment datasets. The first one is a Twitter sentiment dataset (denoted as *Twitter*). It was created by selecting top 1,000 users with highest numbers of messages in Sentiment140

sentiment corpus[1], which was crawled via Twitter API using emoticons, such as ":)" and ":(", as queries. Our Twitter dataset contains 78,135 messages, each with a sentiment label automatically assigned by emoticons. However, since emoticons contain noise, we manually labeled 10 randomly selected messages for each user as test data. The remaining messages were used for training. Since the social relations between users are not included in this dataset, we used the follower graph crawled by Kwak et al. (2010) to extract the following relations as social relations. The second dataset was crawled from Weibo[2], the most popular microblogging platform in China. This dataset is denoted as *Weibo*, and contains 296 users and all their messages and relations. Similar with *Twitter* dataset, we used the messages with emoticons as training data, and manually labeled 10 messages for each user as test data. The detailed statistics of these datasets are summarized in Table 1. Preprocessing was conducted on these datasets according to the suggestions in (Liu, Li, and Guo 2012). Unigram features were used in our experiments. The parameter values of our approach and all the baseline methods were selected via cross-validation on training data.

Table 1: Dataset Statistics. *#Train* and *#Test* represent the numbers of messages used for training and test respectively.

Dataset	#User	#Train	#Test	#Relation
Twitter	1,000	68,135	10,000	7,937
Weibo	296	22,534	2,960	766

Model Effectiveness

In this section we conducted experiments to verify the effectiveness of our model. Specifically, we want to answer two questions: 1) Whether our approach can improve microblog sentiment classification performance by modeling global sentiment knowledge and user-specific sentiment knowledge of each user simultaneously; 2) Whether incorporating social relations can improve the performance of our approach. We implemented different versions of our approach, i.e., only with global model, only with individual models, both global and individual models, global and individual models plus social regularization. We tested them on both datasets. The experimental results are shown in Figure 1.

From Figure 1 we can see that the sentiment classification accuracies of both the global model and the individual models are not very high. By combining the global sentiment model with user-specific models, our approach can improve the sentiment classification accuracy significantly. In addition, incorporating the social relations of users can further improve the performance of our approach. Thus, the experimental results validate the effectiveness of our model.

Performance Evaluation

In this section we evaluate the performance of our approach by comparing it with a series of state-of-the-art methods. The methods to be compared are: 1) LS-individual, SVM-individual and LR-individual, i.e., Least Squares method, support vector machine and Logistic Regression, trained and tested on individual user; 2) LS-global, SVM-global and LR-global, i.e., Least Squares method, support vector machine and Logistic Regression, trained and tested on all users; 3) RMTL, the regularized multi-task learning method (Evgeniou and Pontil 2004); 4) MTFL21R, multi-task feature learning with $l_{2,1}$ norm regularization (Liu, Ji, and Ye 2009); 5) MTLGraph, multi-task learning with graph regularization (Zhou, Chen, and Ye 2011); 6) LFM, personalized sentiment classification via Latent Factor Model (Song et al. 2015); 7) PMSC-LS, PMSC-SVM, PMSC-Log, our personalized microblog sentiment classification approach with squared loss, hinge loss and log loss respectively. The experimental results are summarized in Table 2.

From Table 2, our approach outperforms all the baseline methods on both datasets. We also conducted two-sample one-tail t-tests to compare the results of our approach with those of baseline methods, and the experimental re-

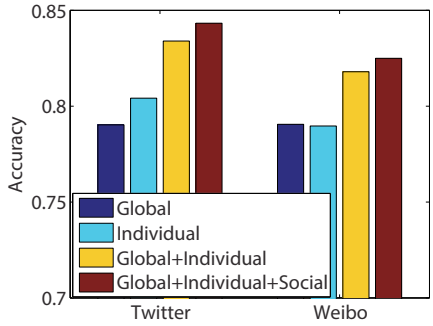

Figure 1: Effectiveness of our model.

Table 2: Performance of different methods. *50%* and *100%* represent using 50% and 100% of training data for model learning respectively.

Method	Twitter		Weibo	
	50%	100%	50%	100%
LS-individual	0.7881	0.8002	0.7621	0.7834
SVM-individual	0.7822	0.7952	0.7689	0.7885
LR-individual	0.7895	0.8042	0.7697	0.7897
LS-global	0.7865	0.7944	0.7772	0.7867
SVM-global	0.7829	0.7922	0.7809	0.7875
LR-global	0.7863	0.7903	0.7852	0.7906
RMTL	0.8100	0.8180	0.7950	0.8070
MTFL21R	0.7960	0.8090	0.7820	0.7930
MTLGraph	0.8010	0.8150	0.7810	0.8020
LFM	0.8145	0.8207	0.7945	0.8089
PMSC-LS	0.8280	0.8373	0.8110	0.8170
PMSC-SVM	0.8290	0.8393	**0.8140**	0.8220
PMSC-Log	**0.8350**	**0.8433**	0.8110	**0.8250**

[1]http://help.sentiment140.com/for-students

[2]http://www.weibo.com/

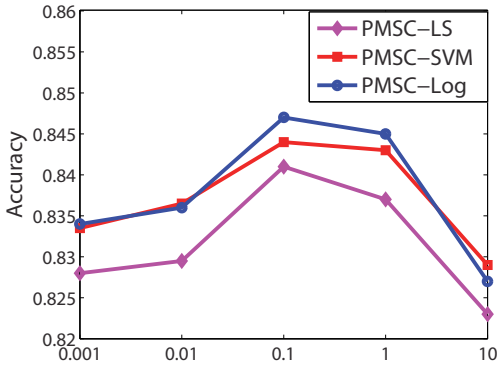

Figure 2: The influence of value of parameter α.

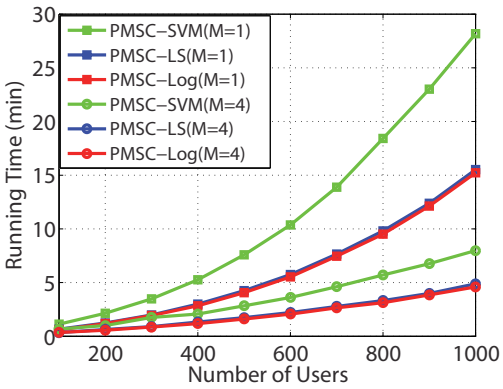

Figure 3: The running time of our approaches with different numbers of users. M represents the number of parallel nodes used in our distributed algorithm.

sults show that our approach performs significantly better than baseline methods with significance level 0.01. Both the accuracies of global methods, such as LS-global, SVM-global and LR-global, and those of individual methods, such as LS-individual, SVM-individual and LR-individual, are lower than our method. This is because the global methods fail to capture the individuality of each user, and the individual methods suffer from data sparseness. Our approach can outperform both global and individual methods because our method can capture users' individuality and at the same time exploit the common knowledge shared by different users to handle data sparseness problem. In addition, our approach performs better than LFM method. It indicates that our method is a more appropriate way to personalized sentiment classification than Latent Factor Model. Besides, our approach outperforms RMTL, MTFL21R and MTLGraph, which demonstrates that our approach is more suitable for personalized microblog sentiment classification than the state-of-the-art multi-task learning methods.

Parameter Analysis

In this section we explore the influence of parameter values on the performance of our approach. The most important parameter in our approach is α. It controls the relative importance of the social relations. We conducted experiments on both datasets and show the results on Twitter dataset in Figure 2. The results on Weibo dataset show similar patterns.

Figure 2 shows that a moderate value, such as a value in $[0.1, 1]$, is most appropriate for our approach. When α is too small, the useful information in the users' social relations is not fully used, and the performance is not optimal. Thus, the performance improves as α increases from a small value. However, when α becomes too big, the information of social relations is overemphasized, and the individuality of many users is lost. Thus the performance starts to decline.

Time Complexity

Time complexity is an important issue for our approach because there are a very large number of users in microblogging sites. In this section we explore the time complexity of our approach using experiments. We implemented our algorithm using Matlab 2014a, and conducted the experiments on a computer with Intel Core i7 CPU and 16 GB RAM. We distributed our algorithm across multiple cores of this computer. Note that our algorithm is also capable of being distributed across multiple computers. Each experiment was repeated 10 times and average results were reported. The results on Twitter dataset are shown in Figure 3.

From Figure 3 we can see that the running time of our approach using 4 cores is much less than that using only 1 core, indicating that incorporating more parallel nodes can speed up the training process. Thus, the experimental results validate the effectiveness of our distributed algorithm in reducing the time complexity of our approach by training a large number of models in parallel. In addition, the running time of our approach with squared loss and log loss is much less than that with hinge loss. This result validates the usefulness of the FISTA-based algorithms in accelerating the most time-consuming steps in our distributed algorithm.

Conclusion

This paper presents a personalized microblog sentiment classification approach. In our approach, each user has a personalized sentiment classifier, which contains two components, a global one and a user-specific one. The global sentiment classifier is used to model the general sentiment knowledge, and the user-specific sentiment classifier is used to capture the individuality of each user. Since the data of an individual user is usually sparse, we proposed to train the personalized sentiment classifiers of massive users simultaneously in a collaborative way. In addition, social relations between users are incorporated into our approach to strengthen the learning of the personalized sentiment classifiers. In order to improve the scalability and efficiency of our approach, we proposed a distributed algorithm based on ADMM to solve our approach in parallel. The experimental results on two real-world microblog sentiment datasets show that our approach can significantly improve the microblog sentiment classification accuracy.

Acknowledgments

The authors would like to thank all the anonymous reviewers for their insightful comments and suggestions on improving this paper. This research is supported by the Key Program of National Natural Science Foundation of China under Grant No. U1405254, and National Natural Science Foundation of China under Grant No. 61472092 and 61472092.

References

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* 2(1):183–202.

Bollen, J.; Mao, H.; and Zeng, X. 2011. Twitter mood predicts the stock market. *Journal of Computational Science* 2(1):1–8.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3(1):1–122.

Evgeniou, T., and Pontil, M. 2004. Regularized multi–task learning. In *KDD*, 109–117. ACM.

Go, A.; Bhayani, R.; and Huang, L. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford* 1–12.

Hu, X.; Tang, L.; Tang, J.; and Liu, H. 2013. Exploiting social relations for sentiment analysis in microblogging. In *WSDM*, 537–546.

Kiritchenko, S.; Zhu, X.; and Mohammad, S. M. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research (JAIR)* 50:723–762.

Kramer, A. D.; Guillory, J. E.; and Hancock, J. T. 2014. Experimental evidence of massive-scale emotional contagion through social networks. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)* 111(24):8788–8790.

Kwak, H.; Lee, C.; Park, H.; and Moon, S. 2010. What is twitter, a social network or a news media? In *WWW*, 591–600. ACM.

Li, G.; Hoi, S. C.; Chang, K.; and Jain, R. 2010. Microblogging sentiment detection by collaborative online learning. In *ICDM*, 893–898. IEEE.

Li, G.; Hoi, S. C.; Chang, K.; Liu, W.; and Jain, R. 2014. Collaborative online multitask learning. *IEEE Transactions on Knowledge and Data Engineering* 26(8):1866–1876.

Liu, S.; Li, F.; Li, F.; Cheng, X.; and Shen, H. 2013. Adaptive co-training svm for sentiment classification on tweets. In *CIKM*, 2079–2088. ACM.

Liu, J.; Ji, S.; and Ye, J. 2009. Multi-task feature learning via efficient l 2, 1-norm minimization. In *UAI*, 339–348. AUAI Press.

Liu, K.-L.; Li, W.-J.; and Guo, M. 2012. Emoticon smoothed language models for twitter sentiment analysis. In *AAAI*, 1678–1684.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 415–444.

Parikh, N., and Boyd, S. 2013. Proximal algorithms. *Foundations and Trends in Optimization* 1(3):123–231.

Ren, F., and Wu, Y. 2013. Predicting user-topic opinions in twitter with social and topical context. *IEEE Transactions on Affective Computing* 4(4):412–424.

Song, Y.; Lu, Z.; Leung, C. W.-k.; and Yang, Q. 2013. Collaborative boosting for activity classification in microblogs. In *KDD*, 482–490. ACM.

Song, K.; Feng, S.; Gao, W.; Wang, D.; Yu, G.; and Wong, K.-F. 2015. Personalized sentiment classification based on latent individuality of microblog users. In *IJCAI*, 2277–2283.

Tang, D.; Qin, B.; Liu, T.; and Yang, Y. 2015. User modeling with neural network for review rating prediction. In *IJCAI*, 1340–1346.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.

Tumasjan, A.; Sprenger, T. O.; Sandner, P. G.; and Welpe, I. M. 2010. Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM* 10:178–185.

Wu, Y.; Liu, S.; Yan, K.; Liu, M.; and Wu, F. 2014. Opinionflow: Visual analysis of opinion diffusion on social media. *IEEE Transactions on Visualization and Computer Graphics* 20(12):1763–1772.

Wu, F.; Song, Y.; and Huang, Y. 2015. Microblog sentiment classification with contextual knowledge regularization. In *AAAI*, 2332–2338.

Zhou, J.; Chen, J.; and Ye, J. 2011. Malsar: Multi-task learning via structural regularization. *Arizona State University*.