Ranking Tweets by Labeled and Collaboratively Selected Pairs with Transitive Closure

Shenghua Liu and **Xueqi Cheng**
Institute of Computing Technology, Chinese Academy of Sciences
No.6 Kexueyuan South Road, Haidian District
Beijing, China 100190

Fangtao Li
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043

Abstract

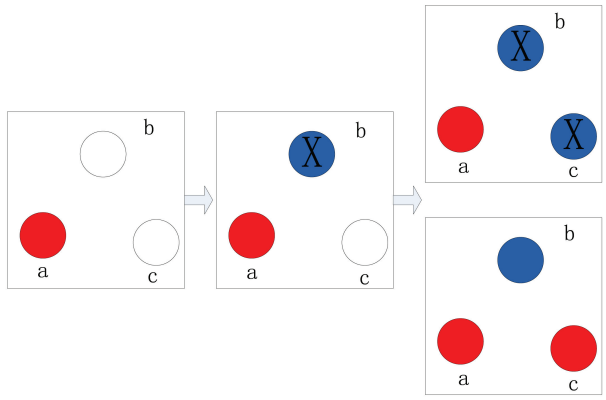
Tweets ranking is important for information acquisition in Microblog. Due to the content sparsity and lack of labeled data, it is better to employ semi-supervised learning methods to utilize the unlabeled data. However, most of previous semi-supervised learning methods do not consider the pair conflict problem, which means that the new selected unlabeled data may have order conflict with the labeled and previously selected data. It will hurt the learning performance, if the training data contains many conflict pairs. In this paper, we propose a new collaborative semi-supervised SVM ranking model (CSR-TC), selecting unlabeled data based on a dynamically maintained transitive closure graph to avoid pair conflict. We also investigate the two views of features, intrinsic and content-relevant features, for the proposed model. Extensive experiments are conducted on TREC Microblogging corpus. The results demonstrate that our proposed method achieves significant improvement, compared to several state-of-the-art models.

1 Introduction

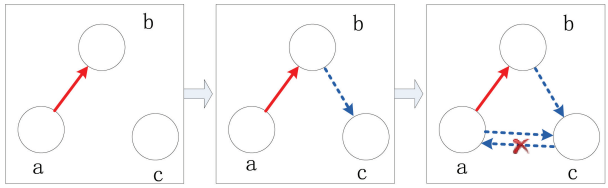
Microblog search aims to find tweets on a topic, given users' descriptions (O'Connor, Krieger, and Ahn 2010; I. et al. 2011; I., I., and J 2012). People searching tweets usually begin to read query results from the top to somewhere they get satisfied. Thus the task of our work generally helps rank the most relevant tweets on top of the page, and the irrelevant tweets at the end to avoid reading much irrelevant information. For example, a user launching the query of "2022 FIFA soccer" in TREC Microblogging corpus [1] aims to find relevant tweets about the soccer World Cup in 2022. If requiring a fully match of the query words, the results would be empty since the fragments of tweets. Otherwise, without carefully ranking them, a lot of tweets about FIFA Games for Xbox 360 are returned and mixed with relevant ones on top of result list. For example the tweet in example 1 matches two underlined words, $i.e.$ "FIFA" and "soccer", but irrelevant to the soccer World Cup.

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] http://trec.nist.gov/data/tweets/



(a) Unlabeled items selected for classification.



(b) Unlabeled pairs selected for ranking.

Figure 1: Selecting unlabeled data for semi-supervised learning.

Example 1. "played Crysis 2 Demo (360) and FIFA Soccer 10 (360) in the last 24 hours. http://raptr.com/DeXtRoXiToHeRo"

As a result, users may waste time to read much noisy messages before getting useful information about the real-life soccer game in 2022. Therefore ranking as an artificial intelligence helps people capture useful information more efficiently while searching Microblog. Especially in disaster information management, it helps make accurate and timely information available from social media before (early warning and monitoring), during, and after disasters (Sakaki, Okazaki, and Matsuo 2010).

A proper query expansion or an effective feature is a way to solve the task. Previous methods (Massoudi et al. 2011; Sakaki, Okazaki, and Matsuo 2010) separately used lan-

guage model and a classifier with query expansion for Microblog retrieval. Naveed (Naveed et al. 2011) retrieved tweets by introducing the chance a tweet may be retweeted. Abel (Abel et al. 2011) extracted facet values for adaptive faceted search. And social network properties of the authors were considered in (Jabeur, Tamine, and Boughanem 2012; Nagmoti, Teredesai, and Cock 2010). But recently, learning to ranking (LTR) (Liu 2009) achieves promising results on this task (Duan et al. 2010; Zhang, He, and Luo 2012), which includes meaningful objective functions and integrates various features. In this paper, we also follow this trend to utilize LTR framework for this task.

It is labor-intensive to manually label a large number of tweets. Therefore, we want to employ semi-supervised learning methods for LTR. Most of the semi-supervised learning methods were originally proposed for classification, such as semi-supervised Support Vector Machines (Bennett, Demiriz, and others 1999; Fung and Mangasarian 2001), a.k.a. S3VMs, and co-training framework (Blum and Mitchell 1998; Li, Hoi, and Chang 2010; Usunier, Amini, and Goutte 2011). Unlabeled data was selected iteratively and independently in those semi-supervised classification models, given the labeled and previously selected data. Take Figure 1 (a) as an example, in their iterative learning framework, when given labeled item a in red class, and unlabeled item b selected as blue class (containing a "X" mark) previously, it is then free for models to select unlabeled item c as either red or blue class for next training iteration. However, in the case of ranking, labelling data is always a process of deciding the order of a pair of data. As the example in Figure 1 (b), labeled pair (a, b) and previously selected pair (b, c) represented by directed edges are given separately indicating that item a has higher order than item b, and item b has higher order than item c. Based on the knowledge, item a has higher order than item c. So the only way to select unlabeled pair of item a and item c is in order (a, c), depending on labeled pair (a, b) and selected pair (b, c), while selecting pair (c, a) brings order conflict in next iteration of semi-supervised learning. However, as to the best of our knowledge, such a dependency for unlabeled pair selection was not considered before in the existing semi-supervised ranking models, $e.g.$, (Zhang, He, and Luo 2012; Tan et al. 2004). If selecting unlabeled pairs without considering order conflict, it may easily hurt the learning performance especially in Microblog search since of the severe lack of labeled data.

In the work, we propose order conflict constraint for semi-supervised ranking models such as S3VM ranking model and collaboratively S3VMs ranking model. With transitive closure of the graph built on labeled pairs, the order conflict constraint is normally formulated. Finally a multi-objective programming is built to minimize structural risk of labeled and collaboratively selected pairs with consideration of order conflict. The independent "optimization" step and the collaborative "selection with transitive closure" step are performed alternatively in the model learning algorithm. Queries are expended internally on the corpus, combining with the ways of "cluster document" and term co-occurrence. And two independent feature views, $i.e.$ content-relevant features and intrinsic properties of tweets, are extracted. Extensive experiments are conducted on TREC Microblogging corpus, with different parameters and portions of labeled pairs for both transductive (Joachims 1999; Zhang, He, and Luo 2012) and inductive (Tan et al. 2004) learnings. As previous works on LTR fall into three categories: the pointwise model, pairwise model and listwise model, according to different input forms(Xia et al. 2008; Yeh et al. 2007), compare to the state-of-art pairwise, listwise, and pointwise ranking models and the well-known semi-supervised algorithms, our approach achieves significant improvements on metrics.

2 Collaboratively Learning with Transitive Closure

In Microblog search on a topic, the topic is described as an unambiguous query Q_k. Queries can be expanded with words E_k. Then a collection of tweets T_k is collected for ranking, containing every tweet matching at least one of the words in query Q_k, or its expansion E_k. The relevant tweets are scattering in the list T_k. Thus the ranking problem in the paper is to *find an order of tweet list T_k for unambiguous query Q_k, such that users can catch as many relevant tweets to the topic as possible on top of the ranking result.*

In the section, to use the massive unlabeled data, we firstly introduce S3VM ranking model considering the pair conflict with transitive closure. To remit content sparseness of tweets, we use two feature views χ_1 and χ_2 to separately train two ranking models for collaborative pair selection. And an iterative learning algorithm are described.

2.1 S3VM ranking model with order conflict constraint

A supervised ranking SVM model with soft margins ξ_{ijk} is as follows.

$$\min_{\boldsymbol{\omega},\xi} \quad \frac{1}{2}||\boldsymbol{\omega}||_2 + \frac{C}{||L||_1}\sum_{L_k,k}\xi_{ijk} \tag{1}$$

$$s.t. \quad \boldsymbol{\omega}^T\Phi(Q_k,t_i) - \boldsymbol{\omega}^T\Phi(Q_k,t_j) \geq 1-\xi_{ijk}, \forall(t_i,t_j)\in L_k$$

$$\xi \succeq \mathbf{0}$$

where $\Phi(Q_k,t_i)$ is the feature vector of tweet t_i for query Q_k, and $\boldsymbol{\omega}$ is the coefficient vector for corresponding feature values. (t_i, t_j) is the labeled pairs of training data. $||\boldsymbol{\omega}||_2$ is the L-2 norm of vector $\boldsymbol{\omega}$. $||L||_1 = \sum_k |L_k|$ is the total number of training pairs for all labeled queries, and $|\cdot|$ is the size of a set. C is a constant coefficient. Let S_{ik} be the score of tweet t_i queried by Q_k, and $S_{ik} = \boldsymbol{\omega}^T\Phi(Q_k,t_i)$.

S3VM ranking model can be viewed as ranking SVM with an additional optimization term on unlabeled pairs. Unlabeled data can be viewed as a collection of ordered pairs as well. A pair of tweets t_i and t_j is said to be unlabeled, if the ranking order of t_i and t_j is not decided yet. Let U_k be the unlabeled pair set of query Q_k, and both pairs (t_i, t_j) and (t_j, t_i) are in U_k. Once either of them is selected, the other pair is also removed. It is seen later that not all the pairs in U_k are valid to select for semi-supervised learning, and we denote the final selected pairs as set $\hat{U}_k$ for query Q_k. Therefore, by introducing the hinge loss function,

$h_l(x) = \max\{0, 1-x\}$, S3VM ranking model is formulated as (2).

$$\min_{\boldsymbol{\omega}} \quad \frac{1}{2}||\boldsymbol{\omega}||_2 + \frac{C}{||L||_1} \sum_{(t_i,t_j)\in L_k,k} h_l(\boldsymbol{\omega}^T \Delta_{ijk}) \qquad (2)$$
$$+ \frac{C'}{||\hat{U}||_1} \sum_{(t_i,t_j)\in \hat{U}_k,k} h_l(\boldsymbol{\omega}^T \Delta_{ijk})$$

where we have defined the shorthand $\Delta_{ijk} \equiv \Phi(Q_k, t_i) - \Phi(Q_k, t_j)$, and $||\hat{U}||_1 = \sum_k |\hat{U}_k|$ is the sum of selected pairs for all queries. C' is scaling constant for the loss of unlabeled pairs.

Unfortunately, deciding the selected pair set $\hat{U}_k$ is non-trivial. On one hand, those unlabeled pairs $(t_i, t_j) \in \hat{U}_k$ are required to satisfy $S_{ik} - S_{jk} > 0$ in the ranking model. We then define boolean variables α_{ijk} as equation (3).

$$\begin{aligned} \alpha_{ijk} &= \begin{cases} 1, & \boldsymbol{\omega}^T \Delta_{ijk} > 0 \\ 0, & \text{otherwise.} \end{cases} \\ &= \max\{0, \text{sign}(\boldsymbol{\omega}^T \Delta_{ijk})\}. \end{aligned} \qquad (3)$$

On the other hand, the selected pairs of set $\hat{U}_k$ should avoid order conflict discussed in Figure 1 (b). We can build a directed graph $G(\hat{U}_k \cup L_k)$ on the pairs from $\hat{U}_k \cup L_k$, with all tweets in T_k as vertices and each pair as a directed edge pointing from the higher order tweet to the lower one. Thus directed graph $G(\hat{U}_k \cup L_k)$ is required acyclic to avoid order conflict. Furthermore, based on the truth that there are no order conflicts in labeled pair set L_k, the order conflict constraint for selected pairs can be generally described as constraint (4)

$$\forall (t_u, t_v) \in \hat{U}_k, \text{such that } \neg\exists P_{vu} \subseteq G(\hat{U}_k \cup L_k) \qquad (4)$$

where P_{vu} is a directed path from vertex t_v to vertex t_u.

However, it is time-consuming to check all the pair candidates and possible paths between them from the selected and labeled pair set, according to constraint (4). Therefore, we introduce a concept of "transitive closure", which is often used in Graph Theory.

Definition 1. *Graph $\dot{G}$ is said to be transitive closure, if and only if there is a directed path from vertex t_u to t_v, then there exists a directed edge from vertex t_u to t_v.*

Let graph $G(L_k)$ be built on all the tweets in T_k as vertices and $\forall (t_i, t_j) \in L_k$ as directed edges. With the transitive property, we can add a minimum number of pairs that make graph $G(L_k)$ transitive closure, keeping the essential ranking problem unchanged. The transitive closure graph is denoted as $\dot{G}(L_k)$. The newly added pairs $(t'_i, t'_j) \in \dot{G}(L_k)$ and its reverse pairs (t'_j, t'_i) are removed from unlabeled set U_k. And the rest of unlabeled set is denoted as $U'_k \subseteq U_k$. Let graph $G(U'_k)$ be built on all the tweets in list T_k as vertices, and pairs in U'_k as directed edges. Let $P_{v_1 v_n} = \{t_{v_1}, t_{v_2}, \cdots, t_{v_n}\}$ be a path of length $n-1$ in graph $G(U'_k)$. Hence with transitive closure graph $\dot{G}(L_k)$, the order conflict constraint (4) is consequently simplified as follows.

$$\prod_{i=1}^{n-1} \theta_{v_i v_{i+1} k} = 0, \quad \theta_{v_i v_{i+1} k} \in \{0, 1\} \qquad (5)$$
$$\forall (t_{v_n}, t_{v_1}) \in \dot{G}(L_k) \text{ and } \forall P_{v_1 v_n} \subseteq G(U'_k), \quad n = 3, 4, \cdots, |U'_k|.$$

At last, the selection set $\hat{U}_k$ is determined by $\theta_{ijk} \cdot \alpha_{ijk}$. The constraint (5) avoids any directed path from t_{v_1} to t_{v_n}, which keeps graph $G(\hat{U}_k \cup L_k)$ acyclic, since edge (t_{v_n}, t_{v_1}) in transitive closure graph $\dot{G}(L_k)$. Therefore, the S3VM ranking model considering order conflict constraints for all queries is formulated as (6).

$$\min_{\boldsymbol{\omega}} \quad \frac{1}{2}||\boldsymbol{\omega}||_2 + \frac{C}{||L||_1} \sum_{L_k,k} h_l(\boldsymbol{\omega}^T \Delta_{ijk}) \qquad (6)$$
$$+ \frac{C'}{\sum \alpha_{ijk}\theta_{ijk}} \sum_{U'_k,k} \alpha_{ijk}\theta_{ijk} h_l(\boldsymbol{\omega}^T \Delta_{ijk}) + \frac{C''}{\sum \alpha_{ijk}\theta_{ijk}}$$
$$s.t. \quad \alpha_{ijk} = \max\left\{0, \text{sign}(\boldsymbol{\omega}^T \Delta_{ijk})\right\}, \forall (t_i, t_j) \in U'_k$$
$$\text{order conflict constraint } (5) \text{ for } \boldsymbol{\theta}.$$

where C'' is scaling constant for encouraging the model to to choose as many validated pairs as possible.

2.2 Collaboratively ranking with order conflict constraint

As discussed in (Li and Zhou 2010), without carefully choosing the unlabeled data, the selected unlabeled pairs may degenerate learning performance, and the ranking ability may be even worse than only using the labeled data. Thus we try to choose the most confident labeled pairs instead of all legal pairs, to reduce the risk of performance degeneration. The confidence of $\forall$ pair $(t_i, t_j) \in U'_k$ is intuitively measured by a function of the difference between their ranking scores, *i.e.* $\boldsymbol{\omega}^T \Delta_{ijk}$. Given δ as the confidence threshold, $\alpha_{ijk} = \max\left\{0, \text{sign}(\boldsymbol{\omega}^T \Delta_{ijk} - \delta)\right\}$, where $\delta \in [0, 1)$.

Furthermore, splitting features into two views χ_1 and χ_2 , and selecting the unlabeled pairs collaboratively are also helpful to reduce the risk of introducing noisy pairs. In the primitive co-training framework, two learning models are trained using two split feature views. The unlabeled pairs selected according to both ranking models are added into labeled data for the next co-training iteration. Though RSCF (Tan et al. 2004) in ranking webpages selected unlabeled pairs as either of ranking models felt confident, the "agreement" strategy (Collins and Singer 1999) is adopted to further guarantee the quality of selected pairs, since of the extreme sparse features of tweets. So only those unlabeled pairs that both rankers agree with are collaboratively selected for optimization.

Let $\Delta^{(1)}_{ijk}$ and $\Delta^{(1)}_{ijk}$ be the feature differences of pair (t_i, t_j) separately on two feature views χ_1 and χ_2 for query Q_k. We use α_{ijk} and α'_{ijk} as the boolean indicators of

confidence for two rankers respectively. The selected pairs in set $\hat{U}_k$ are then decided by $\theta_{ijk}\alpha_{ijk}\alpha'_{ijk}$, and define $\mu_{ijk} = \theta_{ijk}\alpha_{ijk}\alpha'_{ijk}$. Finally, the collaboratively S3VMs ranking model is formulated as follows.

$$\min_{\boldsymbol{\omega}_{(1)},\boldsymbol{\omega}_{(2)}} \begin{pmatrix} \frac{1}{2}||\boldsymbol{\omega}_{(1)}||_2 + \frac{C}{||L||_1}\sum_{L_k,k} h_l(\boldsymbol{\omega}_{(1)}^T\Delta_{ijk}^{(1)}) \\ +\frac{C'}{\sum\mu_{ijk}}\sum_{U'_k,k}\mu_{ijk}h_l(\boldsymbol{\omega}_{(1)}^T\Delta_{ijk}^{(1)}) + \frac{C''}{\sum\mu_{ijk}} \\ \frac{1}{2}||\boldsymbol{\omega}_{(2)}||_2 + \frac{C}{||L||_1}\sum_{L_k,k} h_l(\boldsymbol{\omega}_{(2)}^T\Delta_{ijk}^{(2)}) \\ +\frac{C'}{\sum\mu_{ijk}}\sum_{U'_k,k}\mu_{ijk}h_l(\boldsymbol{\omega}_{(2)}^T\Delta_{ijk}^{(2)}) + \frac{C''}{\sum\mu_{ijk}} \end{pmatrix} \quad (7a)$$

$$s.t. \quad \alpha_{ijk} = \max\left\{0, \text{sign}(\boldsymbol{\omega}_{(1)}^T\Delta_{ijk}^{(1)} - \delta)\right\}, \quad (7b)$$

$$\alpha'_{ijk} = \max\left\{0, \text{sign}(\boldsymbol{\omega}_{(2)}^T\Delta_{ijk}^{(2)} - \delta)\right\}, \quad (7c)$$

$$\forall(t_i,t_j) \in U'_k.$$

order conflict constraint (5) for $\boldsymbol{\theta}$.

The coefficients $\boldsymbol{\omega}_{(1)}$ and $\boldsymbol{\omega}_{(2)}$ are corresponding to two feature views. They have the same dimension as the whole feature space. And those coefficients in $\boldsymbol{\omega}_{(\cdot)}$ are set to be 0 while the corresponding features are not in the features view.

2.3 Model learning

Since the collaboratively ranking model (7) is nonconvex and multiobjective, we use an iterative and heuristic algorithm to learn the model, by fixing boolean vectors $\boldsymbol{\mu}$ ($i.e.$ the selection of unlabeled pairs) for structural risk minimization and coefficient vector $\boldsymbol{\omega}_{(1)}, \boldsymbol{\omega}_{(2)}$ for unlabeled pair selection alternatively. Thus there are two steps: the optimization step and the selection step with transitive closure in our learning algorithm.

In the optimization step: $\boldsymbol{\mu}$ is fixed. The structural risk of labeled and collaboratively selected pairs $\hat{U}_k$ is minimized for every query Q_k. The model (7) is reduced to two independent SVMs optimizations, since of the assumption that two views of features χ_1 and χ_2 are independent. With solving the general SVMs optimization problems, the model parameters $\boldsymbol{\omega}_{(1)}$ and $\boldsymbol{\omega}_{(2)}$ are estimated.

In the selection step with transitive closure: $(\boldsymbol{\omega}_{(1)}, \boldsymbol{\omega}_{(2)})$ is fixed. $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}'$ are calculated as constraints (7b) and (7b) respectively. We notice that the pairs in U'_k count for optimization target (7a) only if $\alpha.\alpha'_. = 1$ simultaneously. Thus we only need to consider those edges in graph $G(U'_k)$ such that both confidence indicators $\alpha.$ and $\alpha'_.$ are nonzero. Furthermore, in order to avoid find all the validated paths in graph $G(U'_k)$ in each iteration, a transitive closure graph $\dot{G}(\hat{U}_k \cup L_k)$ is dynamically maintained on labeled and previously selected pairs. We heuristically and sequentially add unlabeled pairs into $\hat{U}_k$ from higher confident pairs to lower one. Once a pair is added, the transitive closure graph $\dot{G}(\hat{U}_k \cup L_k)$ are maintained by adding the minimum of extra pairs, and the added pairs are not withdrew anymore. Therefore, the only thing we do to decide θ_{ijk} of an unlabeled pair (t_i, t_j) is to check if its reverse pair (t_j, t_i) is in graph $\dot{G}(\hat{U}_k \cup L_k)$. At last, selection indicators $\boldsymbol{\mu}$ is calculated accordingly.

In practice, we limit the number of confident pairs selected in each iteration to keep out some noisy data especially at the early iterations of our algorithm. The algorithm (Yellin 1993) is used to incrementally maintains the transitive closure. Finally, the iteration terminates until no confident pairs can be added for all queries with agreement.

2.4 Query Expansion and Two-view Features on Tweets

At first, we leverage two Pseudo-Relevance Feedback methods for internal expansions for query Q_k. One is that we assume that the retrieved top tweets with the highest ranking scores are more relevant. By viewing them as a "document", we estimate each word's weight by the following formula.

$$W_k^f(\tau) = idf(\tau) \cdot \frac{\sum_{i\in\boldsymbol{\pi}} S_{ik} \cdot tf(t_i,\tau)}{\sum_{i\in\boldsymbol{\pi}} S_{ik}} \quad (8)$$

where $\boldsymbol{\pi} = \pi(1), \pi(2), \ldots, \pi(K)$ is the index set of the top K returned tweets, $tf(t_i,\tau)$ is the term frequency of τ in tweet t_i, and $idf(\tau)$ is the inverse document frequency in the corpus. Besides, we use term co-occurrence in the corpus as a second method to choose expansion words, and their weights are estimated using the following formula.

$$W_k^c(\tau) = \frac{1}{\sum_{i\in\boldsymbol{\pi}} S_{ik}} \sum_{i\in\boldsymbol{\pi}} S_{ik} \cdot \frac{tc_k(\tau)}{\sum_{o\in t_i} tc_k(o)} \quad (9)$$

where $tc_k(\tau)$ indicates the term τ co-occurrence with query terms of Q_k, and it is normalized by $\sum_{o\in t_i} tc_k(o)$. At last, those expansions to query Q_k form vector E_k with weight $W_k^f(\tau) \cdot W_k^c(\tau)$, which is used as a weighting scalar for feature calculation.

We explore content-relevant features of tweet t_i denoted separately as R_{ik} and RE_{ik} for query Q_k and its expansion E_k, and the tweets intrinsic properties denoted as I_i. Totally, there are 51 features extracted as discussed in (Zhu et al. 2012). Due to the space constraint, we only introduce some of the features. In features $\boldsymbol{R}$, we use Boolean model, vector space model (VSM), Okapi BM25[2] model, and language model for IR (lmir) (Ponte and Croft 1998) for the scores of the retrieval models. Meanwhile, three smoothing methods are used for lmir respectively, $i.e.$ Dirichlet Prior, Jelinek-Mercer and Absolute Discounting. Features in $\boldsymbol{R}E$ are the corresponding features for query expansion. In the set of intrinsic features $\boldsymbol{I}$, we calculate ratios of out-of-vocabulary words and unique word ratio in a tweet, properties of URLs, hash tags and "@", and etc. At last, we split the features in $\boldsymbol{R}$, $\boldsymbol{R}E$ and $\boldsymbol{I}$ into two views of χ_1 and χ_2: content-relevant view of features $\boldsymbol{R}$ and $\boldsymbol{R}E$, and intrinsic view of features $\boldsymbol{I}$.

3 Experiments

We use TREC Microblogging corpus from the 2012 release. There are about 113,928 labeled tweets on 109 topic queries,

[2]http://en.wikipedia.org/wiki/Okapi_BM25

out of 7,443,387 tweets totally. The tweets are officially labeled as highly relevant (2), relevant (1), and irrelevant (0) as our ground truth. We use the first 49 queries as training queries, and the rest of 60 as testing queries, according to the official setting of TREC 2012. In the experiment, only a part of data from training queries are viewed as labeled for learning to rank. Two kinds of sampling methods are used to generate the labeled data and unlabeled data from training queries. One is "pairwise sampling": we generate ordered pair sets L_k^*, according to the three relevance levels, where $k = 1, 2, \ldots, 49$, and get totally 2,009,384 pairs. Different ratio of pairs are randomly sampled from L_k^* as labeled data L_k, and the rest are viewed as unlabeled for training. Such training data actually are useful for integrating with active learning (Tong and Koller 2001; Yu 2005). The other is "pointwise sampling": we randomly sample a specific percent of 3 relevance levels of tweets instead of pairs from training queries, and the rest tweets are viewed as unlabeled. In such a traditional way, we can compare our method with a wide range of baselines, including list-wise and point-wise ranking models. In the testing phase, we return at most 1000 tweets for each testing query, and evaluate the results with official scripts.

For shortness, our approach of "collaboratively S3VMs ranking with transitive closure" is denoted as "CSR-TC". And the following baselines are compared, including pairwise models, $i.e.$ ranking SVM and RankBoost; listwise models, $i.e.$ ListNet, AdaRank and LambdaMART; pointwise models, $i.e.$ MART and RF (Random Forests), which are implemented in RankLib[3]. And semi-supervised algorithms, namely, SR (S3VM ranking) (Zhang, He, and Luo 2012) which is sled-training, and RSCF (Tan et al. 2004) in co-training framework are also compared. All the supervised and self-training baselines are trained on combined features of $\{\chi_1, \chi_2\}$. The metrics for the ranking results are then P@10 (precision at top 10), P@20, P@30, MAP (mean average precision) and AUC (area under ROC, receiver operating characteristic curve).

3.1 Experimental Results

Start with different portions of labeled data In order to show the performance and convergence of our approach CSR-TC, we randomly sample labeled pairs from training queries in pairwise with percentage τ as set L_k, and the rest pairs and those pairs in testing query are viewed as unlabeled pairs for training, which is the way of transductive learning. In Table 1, it gives the results on P@30s, MAPs and pairCounts that are respectively the number of initially labeled pairs of RankSVM, and the summation of labeled and final selected pairs at the end of CSR-TC for all training queries. The results start with different sample percentage τ are given in groups separately. At every last row "incr" of the groups, the increase of CSR-TC are given, compared to rankSVM. It is seen that even there are only 17 pairs of training data, CSR-TC achieves an acceptable P@30 of 0.2109 on average, which improves 11.31% compared to rankSVM. With $\tau = 0.1\%$, the P@30 and MAP become comparable to the results with sampling percentage of 1%. Thus CSR-TC can achieve convinced and converged ranking results by labeling a little portion of pairs.

[3]http://people.cs.umass.edu/˜vdang/ranklib.html

Table 1: Improvement of CSR-TC with different sampling percentages.

τ= %		P@30	MAP	pairCount
0.001	RankSVM	0.1895	0.1785	17
	CSR-TC	0.2109	0.2015	42353
	incr(%)	11.31	12.85	–
0.01	RankSVM	0.1927	0.1873	201
	CSR-TC	0.2125	0.2020	63320
	incr(%)	10.28	7.88	–
0.1	RankSVM	0.1919	0.1848	2024
	CSR-TC	0.2153	0.2065	242665
	incr(%)	12.22	11.72	–
1	RankSVM	0.2006	0.1877	21814
	CSR-TC	0.2153	0.2134	526161
	incr(%)	7.35	13.67	–
10	RankSVM	0.2071	0.1885	280885
	CSR-TC	0.2220	0.2311	600613
	incr(%)	7.19	22.60	–
50	RankSVM	0.2158	0.2030	1161108
	CSR-TC	0.2243	0.2112	1555504
	incr(%)	3.94	4.04	–
100	RankSVM	0.2203	0.2053	2009384
	CSR-TC	0.2316	0.2228	2028424
	incr(%)	5.13	8.52	–

Table 2: CSR-TC with different confidence thresholds.

$\delta =$	0.1	0.3	0.5	0.7
P@30	0.2232	0.2226	0.2316	0.2203
MAP	0.2231	0.2249	0.2228	0.2053
IterCount	645	663	125	12

CSR-TC on different confidence thresholds Experiments with different confidence thresholds δ are shown in Table 2. From the results, we can see that in the setting of confidence $\delta = 0.5$, CSR-TC performs better than others. It shows that too strict selections with higher confidence 0.7, stop bringing enough unlabeled pairs to boost our semi-supervised model, which terminate within 12 iterations.

Comparison with baselines We sample 50% of the labeled data from 49 training queries in a pointwise way, and generate labeled pairs L_k. The rest of tweets from training queries are used as unlabeled data for inductive training, $i.e.$ data from testing queries are not added for training. Our CSR-TC is compared with the baselines in Table 3, on the metrics, and the improvements of our CSR-TC to the others are illustrated in columns. Since the AUC values are very closed to each other, we only list the improvement percentages of CSR-TC to other approaches in the last column. Our CSR-TC is trained with confidence threshold $\delta = 0.7$. Besides we implement SR that takes the consideration of order

Table 3: Comparison with baselines on 50% partially labeled tweets.

	P@10	our impr+ (%)	P@20	our impr (%)	P@30	our impr (%)	MAP	our impr (%)	AUC our impr (10^{-5})
RankSVM	0.2864	8.31	0.2500	2.36	0.2186	3.11	0.1910	10.16	5.48
RankBoost	0.3000	3.40	0.2424	5.57	0.2153	4.69	0.1918	9.70	18.59
ListNet	0.2932	5.80	0.2297	11.41	0.2034	10.82	0.1921	9.53	26.14
AdaRank	0.2847	8.96	0.2441	4.83	0.2102	7.23	0.1868	12.63	14.66
CA	0.2898	7.04	0.2398	6.71	0.2056	9.63	0.1914	9.93	7.84
LambdaMART	0.2305	34.58	0.2025	26.37	0.1746	29.10	0.1551	35.65	22.23
MART	0.2492	24.48	0.2076	23.27	0.1768	27.49	0.1460	44.11	16.63
RF	0.2847	8.96	0.2415	5.96	0.2096	7.54	0.1906	10.39	13.16
SR	0.2898	7.04	0.2508	2.03	0.2192	2.83	0.1917	9.75	4.09
SR-TC	0.3017	2.82	0.2517	1.67	0.2209	2.04	0.1930	9.02	11.03
RSCF	0.3034	2.24	0.2517	1.67	0.2215	1.76	0.2087	0.81	7.00
CSR	0.3051	1.67	0.2517	1.67	0.2220	1.53	0.2094	0.48	5.92
CSR-TC[4]	0.3102	–	0.2559	–	0.2254	–	0.2104	–	–

* p-values ≤ 0.012 in statistical significant testing of "our impr".
+ "our impr" is the improvement of our CSR-TC to the methods in the corresponding row.

conflict, denoted as SR-TC, and the CSR-TC without considering order conflict, denoted as CSR.

It is seen that CSR-TC improves P@30 by 3.11% and 4.69%, and MAP by 10.16% and 9.07%, compared to the pairwise models, $i.e.$, RankSVM and RankBoost separately; it improves P@30 between 7.23% and 29.10% and MAP within 9.53% and 35.65%, compared to the listwise models, $i.e.$, ListNet, AdaRank, CA, and LambdaMART; and it improves P@30 by 27.49% and 7.54%, and MAP by 44.11% and 10.39%, compared to the pointwise models, $i.e.$ MART and RF separately. We can see that CSR-TC in transductive scheme outperforms all the well-know supervised baselines. Besides, the pairwise models consistently perform better than the listwise models, which exactly shows that with only 3 levels of labeled data and extremely sparse content of tweets, listwise models cannot take its advantages to model the total list. As for those semi-supervised approaches, CSR-TC achieves 2.83% and 1.76% improvements in P@30, and 9.75% and 0.81% improvements in MAP, separately comparing to previous self-training work SR in (Zhang, He, and Luo 2012), and co-training work RSCF in (Tan et al. 2004).

Furthermore, in order to show the effects of order conflict constraint with transitive closure, we separately compared SR with SR-TC, and CSR without order conflict constraint with our CSR-TC. It is seen that both SR-TC and CSR-TC considering order conflict constraint outperforms SR and CSR consistently in precisions (P@10, P@20, P@30) and MAP. But the average AUC of SR-TC inferior to that of SR indicates the quickly descending of precision increases after the top 30 of overall 1000 returned tweets in some testing queries. Meanwhile compared with SR, SR-TC gets the increase ratio 0.78% in P@30 lower than the increase ratio 0.68% in MAP reflecting the descending as well. However, lower AUC by SR-TC does not hurt the performance to ranking tweets in practice, since people usually would not read through such a long list. The relatively small increases of SR-TC vs. SR compared to that of CSR-TC vs. CSR just give an evidence that collaboratively learning is affected by conflict pairs more easily.

The significance testings of t-test are conducted on the average improvement ratios of CSR-TC compared to all the other methods in Table 3. And p-values ≤ 0.012 indicating that CSR-TC approach outperforms others significantly.

4 Conclusions

Ranking tweets is a vital component of Microblog Search on topics, helping users get useful and timely information in an efficient way. CSR-TC is proposed to rank tweets by labeled and collaboratively selected pairs with transitive closure to avoid order conflict, due to the content sparseness and lack of labeled data in Microblog search. 51 features are extracted and split into two views, i.e. content-relevance view and tweets intrinsic view. An iterative learning algorithm is designed, and the dynamically maintained transitive closure graph on labeled and unlabeled pairs helps to honor order conflict heuristically yet efficiently. Experiments show us the convincing results.

The way we learning to rank tweets is in accordance with the schema of human learning, exploring unknown objects with what they known (semi-supervised), and the data are not i.i.d., $i.e.$, connections between an unknown objects and the one they already known. Thus our attempt to solve the specific problem in such an angle is a practice of artificial intelligence. At last, it is worth mentioning that such work contributes for the use of social media for social goodness, and can be applied to effective information management for disasters.

5 Acknowledgements

This paper is partially supported by National Grand Fundamental Research 973 Program of China (No. 2012CB316303, 2014CB340401), National Natural Science Foundation of China (No. 61202213,

61232010, 61202215, 61173064), Projects of Development Plan of the State High Technology Research (No. 2012AA011003,2014AA015204), and National Science-technology Support Plan Project (2012BAH39B04).

References

Abel, F.; Celik, I.; Houben, G.-J.; and Siehndel, P. 2011. Leveraging the semantics of tweets for adaptive faceted search on twitter. In *The Semantic Web–ISWC 2011*. Springer. 1–17.

Bennett, K.; Demiriz, A.; et al. 1999. Semi-supervised support vector machines. *Advances in Neural Information processing systems* 368–374.

Blum, A., and Mitchell, T. M. 1998. Combining Labeled and Unlabeled Data with Co-training. In *Computational Learning Theory*, 92–100.

Collins, M., and Singer, Y. 1999. Unsupervised models for named entity classification. In *Proceedings of the joint SIGDAT conference on empirical methods in natural language processing and very large corpora*, 189–196.

Duan, Y.; Jiang, L.; Qin, T.; Zhou, M.; and Shum, H.-Y. 2010. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, 295–303. Association for Computational Linguistics.

Fung, G., and Mangasarian, O. L. 2001. Semi-supervised support vector machines for unlabeled data classification. *Optimization Methods & Software* 15:29–44.

I., O.; C., M.; J., L.; and I, S. 2011. Overview of the TREC 2011 microblog track. In *TREC'11*.

I., S.; I., O.; and J, L. 2012. Overview of the TREC 2012 microblog track. In *TREC'12*.

Jabeur, L. B.; Tamine, L.; and Boughanem, M. 2012. Uprising microblogs: A bayesian network retrieval model for tweet search. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, 943–948. ACM.

Joachims, T. 1999. Transductive Inference for Text Classification using Support Vector Machines. In *International Conference on Machine Learning*, 200–209.

Li, Y.-F., and Zhou, Z.-H. 2010. S4VM: Safe Semi-Supervised Support Vector Machine. *Computing Research Repository* abs/1005.1.

Li, G.; Hoi, S. C. H.; and Chang, K. 2010. Two-view transductive support vector machines. In *SIAM International Conference on Data Mining*, 235–244.

Liu, T.-Y. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3):225–331.

Massoudi, K.; Tsagkias, M.; de Rijke, M.; and Weerkamp, W. 2011. Incorporating query expansion and quality indicators in searching microblog posts. In *Advances in Information Retrieval*. Springer. 362–367.

Nagmoti, R.; Teredesai, A.; and Cock, M. D. 2010. Ranking Approaches for Microblog Search. In *Web Intelligence*, 153–157.

Naveed, N.; Gottron, T.; Kunegis, J.; and Alhadi, A. C. 2011. Searching Microblogs: Coping with Sparsity and Document Quality. In *International Conference on Information and Knowledge Management*, 183–188.

O'Connor, B.; Krieger, M.; and Ahn, D. 2010. Tweetmotif: Exploratory search and topic summarization for twitter. *Proceedings of ICWSM* 2–3.

Ponte, J. M., and Croft, W. B. 1998. A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, 275–281.

Sakaki, T.; Okazaki, M.; and Matsuo, Y. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *World Wide Web Conference Series*, 851–860.

Tan, Q.; Chai, X.; Wilfred, N.; and Lee, D.-L. 2004. Applying co-training to clickthrough data for search engine adaptation. In *Database Systems for Advanced Applications*, 519–532. Springer.

Tong, S., and Koller, D. 2001. Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research* 2:45–66.

Usunier, N.; Amini, M.-R.; and Goutte, C. 2011. Multiview semi-supervised learning for ranking multilingual documents. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 443–458.

Xia, F.; Liu, T.-Y.; Wang, J.; Zhang, W.; and Li, H. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*, 1192–1199. ACM.

Yeh, J.-Y.; Lin, J.-Y.; Ke, H.-R.; and Yang, W.-P. 2007. Learning to rank for information retrieval using genetic programming. In *Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval (LR4IR 2007)*.

Yellin, D. M. 1993. Speeding up dynamic transitive closure for bounded degree graphs. *Acta Informatica* 30(4):369–384.

Yu, H. 2005. Svm selective sampling for ranking with application to data retrieval. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 354–363. ACM.

Zhang, X.; He, B.; and Luo, T. 2012. Transductive Learning for Real-Time Twitter Search. In *ICWSM'12*.

Zhu, B.; Gao, J.; Han, X.; Shi, C.; Liu, S.; Liu, Y.; and Cheng, X. 2012. Ictnet at microblog track trec 2012. *Proceeding of the Twenty-First Text REtrieval Conference. NIST, Gaithersburg*.