Mining Competitive Relationships by Learning across Heterogeneous Networks

Yang Yang[⋆§], Jie Tang[⋆§], Jacklyne Keomany[⋆§], Yanting Zhao[⋆§], Juanzi Li[⋆§], Ying Ding[‡], Tian Li[⋆§], and Liangwei Wang[†]

[⋆]Tsinghua National Laboratory for Information Science and Technology (TNList)
[§]Department of Computer Science and Technology, Tsinghua University
[‡]Department of Information Science, Indiana University
[†]Huawei Noah's Ark Lab

ABSTRACT

Detecting and monitoring competitors is fundamental to a company to stay ahead in the global market. Existing studies mainly focus on mining competitive relationships within a single data source, while competing information is usually distributed in multiple networks. How to discover the underlying patterns and utilize the heterogeneous knowledge to avoid biased aspects in this issue is a challenging problem. In this paper, we study the problem of mining competitive relationships by learning across heterogeneous networks. We use Twitter and patent records as our data sources and statistically study the patterns behind the competitive relationships. We find that the two networks exhibit different but complementary patterns of competitions. Our proposed model, Topical Factor Graph Model (TFGM), defines a latent topic layer to bridge the two networks and learns a semi-supervised learning model to classify the relationships between entities (e.g., companies or products). We test the proposed model on two real data sets and the experimental results validate the effectiveness of our model, with an average of +46% improvement over alternative methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Experimentation

Keywords

Web mining, Social network, Competitive relationship

1. INTRODUCTION

"Competitive strategy is an area of primary concern to managers, depending critically on a subtle understanding of industries and competitors" [21]. Indeed, competition is becoming extremely fierce in every domain with companies all over the world striving for limited resources and markets. Detecting and monitoring competitors becomes a critical issue for a company to make marketing strategies. Traditional competitor detections are usually based on observations, conjectures or sales reports. However, it is highly infeasible to manually collect the competitive relationships, considering the innumerous companies/products in the world.[1]

Recently, a few researchers have studied the problem of competitor detection. For example, Bao et al. [1] proposed an algorithm called CoMiner to identify competitors for a given entity. In this work, competitors are ranked according to the combination of several metrics including mutual information, match count, and candidate confidence. Sun et al. [22] studied the comparative web search problem, in which the user inputs a set of entities (keywords) and the system tries to find relevant and comparative information from the web for those entities. However, both works are motivated by only mining competitive relationships and not trying to reveal in which topic two entities are competing on (such as Game, Hardware, or Operation System).

In this work, we aim to conduct a systematic investigation of the problem of mining competitive relationships between entities (e.g., companies or products). Different from the related works, we try to utilize and learn from two data sources: text documents (patents) and social networks (Twitter). The reason we are using more than one data source is to avoid potential problems caused by information asymmetry. For example, some emerging companies or startups may not have any patent records. A challenge we met is how to intertwine the two sources' information properly. After all, Twitter is usually a place where the public discusses about outside apparent features yet patent records document inner core technologies that enable such features. They are entirely different in terms of contents and perspectives. Ideally, the method should combine the two pieces of information together as a heterogeneous network and thereby mine competitive relationships within it.

To clearly demonstrate the problem, Figure 1 gives an example of competitive relationships. The centered nodes are two companies: Google and Microsoft. The labels on each link indicate the fields on which the linked two companies compete with each other and the probability that connected nodes are competitors. For example, there are some well-known competitive relationships: Google competes with Facebook on social network, and competes with Microsoft on search engine. Some other competitive relation-

[1]Merely in U.S., there are more than 27 million companies, http://www.census.gov/econ/smallbus.html.

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

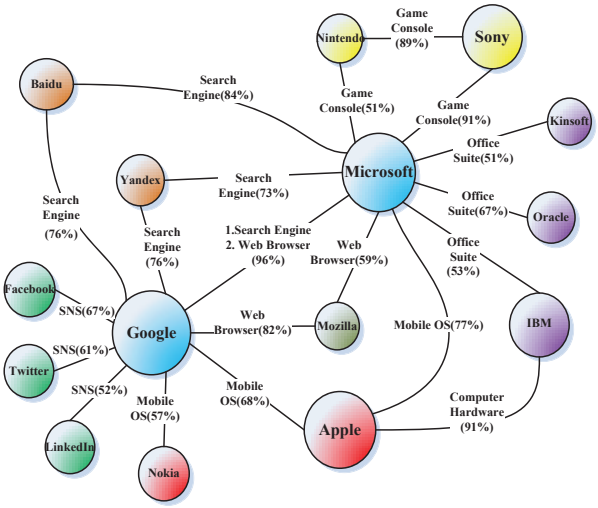


Figure 1: Examples of topic-level competitors. Each edge is associated with fields on which the connected nodes are competitive and the probability (to which extent) they are competing on.

ships (such as Microsoft competes with Kingsoft[2]) that are not so obvious and may be ignored by manual analysis can also be found in the figure. Such a graph of competitive relationships would be significantly helpful for a company to design market strategies. The problem is non-trivial and poses a set of challenges:

- *Multi-aspects*. A company is often associated with different topics and has different competitors on each of the fields corresponding to the topics. It is important to extract the topics and associate each competitive relationship with the topic information.
- *User generated content*. User generated content is an important source for mining entities' relationships. For example, [16] employed comparable questions to identify comparable entities. We also find a "10-minute phenomenon" from the Twitter data: as shown in Figure 4(a), if two companies are mentioned by a user in her tweet(s) in 10 minutes, there is a likelihood of 44% that the two companies are competitors, which is 25 times higher than chance. While on the other hand, the user generated data is very unbalanced and sparse: less than 20% of the company names examined in our experiments are mentioned on Twitter.
- *Heterogeneous sources*. Patent record is another important source for mining competitive relationships, in particular on technologies. Different from the user generated content, patents contain rich, but also much irrelevant "information" such as the disclosure statement. An interesting, but challenging, question is how to combine the user generated content and the patent information together for mining competitive relationships.

In this paper, we precisely define the problem of mining competitive relationships by learning across heterogeneous networks and propose a semi-supervised Topical Factor Graph Model (TFGM). An efficient algorithm is developed to learn the proposed model. We evaluate the proposed model on a large patent network and the Twitter network. Experimental results demonstrate that the proposed model can extensively improve the performance (averagely +46% in terms of F1-Measure) over several alternative methods. To summarize, we have the following findings through this study:

- Social network information is important for competitor mining. Actually, merely based on companies' attributes on Twitter, we can obtain a better performance (+6-57%) for mining competitive relationships than only mining on the patent data.
- Learning by utilizing heterogeneous networks can significantly improve the mining performance (+17-45%) comparing with learning over only a single network.
- It is intriguing that our experiments offer some empirical evidences for the theory of social balance [6]: "My enemy's enemy is my friend". We find a high degree (more than 90%) of balanced triads in the competitive network.

Organization Section 2 formulates the problem. Section 3 introduces the data sets and some observations we discovered. Section 4 explains our proposed model and describes the algorithm for model learning. Section 5 introduces our experiment that validates the effectiveness of our methodology, including its setup, baseline methods and results. Finally, Section 6 reviews some previous works related to ours and Section 7 concludes this work.

2. PROBLEM DEFINITION

We introduce some necessary definitions and then formulate the problem. To keep things concrete, we will use company as the example to explain the competitive relationship mining problem. The problem can be easily generalized to other entities such as products.

We consider two heterogeneous data sources: Patent and Twitter. From patent records, we extract companies, inventors, and patents. We create a network of companies $G = (V, E, \mathbf{S}), E \subseteq V \times V$, where V represents a set of companies, E represents the relationship between companies, and $\mathbf{S}$ is a matrix describing attributes associated with the companies, in which every row corresponds to a vector of attribute values of a company. For example, the attributes of a company can be inventors of those patents owned by the company and keywords occurring in the patent descriptions. Moreover, we augment the company network with social networking information. Specifically, we consider Twitter users who have discussed the companies and tweets which have mentioned the company names. Thus, the augmented network is represented as $G = (V, E, \mathbf{S}, \mathbf{U}, \mathbf{M})$, with each row of matrix $\mathbf{U}$ denoting users who have posted tweets containing the corresponding company name and each row of matrix $\mathbf{M}$ denoting tweets which contain the corresponding company name. As a conclusion, $\mathbf{S}$ is correlated to the text document (patent) data source. $\mathbf{U}$ and $\mathbf{M}$ are correlated to the social network (Twitter) data source. We further assume that each company is associated with a topic distribution. In particular, we have the following definition:

Definition 1. **Topic model of company.** A topic model θ_d of a patent d is a multinomial distribution of words $\{P(w|\theta_d)\}$. Then a company v_i is considered as a mixture of topic models, denoted as θ_{v_i}, extracted from those patents owned by the company.

The underlying assumption for the topic model is that words appearing in the patents are sampled from a distribution corresponding to each topic, i.e., $P(w|\theta_d)$. Thus, words with the highest probabilities associated with each topic would suggest the semanteme represented by the topic. For example, a "Search Engine" topic can

[2]Kingsoft has the second largest market share in Japan on office suite.

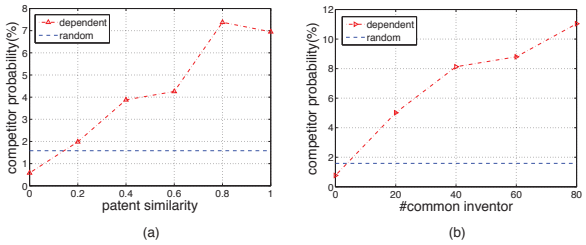


Figure 2: (a) Patent similarity correlation. X-axis: patent similarity between two companies. (b) Common employed inventors correlation. X-axis: the number of common inventors of two companies.

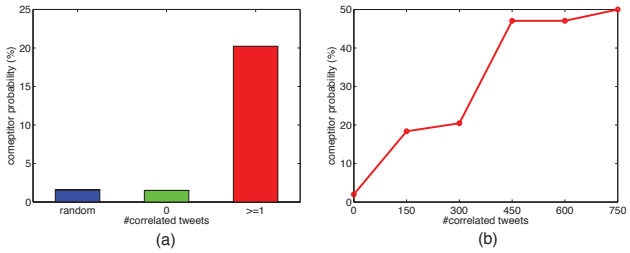


Figure 3: Tweet-level analysis. Y-axis: the probability of two companies being competitors, conditioned on the number of their co-occurring tweets.

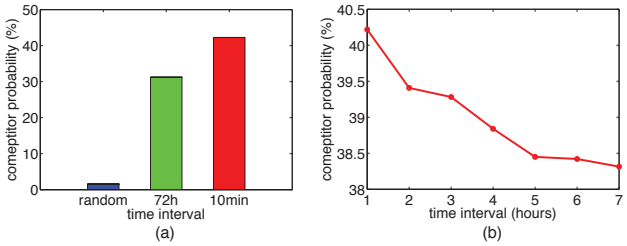


Figure 4: User-level analysis. Y-axis: the probability of two companies being competitors, conditioned on a user mentions the two companies within a particular time interval.

be represented by keywords "search", "advertisement", and "ranking".

For each edge $e \in E$, we associate it with a label $y \in \{0, 1\}$. $y = 1$ indicates corresponding two companies have a competitive relationship. Given that, we can define the problem addressed in this paper:

Problem 1. **Competitive relationship mining.** Given a network, $G = (V, E, \mathbf{S}, \mathbf{U}, \mathbf{M})$ and topic models $\{\theta\}$ of all companies, the goal is to learn a predictive function $f : (E|G) \rightarrow Y$ to infer the competitive label of each relationship between companies.

There are two things worth mentioning. The first is in the network G, we may have some labeled data, i.e., labeled competitive relationships from some online databases, but for most of the relationships the labels are unknown. The second is that the network is theoretically a complete network. We could use some parameters or human knowledge to control the density of the network. For example, only when the similarity of two companies (based on content or network information) is larger than a predefined threshold, we add an edge between them.

3. DATA AND OBSERVATION

Before presenting our approach for competitor detection, we first convey a series of discoveries we observed from the data.

3.1 Data Collection

In this study, we consider two data sources: Patent and Twitter. We have collected all the patents (3,770,411 patents) from USPTO[3], from which we extracted 195,263 companies and 2,430,375 inventors. For each company, we used it as the query to search Twitter and retrieved the top returned tweets, from which we further extracted the information of users. So far, we have collected 1,033,750 tweets written by 87,603 Twitter users, which cover 1393 major companies. In looking for benchmark data, we turn to *Yahoo! Finance*[4] and use it as the ground truth source[5]. Each company name was sent as a query to obtain its competitor list.

The probability of two randomly picked companies being competitors among the whole data set (1.59%, testified) is assigned to be the baseline probability. We compare our observation with it in order to see how different network features affect the probability.

3.2 Observations

We evaluate how patent information and Twitter reflect companies' competitive relationship from several aspects: (1) probability that two companies are competitors, conditioned on whether or not they have published similar patents or employed same inventors; (2) probability that two companies are competitors, conditioned on their names were mentioned in a same tweet or mentioned by a same user. We also study whether the phenomenon of "my enemy's enemy is my friend" exists in the competitive network.

Patent Analysis Social theory homophily suggests that similar individuals tend to associate with each other [14]. Here, we show how similarity degree of two companies correlates with the competitive relationship between them. We consider two types of similarities. The first one is based on words occurring in the descriptions of patents owned by the two companies. The second one is based on the number of common inventors, i.e., inventors that used to work for both companies at different times. For the former, we respectively generate two topic distributions θ_{v_i} and θ_{v_j} of the two companies v_i and v_j by PLSA [7] (see §4 for details). The similarity between the two companies is calculated by cosine similarity:

$$Sim(v_i, v_j) = \frac{\theta_{v_i} \cdot \theta_{v_j}}{||\theta_{v_i}|| ||\theta_{v_j}||} \quad (1)$$

Figure 2(a) clearly shows that, when the similarity of two companies increases from zero, the likelihood of them being competitors rapidly increases and becomes four times the likelihood of two random companies. We observe a similar pattern for the analysis on inventors as shown in Figure 2 (b). When no common employed inventors can be found from two companies, the probability of them being competitors drops to 1.32%, lower than the baseline probability. However, with more common inventors being detected, the probability outnumbers the baseline data and keeps increasing.

Twitter Analysis We study the likelihood of two companies being competitors when their names co-occur in tweets. Figure 3 shows the analysis results. It is striking that when the names of two companies are mentioned together in one tweet, the likelihood of the two companies being competitors becomes more than 10 times

[3]http://www.uspto.gov/
[4]http://finance.yahoo.com/
[5]For example, IBM's competitors can be found at this page: http://finance.yahoo.com/q/co?s=IBM+Competitors

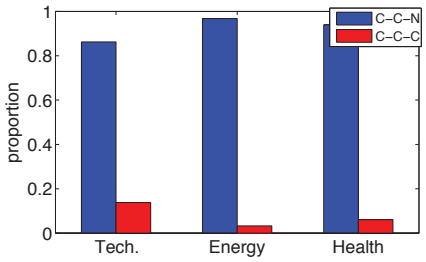


Figure 5: Comparison of two triad relationships. C-C-N: competitor-competitor-non-competitor (balanced); C-C-C: competitor-competitor-competitor (unbalanced). Y-axis: the proportion of the balanced and unbalanced triad relationships.

higher than chance. Figure 3(b) further demonstrates that the likelihood will continue to increase when the number of co-occurring tweets increase.

Besides the tweet-level co-occurrence, we conduct another analysis on the user-level. Figure 4(a) shows that when a user mentions two companies in 10 minutes (may in different tweets), the likelihood of the two companies being competitors is 25 times higher than chance. Figure 4(b) further illustrates that the likelihood drops when we set the time interval larger. We suppose that since tweets have length limitation, when a user discusses a company or its product in one tweet, she may follow up with another to mention its competitors or competitors' products.

Is my enemy's enemy my friend? We study whether competitors form a balanced network structure. The phenomenon of "the enemy of my enemy is my friend" is one of the underlying balanced triad suggested by the social balance theory [6].

In particular, we split the data into three domains: Tech. (technology), Energy and Health. In each domain, companies are grouped in triads. Suppose $e_{ij} = 1$ means company v_i and v_j are competitors, while $e_{ij} = 0$ means not. Given a triad (v_i, v_j, v_k), we compare the likelihood of $(e_{ij} = 1 \wedge e_{jk} = 1) \Rightarrow e_{ik} = 0$ (denoted as C-C-N) and that of $(e_{ij} = 1 \wedge e_{jk} = 1) \Rightarrow e_{ik} = 1$ (denoted as C-C-C). Figure 5 shows the probability of balanced triad in the three domains, from which we could have the following summary: my enemy's enemy's is not necessary my friend, but can hardly be my enemy again.

To sum up, according to the statistics shown above, we have the following discoveries:

1. As expected, similar companies tend to be competitors, with a probability of 4 times higher than chance.
2. Social network information is a very important indicator for competitors. The likelihood of two companies being competitors is 10 times higher than chance when they are mentioned in a same tweet and increases to 25 times when they are mentioned by the same user within 10 minutes.
3. My enemy's enemy is not necessary my friend, but should not be my enemy again (with a 90% likelihood).

4. OUR APPROACH

In this section, we first briefly discuss two basic models: topic models and factor graphs. We then propose a Topical Factor Graph Model (TFGM), which leverages the power of the two basic models and formulates the competitor detection problem in a unified learning framework.

4.1 Preliminary

Topic Model We first discuss the basic statistical topic models, which have been successfully applied to many text mining tasks [2, 7]. The basic idea of these models is to model documents with a finite mixture model of K topics and estimate model parameters by fitting a data set with the model. Two basic statistical topic models are Probabilistic Latent Semantic Analysis (PLSA) [7] and Latent Dirichlet Allocation (LDA) [2]. For example, the log likelihood of a collection D to be generated with PLSA is given as follows:

$$\log p(D) = \sum_d \sum_w n(w,d) \sum_{j=1}^{k} \log[p(w|z_j)p(z_j|d)] \quad (2)$$

where $n(w,d)$ denotes the occurrences of word w in a text document d, z_j is a topic and the parameters to estimate in PLSA model are $p(w|z_j)$ and $p(z_j|d)$ (or θ_d). An example of PLSA's graphical representation is shown in Figure 6(b). θ in the figure stands for the topic distribution of each text document in the data set. Given this, we can define the topic distribution of each vertex (or entity, e.g., company, product) v_i in G as a mixture of topic distribution over text documents (e.g., patents) D_{v_i} associated with v_i, i.e.,

$$\theta_{v_i} = p(z_j|v_i) = \sum_{d \in D_{v_i}} p(z_j|d)p(d|v_i) = \sum_{d \in D_{v_i}} \frac{p(z_j|d)}{|D_{v_i}|} \quad (3)$$

Factor Graph A factor graph consists of two layers of nodes, i.e., variable nodes and factor nodes, with links between them. The joint distribution over the whole set of variables can be factorized as a product of all factors. A factor graph can be learned via some efficient algorithms like the sum-product algorithm [12].

Figure 6(c) gives an example of modeling our problem with the factor graph, which incorporates entity pairs' information and labels of their relationships. For each pair of entities (v_i, v_j), we create an instance node c_k in the factor graph. For easy explanation, we use c_k^1 and c_k^2 to denote v_i and v_j respectively. The hidden variable y_k stands for the label of the relationship, with $y_k = 1$ indicating c_k^1 and c_k^2 have a competitive relationship, $y_k = 0$ not, and $y_k =?$ unknown. Our objective in the factor graph is to assign a value to the unknown y_k with high accuracy.

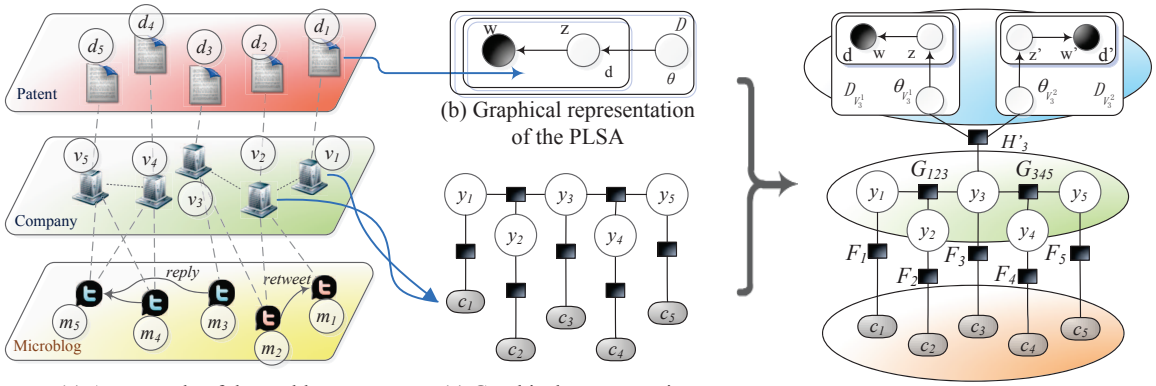
4.2 Topical Factor Graph Model

We propose a novel model referred to as Topical Factor Graph Model (TFGM) for mining competitive relationships. As we mentioned in §3, entities that have similar topic distributions are more likely to be competitors and vice versa, competitors may have similar topic distributions. Thus, the basic idea of the proposed model is to combine factor graph and topic model together, and learn them simultaneously.

Given a network $G = (V, E, \mathbf{S}, \mathbf{U}, \mathbf{M})$ with some labeled relationships Y, our objective can be formalized as to maximize the following posterior probability:

$$p(Y|G) \propto p(D|\Theta)p(Y|G, D, \Theta) \quad (4)$$

where D is a collection of all text documents. The first term on the right side of Eq. (4) can be defined according to the topic model and the second term can be defined as a factor graph. Further, to incorporate the intuition that competitors may have a similar topic distribution, we define a regularizer, which is similar to the graph



(a) An example of the problem (b) Graphical representation of the PLSA (c) Graphical representation of a factor graph (d) Graphical representation of the TFGM

Figure 6: An example of the problem and graphical representations of three different models.

harmonic function in [31], to quantify the difference between topic distributions of two entities:

$$R(Y,\Theta) = \frac{1}{2}\sum_{y_i=1}\sum_{j=1}^{K}||\theta_{c_i^1 j} - \theta_{c_i^2 j}||^2 \quad (5)$$

where K is the total number of topics.

By integrating Eqs. (4) and (5) together, we can define the following objective function to our problem:

$$\mathcal{O}(G) = (1-\lambda)\log p(D|\Theta)p(Y|G,D,\Theta) - \lambda R(Y,\Theta) \quad (6)$$

where λ is a parameter to balance the importance of the two terms.

Now we discuss how to instantiate the objective function. We can use any statistical topic model to define $p(D|\Theta)$. In this paper, we use PLSA. As to formalize $p(Y|G,D,\Theta)$, we study the corresponding entities' correlation and attributes, and we define the following three factors according to the intuitions we have discussed.

- *Attribute factor :* $F(\mathbf{x_i}, y_i)$ represents the posterior probability of y_i given the attribute vector $\mathbf{x}_i$, where $\mathbf{x}_i = (\mathbf{x}_i^1, \mathbf{x}_i^2)$, $\mathbf{x}_i^1 = (\mathbf{S}_{c_i^1}, \mathbf{U}_{c_i^1}, \mathbf{M}_{c_i^1})$, and $\mathbf{x}_i^2$ is defined similarly.

- *Balanced triangle factor :* $G(Y_c)$ reflects the correlations between each clique in Y. A set of three label nodes Y_c is a clique if the nodes stand for relationships between three entities.

- *Topic factor :* $H(y_i, \theta_{c_i^1}, \theta_{c_i^2})$ denotes the posterior probability of y_i given two corresponding entities' topic distribution.

Jointing the factors defined above, we have

$$p(Y|G,D,\Theta) = \prod_i F(\mathbf{x_i}, y_i)H(y_i, \theta_{c_i^1}, \theta_{c_i^2})\prod_c G(Y_c) \quad (7)$$

where Y_c is a triad derived from the input network. The three factors can be instantiated in different ways. In this work, we use exponential-linear functions. In particular, we define the three factors as follows:

$$F(\mathbf{x_i}, y_i) = \frac{1}{Z_1}\exp\{\sum_{j=1}^{|x_i|}\alpha_j f_j(x_{ij}, y_i)\} \quad (8)$$

$$G(Y_c) = \frac{1}{Z_2}\exp\{\beta g(Y_c)\} \quad (9)$$

$$H(y_i, \theta_{c_i^1}, \theta_{c_i^2}) = \frac{1}{Z_3}\exp\{\gamma h(y_i, \theta_{c_i^1}, \theta_{c_i^2})\} \quad (10)$$

where Z_1, Z_2 and Z_3 are normalization factors. $f_j(x_{ij}, y_i)$ and $h(y_i, \theta_{v_i^1}, \theta_{v_i^2})$ can be defined as either a binary function or real-valued function. $g(Y_c)$ can be defined as an indicator function.

Finally, by plugging Eqs. (2) and (7-10) into Eq. (6), we have

$$\begin{aligned}\mathcal{O}(\Psi) = (1-\lambda)[&\sum_d\sum_w n(w,d)\log\sum_{j=1}^{k}p(w|z_j)p(z_j|d) \\ &+ \sum_{i=1}^{|Y|}\sum_{j=1}^{d}\alpha_j f_j(x_{ij}, y_i) + \sum_c \beta g(Y_c) \\ &+ \sum_{i=1}^{|Y|}\gamma h(y_i, \theta_{c_i^1}, \theta_{c_i^2}) - \log Z] - \lambda R(Y,\Theta)\end{aligned} \quad (11)$$

where Ψ is the collection of parameters, i.e., $\Psi = \{p(w|z_j)\} \cup \{p(z_j|d)\} \cup \{\alpha_i\} \cup \{\beta\} \cup \{\gamma\}$, and $Z = Z_1 Z_2 Z_3$ is a normalization factor. Our goal is to estimate a parameter configuration Ψ to maximize the objective function $\mathcal{O}(\Psi)$.

The graphical representation of TFGM is shown in Figure 6(d). The upper layer is used for modeling the topic extraction task and the bottom layer is designed to model the competitor detection task. Actually we can combine $R(Y,\Theta)$ and $H(Y,\Theta)$ together as one factor function H' to bridge the two tasks. We separate $R(Y,\Theta)$ and $H(Y,\Theta)$ to easily explain how we learn the model in the rest of this section.

4.3 Model Learning

To estimate the parameters in TFGM, let us first consider the special case when $\lambda = 0$. The objective function degenerates to $\log p(Y|G)$ with no regular function in this case. To maximize $\log p(Y|G)$, we first apply an Expectation Maximization (EM) algorithm, a standard way of parameter estimation of PLSA, to iteratively compute a local maximum of $\log p(D|\Theta)$. After that, we compute the values of Θ based on Eq. (3) and maximize

$\log p(Y|G, D, \Theta)$ by a gradient descent method. We repeat the two steps until the objective function converges.

The details of how to estimate the parameters of PLSA can be seen in [7]. When computing $p(Y|G, D, \Theta)$, we need to sum up the likelihood of possible states for all the nodes, including the unlabeled ones, to normalize Z. To deal with this, we infer the unlabeled labels from known ones. Y^U is denoted as a labeling configuration inferred from known labels. We then have:

$$\begin{aligned}\log p(Y|G, D, \Theta) &= \log \Sigma_{Y^U} p(Y^U|G, D, \Theta) \\ &= \log \Sigma_{Y^U} \exp\{\mu^T \mathbf{Q}(Y^U)\} \\ &\quad - \log \Sigma_Y \exp\{\mu^T \mathbf{Q}(Y)\} \qquad (12)\end{aligned}$$

where $\mathbf{Q}(Y) = ((\sum_i f_j(x_{ij}, y_i))^T, \sum_c g(Y_c), \sum_i h(y_i, \theta_{c_i^1}, \theta_{c_i^2}))^T$, and $\mu = (\alpha^T, \beta, \gamma)^T$.

We introduce the gradient descent method to solve the function. The gradient for each parameter μ is calculated as:

$$\begin{aligned}\nabla &= \frac{\partial \log p(Y|G, D, \Theta)}{\partial \mu} \\ &= \mathbb{E}_{p_\mu(Y^U|G,D,\Theta)} \mathbf{Q}(Y^U) - \mathbb{E}_{p_\mu(Y|G,D,\Theta)} \mathbf{Q}(Y) \qquad (13)\end{aligned}$$

One challenge here is to directly calculate the two expectations. The graphical structure of TFGM may be arbitrary and contain cycles. Thus, we adopt Loopy Belief Propagation (LBP) [20] approximate algorithm to compute the marginal probabilities of Y and Y^U. We are then able to obtain the gradient by summing over all the label nodes. An important point here is that the LBP process needs to be proceeded twice during the learning procedure, one for estimating $p(Y|G, D, \Theta)$ and the other for $p(Y^U|G, D, \Theta)$. We update each parameter with a learning rate ξ with the gradient.

We now discuss the case when $\lambda \neq 0$. In this general case the objective function does not have a closed-form solution. Here, we propose a simple and efficient algorithm which primarily consists of two steps. In the first step, we update $p(z_j|d)$, $p(w|z_j)$ and μ according to the same method in case $\lambda = 0$. In the second step, we fix $p(w|z_j)$ and μ to update $p(z_j|d)$ as follows:

$$\begin{aligned}p_{n+1}(z_j|d_{v_i}) &= (1-\eta) p_n(z_j|d_{v_i}) \\ &\quad + \eta \frac{\sum_{y_{(v_i,v_k)}=1} \sum_{d_{v_k} \in D_{v_k}} p_n(z_j|d_{v_k})}{\sum_{y_{(v_i,v_k)}=1} |D_{v_k}|} \qquad (14)\end{aligned}$$

where D_{v_k} denotes the text documents associated with v_k, $d_{v_i} \in D_{v_i}$, and $y_{(v_i,v_k)}$ stands for the label correlated with entities v_i and v_k. Clearly, $\sum_j p_n(z_j|d_{v_i}) = 1$ and $p_n(z_j|d_{v_i}) > 0$ always hold in Eq. (14). When the step parameter η is set to 1, it means the new topic distribution of a text document, which belongs to entity v_i, is the average of the old distributions from all documents of v_i's competitors. This is related to the random-walk interpretation. A similar algorithm was also used in [19]. See details in Algorithm 1.

In factor graph, we can also consider making use of topic model's results to help mining competitive relationships; however, the topics are treated equally including ones that might be irrelevant to competitions. In contrast, Topical Factor Graph Model, with the regularizer, can distinguish "competition topics" from irrelevant topics thus to mine competitive relationships more effectively.

5. EXPERIMENTAL RESULTS

In this section, we validate the effectiveness of the proposed approach.

Input: a network G, a partially labeled competitor matrix Y, the learning rate η and ξ, maximum iteration number I and J.
Output: estimated parameter Ψ

Initialize $p(z_j|d), p(w|z_j)$ randomly;
Initialize $\mu \leftarrow 0$;
repeat
  Update $p(z_j|w, d)$, $p(w|z_j)$, and $p(z_j|d)$ to maximize $\log p(D|\Theta)$ with EM algorithm.
  Calculate Θ with Eq. (3);
  for $i = 1$ *to* I **do**
    Call LBP to calculate $\mathbb{E}_{p_\mu(Y^U|G,D,\Theta)} \mathbf{Q}(Y^U)$;
    Call LBP to calculate $\mathbb{E}_{p_\mu(Y|G,D,\Theta)} \mathbf{Q}(Y)$;
    Calculate ∇_μ with Eq. (13);
    Update $\mu_{new} = \mu_{old} - \xi \cdot \nabla_\mu$
  end
  for $n = 1$ *to* J **do**
    Update $p_{n+1}(z_j|d_{v_i})$ with Eq. (14);
  end
until *Convergence*;

Algorithm 1: Learning algorithm of TFGM

5.1 Experimental Setup

Data Preparation We consider two data sets in our evaluation: Company and Product.

Company. Description of the company data set is given in §3. As there is no standard ground truth to quantitatively evaluate the performance of mining competitive relationships, for evaluation purpose, we have collected the competitive relationships between companies from *Yahoo! Finance*. Specifically, Yahoo! Finance provides a list of competitors for each company.[6] It also categorizes all the companies into different domains (called sector) such as technology, energy, and health. Each company may be classified into two domains. In this way, we create a ground truth for evaluating topic-level competitive relationships mining. In total, the company data set contains 1,393 companies from three domains.

Product. The product data was extracted from Epinions, a website on which users pose reviews on their purchased products. We extracted information between two products such as price difference, reviewers who had reviewed on both of the products, comments that had both of the products' names as social networks features. The text information which supports the topic model was derived from the products' reviews. The data set consists of 120 products, 972 reviews of the products, and 861 users who wrote comments on these products. Some example products include Canon 550D, Canon 5D Mark II (5d mii), Nikon D90, iPhone 4, iPad 2 and Amazon Kindle 2.

Evaluation We conduct two types of experiments to evaluate the proposed approach. The first one is to identify global competitors. We evaluate the proposed model and compare it with alternative methods in terms of Precision (Prec.), Recall (Rec.), F1-Measure (F1), and Accuracy (Accu.). The second experiment is to detect competitors at specific topics, we define the probability of two competitors v_1 and v_2 competing in an area described by specific topic z as

$$p(v_1, v_2|z) = \frac{p(z|v_1)p(z|v_2)}{p(z)} \qquad (15)$$

In each experiment, we randomly picked 40% in each category as training (labeled) data and the rest as test (unlabeled) data. For evaluating the performance of topic-level competitor detection: we first determine whether two companies have a competitive relation-

[6]For example, http://finance.yahoo.com/q/co?s=MSFT+Competitors

Table 1: Competitor detection performance of different methods in three domains.

Domain	Method	Prec.	Rec.	F1	Accu.
Tech.	CS	0.1858	0.7574	0.2983	0.3706
	TF	0.2312	0.2585	0.2441	0.5544
	RW	0.4605	0.2482	0.3226	0.8489
	SVM	0.6643	0.5793	0.6189	0.8027
	LR	0.5636	0.5671	0.5653	0.7589
	FGM	0.7400	0.6768	0.7070	0.8449
	TFGM	**0.7576**	**0.7622**	**0.7599**	**0.8668**
Energy	CS	0.2072	0.4200	0.2775	0.1335
	TF	0.3158	0.0882	0.1379	0.5930
	RW	0.3488	0.4115	0.3776	0.5774
	SVM	0.4444	0.1429	0.2162	0.7844
	LR	0.3750	0.2143	0.2727	0.7621
	FGM	**0.6644**	0.9340	0.7765	**0.8571**
	TFGM	0.6558	**0.9528**	**0.7769**	0.8546
Health	CS	0.1175	0.0822	0.0967	0.0233
	TF	0.2727	0.0045	0.0089	0.5653
	RW	0.1581	0.1235	0.1387	0.6306
	SVM	0.7000	0.1000	0.1750	0.7471
	LR	0.1667	0.0142	0.0263	0.7165
	FGM	0.9041	0.9429	0.9231	0.9579
	TFGM	**0.9178**	**0.9571**	**0.9371**	**0.9655**

ship or not. After that, given a topic z and a company v_1, we rank its competitors by $p(v_1, v_2|z)$. At last we compare the rank with the ground truth from Yahoo! finance in terms of precision at position n (P@n), mean average precision (MAP) and normalized discount cumulative gain at position n (N@n). A similar method was previously used in [25].

We compare TFGM with the following baseline methods.

Content Similarity (CS). It calculates the cosine similarity between two companies' topic distributions and labels companies as competitors if their similarity value is greater than a threshold (0.2). We design it to see how unsupervised method works in this task.

Twitter Filtering (TF). It simply labels companies who have been mentioned in a same tweet at least one time as competitors. It is also an unsupervised method.

Random Walk with Restart (RW). It uses the network information to identify competitive relationships. Specifically, it builds up a tripartite graph which contains three types of node: inventors, companies, and patent categories (topics). For each company node v and topic node z, it creates a link from v to z and a link with opposite direction. Then the random walk with restart algorithm [28, 27] is applied to rank competitors.

SVM. It uses all the features we defined in TFGM (see Appendix for details) to train a classification model (but SVM does not consider the correlation among the identified competitive relationships). We then employ it to predict the company pairs' labels in the test data. For SVM, we choose LIBSVM [3].

LR. It uses the same features as in the SVM method. The only difference is the way in which it uses logistic regression classification to predict the labels in the test data. The method was used in [15] to predict positive and negative links in social networks.

FGM. It trains a factor graph model with partially labeled data and all factors we defined in §4. This method can also been regarded as a special case of TFGM when $\lambda = 0$. This method was used in [26] to classify the type of social relationships.

Table 2: Topic-level competitor detection performance of different methods in three domains.

Domain	Method	P@5	P@10	MAP	N@5	N@10
Tech.	RW	0.3556	0.2616	0.3614	0.3917	0.3137
	TFGM	0.6762	0.4270	0.7657	0.6342	0.5542
Energy	RW	0.2455	0.1712	0.0518	0.2391	0.1898
	TFGM	0.6182	0.3614	0.8785	0.7079	0.6392
Health	RW	0.1067	0.1046	0.0094	0.1143	0.1104
	TFGM	0.3677	0.2225	0.8861	0.8233	0.7328

All algorithms are implemented in C++, and all experiments are performed on a Mac running Mac OS X with Intel Core i7 2.66 GHz and 4 GB memory. We empirically set the number of topics in TFGM as 100, and set parameters $\eta = 0.1$ and $\lambda = 0.5$ in all other experiments. We will give the sensitivity analysis of these parameters later. We also set the maximum iteration number $I = 500$ and $J = 20$. In general, the efficiency of TFGM is acceptable. It takes 2 hours to learn from the company data set.

5.2 Quantitative Results

Table 1 shows the results of detect competitors globally with different approaches on the company data set. We can see that TFGM clearly outperforms CS, TF, RW, SVM and LR in all domains (+57.98% in terms of the average F1). CS, TF and RW methods only consider content information, which leads to a bad performance. Compared with SVM and LR, one of TFGM's advantages is making use of the unlabeled data. Essentially, it further considers some latent correlations in the data set, which cannot be leveraged with only the labeled training data. At the same time, TFGM also shows satisfying robustness. We can see that SVM and LR have unstable performances over different domains. For example, in Tech. domain, SVM has F1 of 0.62 which falls to 0.18 in Health domain. This is because competitive relationships in Health domain are quite sparse, which makes SVM mostly label company relationships as not competitive. Compared to FGM, with topic model incorporated, TFGM differentiates "competition topics" from those irrelevant topics and obtains a further improvement (e.g., +5% F1-score in Tech. domain).

There are two ways to detect topic-level competitors. One is the method we introduced above in §5.1. However, there is a different method for Random walk with restart: if we remove all "topic nodes" except one of them, the result would be the competitors in the corresponding topic. There are many ways in implementing the first method, e.g., all baselines. However due to the space limitation, we only present results generated by TFGM. Also, baseline methods produced poor results in the first few steps, thus it is reasonable to ignore them. Table 2 shows comparison result of TFGM and RW, from which we can see that TFGM clearly outperforms RW.

5.3 Analysis and Discussion

Factor Contribution To determine the contributions of different factors to the model performance, we remove them one by one (first balanced triangle factor function, followed by the topic factor function), and then train and evaluate the performance. Figure 7 shows

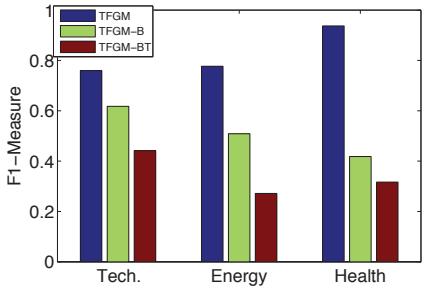


Figure 7: Factors contribution. TFGM-B: ignoring balanced triangle factor function. TFGM-BT: further ignoring topic factor function.

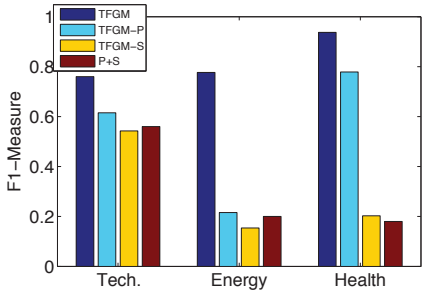


Figure 8: Network contribution. TFGM-P: ignoring the patent information. TFGM-S: ignoring the social network. P+S: combining the two networks by simply labeling competitor relationships basing on whether TFGM-P OR TFGMS has labeled it.

the F1-Measure score after ignoring the factor functions. We can observe clear drops on the performance, which indicates that each factor incorporated in the model has its specific contribution to the final result.

How Heterogeneous Networks Help Social network and patent network are two fundamental constituent parts of the heterogeneous network we are studying on. To study how heterogeneous network helps solve this problem, we dismiss the two data source respectively. Furthermore, we design another method to make use of the heterogeneous data source: we regard two companies as competitors if either of the methods based on a single data source labels them as competitors. Figure 8 shows the F1-Measure of these three methods comparing to the original approach. We can see that the model with both components incorporated exceeds the other two incomplete TFGM greatly in performance, which indicates that our model works better by learning across a heterogeneous network than either of the two networks. P+S's score drops greatly compared with TFGM's. It even underperforms methods based on a single data source. By investigation, we find that if either one of TFGM-P and TFGM-S mistakenly labeled two entities as competitors, P+S keeps the mistake, which has severe adverse impact on the precision of the model.

Sensitivity Analysis We conduct two experiments to test how parameter η and λ influence TFGM's performance. Figure 9 shows the trend of each measure following the changes of η in all domains (λ is fixed as 0.5). TFGM has low sensitivity of η in Energy and Health domains (the largest difference of F1 is less than 4% in both domains). However, in Tech. domain, the precision value slowly rises as η grows and then falls after $\eta = 0.6$. The recall value overall stays stable, yet it has a rapid fall from $\eta = 0.1$ to $\eta = 0.2$. We then fix $\eta = 0.1$ and see how F1-score changes by varying λ. As Figure 10 shows, the score increases slowly at the beginning but falls a bit more quickly when λ becomes larger (> 0.5).

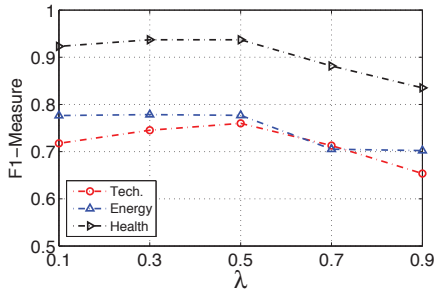


Figure 10: Performance of TFGM by varying parameter λ.

Table 3: Examples of topic-level competitors.

Topic	Words	Competitors
Topic #4	image graphics pixel 3d	NVIDIA Vs. Autodesk
		Adobe Vs. VMware
		VMware Vs. Autodesk
		Microsoft Vs. NVIDIA
Topic #31	database distributed query domain	Oracle Vs. Jabil Circuit
		Yahoo! Vs. Jabil Circuit
		Google Vs. Jabil Circuit
		Microsoft Vs. Google
Topic #76	semiconductor toner compositions chamber	Novellus Systems Vs. Intel
		First Solar Vs. CREE
		Applied Materials Vs. IBM
		Motorola Vs. CREE

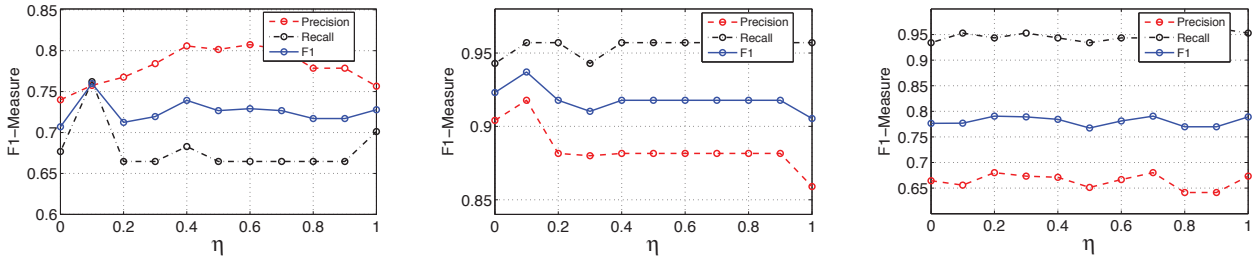
5.4 Qualitative Results

In this section, we demonstrate some examples generated from our experiments to show the effectiveness of our approach.

Topic-level Competitor Analysis We study on topic-level competitor cases to see in real how text topics information helps competitor analysis. Table 3 displays results of several examples, listing the top competitors under the area given by a topic. As in Topic #4 describing graphic design, while the top competitors given by our model, NVIDIA and Autodesk, are two of the industry leaders.

On the other hand, given a pair of competitors, we try to figure out under which areas they are competing. Table 4 shows the top two topics for each pair of competitors according to $p(v_1, v_2|z)$. We can tell that our model finds Samsung and Apple actually correlating to topics like "communicating" etc, indicating mobile phones, and "program","processor", indicating computers – corresponds to the real situation. Similar results can be seen in topics correlated to Microsoft and Google.

Competitive Relationships between Products Our model is flexible and can be easily applied to other data sets. We apply it to find competitive relationships between products. Table 5 shows an example result compared with FGM. As we can see, both TFGM and FGM detect Nikon D90 as a competitor of both of the Canon cameras. But FGM wrongly labels Kindle 2 and 550D as competitors. Under our study, we find that many users discussed about how Kindle 2 or 550D is better than older versions, which makes the two products' distributions of the topic "version" similar to each



(a) Tech. (b) Energy (c) Health

Figure 9: Performance of TFGM by varying step parameter η.

Table 4: Examples of competitors correlating topics.

Competitors	Topic	Hot words
Samsung Vs. Apple	Topic #16	communicating, microprocessors aluminum, cover, operating
	Topic #93	electronic, processor, program memory, monitor, multiple
Microsoft Vs. Google	Topic #48	query, web, knowledge database, based, search
	Topic #81	operating, program, service system, software, manage

Table 5: Examples of competitors among products. ✓: the results of TFGM, △: the results of FGM.

	550d	5d mii	d90	Iphone4	Ipad2	kindle2
550d		✓△	✓△	✓		△
5d mii	✓△		✓△			
d90	✓△	✓△			△	
iphone4	✓					✓
ipad2			△			✓
kindle	△			✓	✓	

other. It, therefore, contributes a positive weight to labeling them as competitors. Yet they are obviously not competitors and "version" is not a classic topic about competition. FGM is misled by this phenomenon while TFGM distinguishes this irrelevant topic from valuable ones.

Another interesting fact is that TFGM considers iPhone 4 and 550D as competitors. This is feasible since iPhone 4, with excellent photo-shooting performance and a similar price, is quite an alternative of 550D from customers' perspectives. At the same time, although iPad 2 has a camera built in, it is not often used for taking pictures. Thus, TFGM does not treat it as competitors of the cameras.

6. RELATED WORK

In this section, we review the related works from three aspects: competitor detection, studies on Twitter, and patent mining.

Competitor Detection Similar studies have been conducted with regards to competitor detection on the web. Using semantic analysis and text mining technique, Chen et al. [4] propose a framework to extract information from a user's website and learn his/her background knowledge. An algorithm that also infers competitive analysis is CoMiner, that Bao el al. [1] propose. CoMiner conducts a Web-scale mining for a company's competitive candidates, domain and competitive strength. Their methods, however, are significantly different from ours. We not only consider the text information, but also incorporate the social network information. Another related work is Liu et.al's methods of discovering unexpected information from competitors' web sites [17]. This work focuses on analyzing competitors' features rather than detecting them, which is obviously different from what we are trying to do. Other related works including Li et al. [16] and Yang et al.'s [30] extract comparable entities by detecting keywords describing comparisons from online text documents. The two works study on a single data source while our method utilizes heterogenous networks.

Twitter Study Existing Twitter studies mainly include: Mathioudakis and Koudas [18] present a system, TwitterMonitor, to extract emerging topics from tweets' content; [13, 29, 9, 23] mainly focus on identifying influential users in Twitter or examining and predicting tweeting behaviors of users; Kwak et al. [13] conduct a study on Twitter network and perceive some notable properties of Twitter; Hopcroft et al. [8] explore the problem of reciprocal relationship prediction on Twitter; Tang et al. [24] have developed a framework for classifying the type of social relationships by learning across heterogeneous networks. As far as we know, few works in the literature have tried to use Twitter or other microblog data for competitor detection.

Patent Mining In this paper, we also employ a set of patents information to assist for this competitor detections problem.There are also many related works on patent mining. Kasravi et al. [11] propose a method to discover business value from patent repositories, Jin et al. [10] introduce a new problem of patent maintenance prediction and propose a method to solve it, while Ernst [5] uses patent information for strategic technology management including competitor monitoring. But these works only consider patent information, while we combine social networks and patents together to solve the competitor detection problem more effectively.

7. CONCLUSION AND FUTURE WORK

In this paper, we study the problem of mining competitive relationships by learning across heterogeneous networks. Some features of competitive relationships, which reflect social network and patent information, are discovered and analyzed. We then formally define the problem in a semi-supervised framework and propose a Topical Factor Graph Model (TFGM) for detecting competitors with social network and text document attributes given. In TFGM, factor graph and topic model are incorporated. Efficient algorithms are proposed for learning parameters as to infer unknown relationships. Experiments on two different data sets have been conducted and results outperform several alternatives greatly.

Another interesting topic to think about is how to detect potential collaborators. We believe that methods of collaborator analysis will resemble the ones that we propose in this paper. In future work, we will try to apply the existing methods on competitive detection to

collaborative detection and figure out whether additional theories or algorithms will need to be involved.

Acknowledgements. The work is supported by the Natural Science Foundation of China (No. 61073073), National Basic Research Program of China (No. 2011CB302302), and Chinese National Key Foundation Research (No. 60933013, No.61035004).

8. REFERENCES

[1] S. Bao, R. Li, Y. Yu, and Y. Cao. Competitor mining with the web. *IEEE Trans. Knowl. Data Eng.*, pages 1297–1310, 2008.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[3] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM TIST*, 2:27:1–27:27, 2011.

[4] X. Chen and Y.-F. B. Wu. Web mining from competitors' websites. In *KDD'05*, pages 550–555, 2005.

[5] H. Ernst. Patent information for strategic technology management. *World Patent Information*, 25(3):233–242, September 2003.

[6] F. Heider. Attitudes and cognitive organization. *Journal of Psychology*, 21(2):107–112, 1946.

[7] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR'99*, pages 50–57, 1999.

[8] J. E. Hopcroft, T. Lou, and J. Tang. Who will follow you back? reciprocal relationship prediction. In *CIKM'11*, 2011.

[9] B. A. Huberman, D. M. Romero, and F. Wu. Social networks that matter: Twitter under the microscope. *First Monday*, 14(1), 2009.

[10] X. Jin, S. Spangler, Y. Chen, K. Cai, R. Ma, L. Zhang, X. Wu, and J. Han. Patent maintenance recommendation with patent information network model. In *ICDM'11*, 2011.

[11] K. Kasravi and M. Risov. Patent mining - discovery of business value from patent repositor ies. *Hawaii International Conference on System Sciences*, 0:54b, 2007.

[12] F. R. Kschischang, B. J. Frey, and H. andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE TOIT*, 47:498–519, 2001.

[13] H. Kwak, C. Lee, H. Park, and S. B. Moon. What is twitter, a social network or a news media? In *WWW*, pages 591–600, 2010.

[14] P. F. Lazarsfeld and R. K. Merton. *Friendship as a social process: A substantive and methodological analysis*, volume 18, pages 18–66. Van Nostrand, 1954.

[15] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Predicting positive and negative links in online social networks. In *WWW'10*, pages 641–650, 2010.

[16] S. Li, C.-Y. Lin, Y.-I. Song, and Z. Li. Comparable entity mining from comparative questions. In *48th AMACL*, ACL '10, pages 650–658, 2010.

[17] B. Liu, Y. Ma, and P. S. Yu. Discovering unexpected information from your competitors' web sites. In *KDD*, pages 144–153, 2001.

[18] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD'10*, pages 1155–1158, 2010.

[19] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *WWW'08*, pages 101–110, 2008.

[20] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI'99*, pages 467–475, 1999.

[21] M. E. Porter. *Competitive Strategy: Techniques for Analyzing Industries and Competitors*. Free Press, 1 edition, June 1998.

[22] J.-T. Sun, X. Wang, D. Shen, H.-J. Zeng, and Z. Chen. Cws: a comparative web search system. In *WWW'06*, pages 467–476, 2006.

[23] C. Tan, J. Tang, J. Sun, Q. Lin, and F. Wang. Social action tracking via noise tolerant time-varying factor graphs. In *KDD'10*, pages 1049–1058, 2010.

[24] J. Tang, T. Lou, and J. Kleinberg. Inferring social ties across heterogenous networks. In *WSDM'12*, pages 743–752, 2012.

[25] J. Tang, B. Wang, Y. Yang, P. Hu, Y. Zhao, X. Yan, B. Gao, M. Huang, P. Xu, W. Li, and A. K. Usadi. Patentminer: Topic-driven patent analysis and mining. In *KDD'2012*, 2012.

[26] W. Tang, H. Zhuang, and J. Tang. Learning to infer social ties in large networks. In *ECML/PKDD (3)'11*, pages 381–397, 2011.

[27] H. Tong, C. Faloutsos, and Y. Koren. Fast direction-aware proximity for graph mining. In *KDD'07*, pages 747–756, 2007.

[28] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM'06*, pages 613–622, 2006.

[29] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In B. D. Davison, T. Suel, N. Craswell, and B. Liu, editors, *WSDM*, pages 261–270, 2010.

[30] S. Yang and Y. Ko. Extracting comparative entities and predicates from texts using comparative type classification. In *49th AMACL*, HLT '11, pages 1636–1644, 2011.

[31] X. Zhu and J. Lafferty. Harmonic mixtures: Combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *ICML'05*, pages 1052–1059, 2005.

Appendix: Factor Function Definition

We introduce how we define the factor functions in our model. For *attribute factor function*, we define three categories of features.

Social correlation In the company data set, we consider tweets related to both companies in two features: the number of tweets with their co-occurrence and the number of tweet-pairs published by one user in a small time interval, in which one tweet is related to one company respectively. In the product data set, we also consider the two similar features corresponding to reviews on products.

Social homophily Whether two companies or products have equal social status. In the company data set, we define three features: the number of tweets related to each company, the number of company's official account's Twitter followers, and the number of users who follow both companies. We define two features in the product data set: the price difference of the two products and the number of reviews on each product.

Local homophily In the company data set, we extract patent and inventor information of each company and consider whether two companies have common points in this. We use two features: the number of common inventors and the number of patents they have. In the product data set, we consider only one feature: the number of users who reviewed both products.

For *balanced triangle factor function*, we define eight features to capture all the possible situations for every three links. We define *topic factor function* as the cosine similarity between the two topic distributions.