Deceptive Answer Prediction with User Preference Graph

Fangtao Li[§], **Yang Gao**[†], **Shuchang Zhou**[§‡], **Xiance Si**[§], **and Decheng Dai**[§]
[§]Google Research, Mountain View
[‡]State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS
{lifangtao,georgezhou,sxc,decheng}@google.com
[†]Department of Computer Science and Technology, Tsinghua University
gao_young@163.com

Abstract

In Community question answering (QA) sites, malicious users may provide deceptive answers to promote their products or services. It is important to identify and filter out these deceptive answers. In this paper, we first solve this problem with the traditional supervised learning methods. Two kinds of features, including textual and contextual features, are investigated for this task. We further propose to exploit the user relationships to identify the deceptive answers, based on the hypothesis that similar users will have similar behaviors to post deceptive or authentic answers. To measure the user similarity, we propose a new user preference graph based on the answer preference expressed by users, such as "helpful" voting and "best answer" selection. The user preference graph is incorporated into traditional supervised learning framework with the graph regularization technique. The experiment results demonstrate that the user preference graph can indeed help improve the performance of deceptive answer prediction.

1 Introduction

Currently, Community QA sites, such as Yahoo! Answers[1] and WikiAnswers[2], have become one of the most important information acquisition methods. In addition to the general-purpose web search engines, the Community QA sites have emerged as popular, and often effective, means of information seeking on the web. By posting questions for other participants to answer, users can obtain answers to their specific questions. The Community QA sites are growing rapidly in popularity. Currently there are hundreds of millions of answers and millions of questions accumulated on the Community QA sites. These resources of past questions and answers are proving to be a valuable knowledge base. From the Community QA sites, users can directly get the answers to meet some specific information need, rather than browse the list of returned documents to find the answers. Hence, in recent years, knowledge mining in Community QA sites has become a popular topic in the field of artificial intelligence (Adamic et al., 2008; Wei et al., 2011).

However, some answers may be deceptive. In the Community QA sites, there are millions of users each day. As the answers can guide the user's behavior, some malicious users are motivated to give deceptive answers to promote their products or services. For example, if someone asks for recommendations about restaurants in the Community QA site, the malicious user may post a deceptive answer to promote the target restaurant. Indeed, because of lucrative financial rewards, in several Community QA sites, some business owners provide incentives for users to post deceptive answers for product promotion.

There are at least two major problems that the deceptive answers cause. On the user side, the deceptive answers are misleading to users. If the users rely on the deceptive answers, they will make the wrong decisions. Or even worse, the promoted link may lead to illegitimate products. On the Community QA side, the deceptive answers will hurt the health of the Community QA sites. A Community QA site without control of deceptive answers could only benefit spammers but could not help askers at all. If the asker was cheated by the provided answers, he will not trust and visit this site again. Therefore, it is a fundamental task to predict and filter out the deceptive answers.

In this paper, we propose to predict deceptive

[1]http://answers.yahoo.com
[2]http://wiki.answers.com

answer, which is defined as the answer, whose purpose is not only to answer the question, but also to promote the authors' self-interest. In the first step, we consider the deceptive answer prediction as a general binary-classification task. We extract two types of features: one is textual features from answer content, including unigram/bigram, URL, phone number, email, and answer length; the other is contextual features from the answer context, including the relevance between answer and the corresponding question, the author of the answer, answer evaluation from other users and duplication with other answers. We further investigate the user relationship for deceptive answer prediction. We assume that similar users tend to have similar behaviors, i.e. posting deceptive answers or posting authentic answers. To measure the user relationship, we propose a new user preference graph, which is constructed based on the answer evaluation expressed by users, such as "helpful" voting and "best answer" selection. The user preference graph is incorporated into traditional supervised learning framework with graph regularization, which can make answers, from users with same preference, tend to have the same category (deceptive or authentic). The experiment results demonstrate that the user preference graph can further help improve the performance for deceptive answer prediction.

2 Related Work

In the past few years, it has become a popular task to mine knowledge from the Community QA sites. Various studies, including retrieving the accumulated question-answer pairs to find the related answer for a new question, finding the expert in a specific domain, summarizing single or multiple answers to provide a concise result, are conducted in the Community QA sites (Jeon et al., 2005; Adamic et al., 2008; Liu et al., 2008; Song et al., 2008; Si et al., 2010a; Figueroa and Atkinson, 2011). However, an important issue which has been neglected so far is the detection of deceptive answers. If the acquired question-answer corpus contains many deceptive answers, it would be meaningless to perform further knowledge mining tasks. Therefore, as the first step, we need to predict and filter out the deceptive answers. Among previous work, answer quality prediction (Song et al., 2010; Harper et al., 2008; Shah and Pomerantz, 2010; Ishikawa et al., 2010) is most related to the deceptive answer prediction task. But these are still significant differences between two tasks. Answer quality prediction measures the overall quality of the answers, which refers to the accuracy, readability, completeness of the answer. While the deceptive answer prediction aims to predict if the main purpose of the provided answer is only to answer the specific question, or includes the user's self-interest to promote something. Some of the previous work (Song et al., 2010; Ishikawa et al., 2010; Bian et al., 2009) views the "best answer" as high quality answers, which are selected by the askers in the Community QA sites. However, the deceptive answer may be selected as high-quality answer by the spammer, or because the general users are mislead. Meanwhile, some answers from non-native speakers may have linguistic errors, which are low-quality answers, but are still authentic answers. Our experiments also show that answer quality prediction is much different from deceptive answer prediction.

Previous QA studies also analyze the user graph to investigate the user relationship (Jurczyk and Agichtein, 2007; Liu et al., 2011). They mainly construct the user graph with asker-answerer relationship to estimate the expertise score in Community QA sites. They assume the answerer is more knowledgeable than the asker. However, we don't care which user is more knowledgeable, but are more likely to know if two users are both spammers or authentic users. In this paper, we propose a novel user preference graph based on their preference towards the target answers. We assume that the spammers may collaboratively promote the target deceptive answers, while the authentic users may generally promote the authentic answers and demote the deceptive answers. The user preference graph is constructed based on their answer evaluation, such as "helpful" voting or "best answer" selection.

3 Proposed Features

We first view the deceptive answer prediction as a binary-classification problem. Two kinds of features, including textual features and contextual features, are described as follows:

3.1 Textual Features

We first aim to predict the deceptive answer by analyzing the answer content. Several textual features are extracted from the answer content:

3.1.1 Unigrams and Bigrams

The most common type of feature for text classification is the bag-of-word. We use an effective feature selection method χ^2 (Yang and Pedersen, 1997) to select the top 200 unigrams and bigrams as features. The top ten unigrams related to deceptive answers are shown on Table 1. We can see that these words are related to the intent for promotion.

professional	service	advice	address
site	telephone	therapy	recommend
hospital	expert		

Table 1: Top 10 Deceptive Related Unigrams

3.1.2 URL Features

Some malicious users may promote their products by linking a URL. We find that URL is good indicator for deceptive answers. However, some URLs may provide the references for the authentic answers. For example, if you ask the weather in mountain view, someone may just post the link to "http://www.weather.com/". Therefore, besides the existence of URL, we also use the following URL features:

1). Length of the URLs: we observe that the longer urls are more likely to be spam.

2). PageRank Score: We employ the PageRank (Page et al., 1999) score of each URL as popularity score.

3.1.3 Phone Numbers and Emails

There are a lot of contact information mentioned in the Community QA sites, such as phone numbers and email addresses, which are very likely to be deceptive, as good answers are found to be less likely to refer to phone numbers or email addresses than the malicious ones. We extract the number of occurrences of email and phone numbers as features.

3.1.4 Length

We have also observed some interesting patterns about the length of answer. Deceptive ones tend to be longer than authentic ones. This can be explained as the deceptive answers may be well prepared to promote the target. We also employ the number of words and sentences in the answer as features.

3.2 Contextual Features

Besides the answer textual features, we further investigate various features from the context of the target answer:

3.2.1 Question Answer Relevance

The main characteristic of answer in Community QA site is that the answer is provided to answer the corresponding question. We can use the corresponding question as one of the context features by measuring the relevance between the answer and the question. We employ three different models for Question-Answer relevance:

Vector Space Model

Each answer or question is viewed as a word vector. Given a question q and the answer a, our vector model uses weighted word counts(e.g.TF-IDF) as well as the cosine similarity $(q \cdot a)$ of their word vectors as relevant function (Salton and McGill, 1986). However, vector model only consider the exact word match, which is a big problem, especially when the question and answer are generally short compared to the document. For example, Barack Obama and the president of the US are the same person. But the vector model would indicate them to be different. To remedy the word-mismatch problem, we also look for the relevance models in higher semantic levels.

Translation Model

A translation model is a mathematical model in which the language translation is modeled in a statistical way. The probability of translating a source sentence (as answer here) into target sentence (as question here) is obtained by aligning the words to maximize the product of all the word probabilities. We train a translation model (Brown et al., 1990; Och and Ney, 2003) using the Community QA data, with the question as the target language, and the corresponding best answer as the source language. With translation model, we can compute the translation score for new question and answer.

Topic Model

To reduce the false negatives of word mismatch in vector model, we also use the topic models to extend matching to semantic topic level. The topic model, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003), considers a collection of documents with K latent topics, where K is much smaller than the number of words. In essence, LDA maps information from the word dimension to a semantic topic dimension, to address the shortcomings of the vector model.

3.2.2 User Profile Features

We extract several user's activity statistics to construct the user profile features, including the level of the user in the Community QA site, the number of questions asked by this user, the number of answers provided by this user, and the best answer ratio of this user.

3.2.3 User Authority Score

Motivated by expert finding task (Jurczyk and Agichtein, 2007; Si et al., 2010a; Li et al., 2011), the second type of author related feature is authority score, which denotes the expertise score of this user. To compute the authority score, we first construct a directed user graph with the user interactions in the community. The nodes of the graph represent users. An edge between two users indicates a contribution from one user to the other. Specifically, on a Q&A site, an edge from A to B is established when user B answered a question asked by A, which shows user B is more likely to be an expert than A. The weight of an edge indicates the number of interactions. We compute the user's authority score (AS) based on the link analysis algorithm $PageRank$:

$$\mathrm{AS}(u_i) = \frac{1-d}{N} + d \sum_{u_j \in M(u_i)} \frac{\mathrm{AS}(u_j)}{L(u_j)} \qquad (1)$$

where $u_1, \ldots, u_N$ are the users in the collection, N is the total number of users, $M(u_i)$ is the set of users whose answers are provided by user u_i, $L(u_i)$ is the number of users who answer u_i's questions, d is a damping factor, which is set as 0.85. The authority score can be computed iteratively with random initial values.

3.2.4 Robot Features

The third type of author related feature is used for detecting whether the author is a robot, which are scripts crafted by malicious users to automatically post answers. We observe that the distributions of the answer-posting time are very different between general user and robot. For example, some robots may make posts continuously and mechanically, hence the time increment may be smaller that human users who would need time to think and process between two posts. Based on this observation, we design an time sequence feature for robot detection. For each author, we can get a list of time points to post answers, $T = \{t_0, t_1, ..., t_n\}$, where t_i is the time point when posting the ith answer. We first convert the time sequence T to time interval sequence $\Delta T = \{\Delta t_0, \Delta t_1, ..., \Delta t_{n-1}\}$, where $\Delta t_i = t_{i+1} - t_i$. Based on the interval sequences for all users, we then construct a matrix $X_{m \times b}$ whose rows correspond to users and columns correspond to interval histogram with predefined range. We can use each row vector as time sequence pattern to detect robot. To reduce the noise and sparse problem, we use the dimension reduction techniques to extract the latent semantic features with Singular Value Decomposition (SVD) (Deerwester et al., 1990; Kim et al., 2006).

3.2.5 Evaluation from Other Users

In the Community QA sites, other users can express their opinions or evaluations on the answer. For example, the asker can choose one of the answers as best answer. We use a bool feature to denote if this answer is selected as the best answer. In addition, other users can label each answer as "helpful" or "not helpful". We also use this helpful evaluation by other users as the contextual feature, which is defined as the ratio between the number of "helpful" votes and the number of total votes.

3.2.6 Duplication with Other Answers

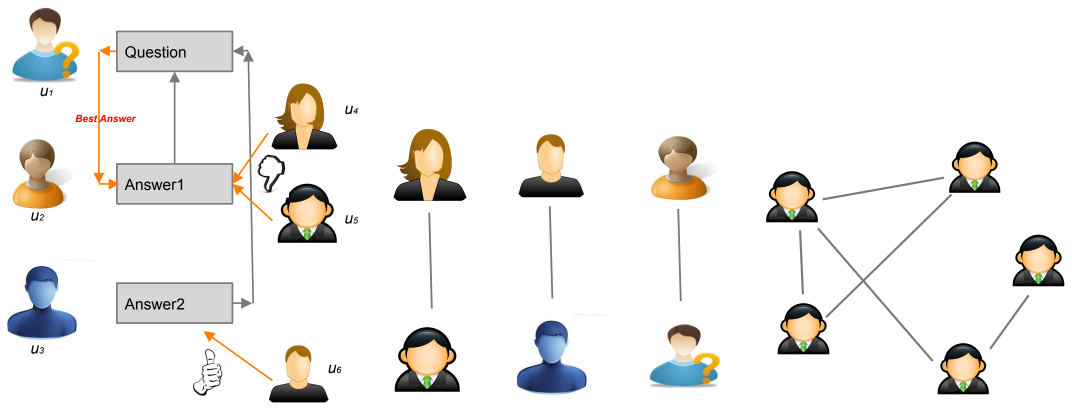
The malicious user may post the pre-written product promotion documents to many answers, or just change the product name. We also compute the similarity between different answers. If the two answers are totally same, but the question is different, these answer is potentially as a deceptive answer. Here, we don't want to measure the semantic similarity between two answers, but just measure if two answers are similar to the word level, therefore, we apply BleuScore (Papineni et al., 2002), which is a standard metric in machine translation for measuring the overlap between n-grams of two text fragments r and c. The duplication score of each answer is the maximum BleuScore compared to all other answers.

4 Deceptive Answer Prediction with User Preference Graph

Besides the textual and contextual features, we also investigate the user relationship for deceptive answer prediction. We assume that similar users tend to perform similar behaviors (posting deceptive answers or posting authentic answers). In this section, we first show how to compute the user similarity (user preference graph construction), and then introduce how to employ the user relationship for deceptive answer prediction.

4.1 User Preference Graph Construction

In this section, we propose a new user graph to describe the relationship among users. Figure 1 (a) shows the general process in a question answering



(a) Question Answering (b) User Preference Relation (c) User Preference Graph

Figure 1: User Preference Graph Construction

thread. The asker, i.e. u_1, asks a question. Then, there will be several answers to answer this question from other users, for example, answerers u_2 and u_3. After the answers are provides, users can also vote each answer as "helpful" or "not helpful" to show their evaluation towards the answer . For example, users u_4, u_5 vote the first answer as "not helpful", and user u_6 votes the second answer as "helpful". Finally, the asker will select one answer as the best answer among all answers. For example, the asker u_1 selects the first answer as the "best answer".

To mine the relationship among users, previous studies mainly focus on the asker-answerer relationship (Jurczyk and Agichtein, 2007; Liu et al., 2011). They assume the answerer is more knowledgeable than the asker. Based on this assumption, they can extract the expert in the community, as discussed in Section 3.2.3. However, we don't care which user is more knowledgeable, but are more interested in whether two users are both malicious users or authentic users. Here, we propose a new user graph based on the user preference. The preference is defined based on the answer evaluation. If two users show same preference towards the target answer, they will have the user-preference relationship. We mainly use two kinds of information: "helpful" evaluation and "best answer" selection. If two users give same "helpful" or "not helpful" to the target answer, we view these two users have same user preference. For example, user u_4 and user u_5 both give "not helpful" evaluation towards the first answer, we can say that they have same user preference. Besides the real "helpful" evaluation, we also assume the author of the answer gives the "helpful" evaluation to his or her own answer. Then if user u_6 give "helpful" evaluation to the second answer, we will view user u_6 has same preference as user u_3, who is the author of the second answer. We also can extract the user preference with "best answer" selection. If the asker selects the "best answer" among all answers, we will view that the asker has same preference as the author of the "best answer". For example, we will view user u_1 and user u_2 have same preference.

Based on the two above assumptions, we can extract three user preference relationships (with same preference) from the question answering example in Figure 1 (a): $u_4 \sim u_5, u_3 \sim u_6, u_1 \sim u_2$, as shown in Figure1 (b). After extracting all user preference relationships, we can construct the user preference graph as shown in Figure 1 (c). Each node represents a user. If two users have the user preference relationship, there will be an edge between them. The edge weight is the number of user preference relationships.

In the Community QA sites, the spammers mainly promote their target products by promoting the deceptive answers. The spammers can collaboratively make the deceptive answers look good, by voting them as high-quality answer, or selecting them as "best answer". However, the authentic users generally have their own judgements to the good and bad answers. Therefore, the evaluation towards the answer reflects the relationship among users. Although there maybe noisy relationship, for example, an authentic user may be cheated, and selects the deceptive answer as "best answer", we hope the overall user preference relation can perform better results than previous user interaction graph for this task.

4.2 Incorporating User Preference Graph

To use the user graph, we can just compute the feature value from the graph, and add it into the supervised method as the features introduced in Section 3. Here, we propose a new technique to employ the user preference graph. We utilize the graph regularizer (Zhang et al., 2006; Lu et al., 2010) to constrain the supervised parameter learning. We will introduce this technique based on a commonly used model $f(\cdot)$, the linear weight model, where the function value is determined by linear combination of the input features:

$$f(\mathbf{x}_i) = \mathbf{w}^T \cdot \mathbf{x}_i = \sum_k w_k \cdot x_{ik} \qquad (2)$$

where $\mathbf{x}_i$ is a K dimension feature vector for the ith answer, the parameter value w_k captures the effect of the kth feature in predicting the deceptive answer. The best parameters $\mathbf{w}^*$ can be found by minimizing the following objective function:

$$\Omega_1(\mathbf{w}) = \sum_i L(\mathbf{w}^T\mathbf{x}_i, y_i) + \alpha \cdot |\mathbf{w}|_F^2 \qquad (3)$$

where $L(\mathbf{w}^T\mathbf{x}_i, y_i)$ is a loss function that measures discrepancy between the predicted label $\mathbf{w}^T \cdot \mathbf{x}_i$ and the true label y_i, where $y_i \in \{+1, -1\}$. The common used loss functions include $L(p, y) = (p - y)^2$ (least square), $L(p, y) = \ln(1 + \exp(-py))$ (logistic regression). For simplicity, here we use the least square loss function. $|\mathbf{w}|_F^2 = \sum_k w_k^2$ is a regularization term defined in terms of the Frobenius norm of the parameter vector $\mathbf{w}$ and plays the role of penalizing overly complex models in order to avoid fitting.

We want to incorporate the user preference relationship into the supervised learning framework. The hypothesis is that similar users tend to have similar behaviors, i.e. posting deceptive answers or authentic answers. Here, we employ the user preference graph to denote the user relationship. Based on this intuition, we propose to incorporate the user graph into the linear weight model with graph regularization. The new objective function is changed as:

$$\Omega_2(\mathbf{w}) = \sum_i L(\mathbf{w}^T\mathbf{x}_i, y_i) + \alpha \cdot |\mathbf{w}|_F^2 + \beta \sum_{u_i,u_j \in N_u} \sum_{x \in A_{u_i}, y \in A_{u_j}} w_{u_i,u_j}(f(x) - f(y))^2 \qquad (4)$$

where N_u is the set of neighboring user pairs in user preference graph, i.e, the user pairs with same preference. A_{u_i} is the set of all answers posted by user u_i. w_{u_i,u_j} is the weight of edge between u_i and u_j in user preference graph. In the above objective function, we impose a user graph regularization term

$$\beta \sum_{u_i,u_j \in N_u} \sum_{x \in A_{u_i}, y \in A_{u_j}} w_{u_i,u_j}(f(x) - f(y))^2$$

to minimize the answer authenticity difference among users with same preference. This regularization term smoothes the labels on the graph structure, where adjacent users with same preference tend to post answers with same label.

5 Experiments

5.1 Experiment Setting

5.1.1 Dataset Construction

In this paper, we employ the Confucius (Si et al., 2010b) data to construct the deceptive answer dataset. Confucius is a community question answering site, developed by Google. We first crawled about 10 million question threads within a time range. Among these data, we further sample a small data set, and ask three trained annotators to manually label the answer as deceptive or not. If two or more people annotate the answer as deceptive, we will extract this answer as a deceptive answer. In total, 12446 answers are marked as deceptive answers. Similarly, we also manually annotate 12446 authentic answers. Finally, we get 24892 answers with deceptive and authentic labels as our dataset. With our labeled data, we employ supervised methods to predict deceptive answers. We conduct 5-fold cross-validation for experiments. The larger question threads data is employed for feature learning, such as translation model, and topic model training.

5.1.2 Evaluation Metrics

The evaluation metrics are $precision$, $recall$ and F-score for authentic answer category and deceptive answer category: $precision = \frac{S_p \cap S_c}{S_p}$, $recall = \frac{S_p \cap S_c}{S_c}$, and $F = \frac{2*precision*recall}{precision+recall}$, where S_c is the set of gold-standard positive instances for the target category, S_p is the set of predicted results. We also use the $accuracy$ as one metric, which is computed as the number of answers predicted correctly, divided by the number of total answers.

	Deceptive Answer			**Authentic Answer**			**Overall**
	Prec.	Rec.	F-Score	Prec.	Rec.	F-Score	Acc.
Random	0.50	0.50	0.50	0.50	0.50	0.50	0.50
Unigram/Bigram (UB)	0.61	0.71	0.66	0.66	0.55	0.60	0.63
URL	0.93	0.26	0.40	0.57	0.98	0.72	0.62
Phone/Mail	0.94	0.15	0.25	0.53	0.99	0.70	0.57
Length	0.56	0.91	0.69	0.76	0.28	0.41	0.60
All Textual Features	0.64	0.67	0.66	0.66	0.63	0.64	0.65
QA Relevance	0.66	0.57	0.61	0.62	0.71	0.66	0.64
User Profile	0.62	0.53	0.57	0.59	0.67	0.63	0.60
User Authority	0.54	0.80	0.65	0.62	0.33	0.43	0.56
Robot	0.66	0.62	0.64	0.61	0.66	0.64	0.64
Answer Evaluation	0.55	0.53	0.54	0.55	0.57	0.56	0.55
Answer Duplication	0.69	0.71	0.70	0.70	0.68	0.69	0.69
All Contextual Feature	0.78	0.74	0.76	0.75	0.79	0.77	0.77
Textutal + Contextual	0.80	0.82	0.81	0.82	0.79	0.80	0.81

Table 2: Results With Textual and Contextual Features

5.2 Results with Textual and Contextual Features

We tried several different classifiers, including SVM, ME and the linear weight models with least square and logistic regression. We find that they can achieve similar results. For simplicity, the linear weight with least square is employed in our experiment. Table 2 shows the experiment results. For textual features, it achieves much better result with unigram/bigram features than the random guess. This is very different from the answer quality prediction task. The previous studies (Jeon et al., 2006; Song et al., 2010) find that the word features can't improve the performance on answer quality prediction. However, from Table 1, we can see that the word features can provide some weak signals for deceptive answer prediction, for example, words "recommend", "address", "professional" express some kinds of promotion intent. Besides unigram and bigram, the most effective textual feature is URL. The phone and email features perform similar results with URL. The observation of length feature for deceptive answer prediction is very different from previous answer quality prediction. For answer quality prediction, length is an effective feature, for example, long-length provides very strong signals for high-quality answer (Shah and Pomerantz, 2010; Song et al., 2010). However, for deceptive answer prediction, we find that the long answers are more potential to be deceptive. This is because most of deceptive answers are well prepared for product promotion. They will write detailed answers to attract user's attention and promote their products. Finally, with all textual features, the experiment achieves the best result, 0.65 in accuracy.

For contextual features, we can see that, the most effective contextual feature is answer duplication. The malicious users may copy the prepared deceptive answers or just simply edit the target name to answer different questions. Question-answer relevance and robot are the second most useful single features for deceptive answer prediction. The main characteristics of the Community QA sites is to accumulate the answers for the target questions. Therefore, all the answers should be relevant to the question. If the answer is not relevant to the corresponding question, this answer is more likely to be deceptive. Robot is one of main sources for deceptive answers. It automatically post the deceptive answers to target questions. Here, we formulate the time series as interval sequence. The experiment result shows that the robot indeed has his own posting behavior patterns. The user profile feature also can contribute a lot to deceptive answer prediction. Among the user profile features, the user level in the Community QA site is a good indicator. The other two contextual features, including user authority and answer evaluation, provide limited improvement. We find the following reasons: First, some malicious users post answers to various questions for product promotion, but don't ask any question. From Equation 1, when iteratively computing the

	Deceptive Answer			Authentic Answer			Overall
	Prec.	Rec.	F-Score	Prec.	Rec.	F-Score	Acc.
Interaction Graph as Feature	0.80	0.82	0.81	0.82	0.79	0.80	0.81
Interaction Graph as Regularizer	0.80	0.83	0.82	0.82	0.80	0.81	0.82
Preference Graph as Feature	0.79	0.83	0.81	0.82	0.78	0.80	0.81
Preference Graph as Regularizer	0.83	0.86	0.85	0.85	0.83	0.84	0.85

Table 3: Results With User Preference Graph

final scores, the authority scores for these malicious users will be accumulated to large values. Therefore, it is hard to distinguish whether the high authority score represents real expert or malicious user. Second, the "best answer" is not a good signal for deceptive answer prediction. This may be selected by malicious users, or the authentic asker was misled, and chose the deceptive answer as "best answer". This also demonstrates that the deceptive answer prediction is very different from the answer quality prediction. When combining all the contextual features, it can achieve the overall accuracy 0.77, which is much better than the textual features. Finally, with all the textual and contextual features, we achieve the overall result, 0.81 in accuracy.

5.3 Results with User Preference Graph

Table 3 shows the results with user preference graph. We compare with several baselines. Interaction graph is constructed by the asker-answerer relationship introduced in Section 3.2.3. When using the user graph as feature, we compute the authority score for each user with PageRank as shown in Equation 1. We also incorporating the interaction graph with a regularizer as shown in Equation 4. Note that we didn't consider the edge direction when using interaction graph as a regularizer. From the table, we can see that when incorporating user preference graph as a feature, it can't achieve a better result than the interaction graph. The reason is similar as the interaction graph. The higher authority score may boosted by other spammer, and can't be a good indicator to distinguish deceptive and authentic answers. When we incorporate the user preference graph as a regularizer, it can achieve about 4% further improvement, which demonstrates that the user evaluation towards answers, such as "helpful" voting and "best answer" selection, is a good signal to generate user relationship for deceptive answer prediction, and the graph regularization is an effective technique to incorporate the user preference graph. We also analyze the parameter sensitivity. β is the tradeoff weight for graph regularization term. Figure 2 shows the results with different values of β. We can see that when β ranges from $10^{-4} \sim 10^{-2}$, the deceptive answer prediction can achieve best results.

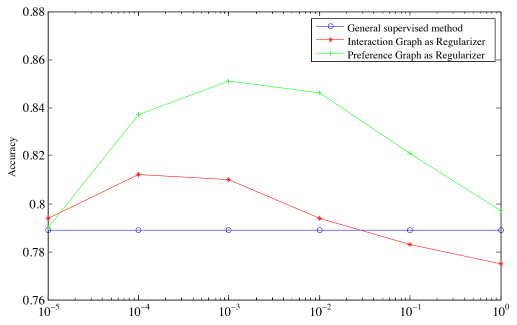


Figure 2: Results with different values of β

6 Conclusions and Future Work

In this paper, we discuss the deceptive answer prediction task in Community QA sites. With the manually labeled data set, we first predict the deceptive answers with traditional classification method. Two types of features, including textual features and contextual features, are extracted and analyzed. We also introduce a new user preference graph, constructed based on the user evaluations towards the target answer, such as "helpful" voting and "best answer" selection. A graph regularization method is proposed to incorporate the user preference graph for deceptive answer prediction. The experiments are conducted to discuss the effects of different features. The experiment results also show that the method with user preference graph can achieve more accurate results for deceptive answer prediction.

In the future work, it is interesting to incorporate more features into deceptive answer prediction. It is also important to predict the deceptive question threads, which are posted and answered both by malicious users for product promotion. Malicious user group detection is also an important task in the future.

References

Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. 2008. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 665–674, New York, NY, USA. ACM.

Jiang Bian, Yandong Liu, Ding Zhou, Eugene Agichtein, and Hongyuan Zha. 2009. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 51–60, NY, USA. ACM.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguist.*, 16:79–85, June.

S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

A. Figueroa and J. Atkinson. 2011. Maximum entropy context models for ranking biographical answers to open-domain definition questions. In *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

F. Maxwell Harper, Daphne Raban, Sheizaf Rafaeli, and Joseph A. Konstan. 2008. Predictors of answer quality in online q&a sites. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 865–874, New York, NY, USA. ACM.

Daisuke Ishikawa, Tetsuya Sakai, and Noriko Kando, 2010. *Overview of the NTCIR-8 Community QA Pilot Task (Part I): The Test Collection and the Task*, pages 421–432. Number Part I.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM CIKM conference*, 05, pages 84–90, NY, USA. ACM.

J. Jeon, W.B. Croft, J.H. Lee, and S. Park. 2006. A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 228–235. ACM.

P. Jurczyk and E. Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM CIKM conference*, pages 919–922. ACM.

H. Kim, P. Howland, and H. Park. 2006. Dimension reduction in text classification with support vector machines. *Journal of Machine Learning Research*, 6(1):37.

Fangtao Li, Minlie Huang, Yi Yang, and Xiaoyan Zhu. 2011. Learning to identify review spam. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 2488–2493. AAAI Press.

Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 497–504, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jing Liu, Young-In Song, and Chin-Yew Lin. 2011. Competition-based user expertise score estimation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 425–434. ACM.

Yue Lu, Panayiotis Tsaparas, Alexandros Ntoulas, and Livia Polanyi. 2010. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on World wide web*, pages 691–700. ACM.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29:19–51, March.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November. SIDL-WP-1999-0120.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. ACL.

Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 411–418, New York, NY, USA. ACM.

X. Si, Z. Gyongyi, and E. Y. Chang. 2010a. Scalable mining of topic-dependent user reputation for improving user generated content search quality. In *Google Technical Report*.

Xiance Si, Edward Y. Chang, Zoltán Gyöngyi, and Maosong Sun. 2010b. Confucius and its intelligent disciples: integrating social with search. *Proc. VLDB Endow.*, 3:1505–1516, September.

Young-In Song, Chin-Yew Lin, Yunbo Cao, and Hae-Chang Rim. 2008. Question utility: a novel static ranking of question search. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 1231–1236. AAAI Press.

Y.I. Song, J. Liu, T. Sakai, X.J. Wang, G. Feng, Y. Cao, H. Suzuki, and C.Y. Lin. 2010. Microsoft research asia with redmond at the ntcir-8 community qa pilot task. In *Proceedings of NTCIR*.

Wei Wei, Gao Cong, Xiaoli Li, See-Kiong Ng, and Guohui Li. 2011. Integrating community question and answer archives. In *AAAI*.

Y. Yang and J.O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 412–420. MORGAN KAUFMANN PUBLISHERS.

Tong Zhang, Alexandrin Popescul, and Byron Dom. 2006. Linear prediction models with graph regularization for web-page categorization. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 821–826. ACM.