SEMANTICS-ENHANCED PRIVACY RECOMMENDATION FOR SOCIAL NETWORKING SITES

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Qingrui Li

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

February 2012

Fargo, North Dakota

North Dakota State University
Graduate School

Title

SEMANTICS-ENHANCED PRIVACY RECOMMENDATION FOR SOCIAL NETWORKING SITES

By

Qingrui Li

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

Master of Science

SUPERVISORY COMMITTEE:

Dr. Juan Li

Chair

Dr. Kendall E. Nygard

Dr. Changhui Yan

Dr. Samee Khan

Approved by Department Chair:

02/28/2012
Date

Kenneth Magel
Signature

ABSTRACT

Privacy protection is a vital issue for safe social interactions within social networking sites (SNS). Although SNSs such as MySpace and Facebook allow users to configure their privacy settings, the task is difficult for normal users with hundreds of online friends. In this paper, I propose an intelligent semantics-based privacy configuration system, named SPAC, to automatically recommend privacy settings for SNS users. SPAC learns users' privacy configuration patterns and make predictions by utilizing machine learning techniques on users' profiles and privacy setting history. To increase the accuracy of the predicted privacy settings, especially in the context of heterogeneous user profiles, I enhance privacy configuration predictor by integrating it with structured semantic knowledge. This allows SPAC to make inferences based on additional source of knowledge, resulting in improved accuracy of privacy recommendation. Our experimental results have proven the effectiveness of our approach.

Keywords: social network; privacy; semantics; ontology; recommendation

ACKNOWLEDGMENTS

The work described in this master paper would not have been possible without the assistance, guidance, advice, and support of many people. In particular, I would like to express my sincere gratitude to my advisor, Dr. Juan Li, who helped me towards my academic as well as my personal success. I appreciate her willingness to help students and wonderful guidance in doing research, which made my research life in NDSU continuously progressive and quite enjoyable.

I would like to thank my thesis committee, Dr. Kendall E. Nygard, Dr. Changhui Yan, and Dr. Samee Khan, who spared much time in their tight schedules and provided insightful and constructive comments on this paper. Their insights proved to be quite helpful in extending and deepening my knowledge in this research field. I benefited a lot from their valuable advices.

My greatest thanks go to my dearest family, for their endless support and love throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

Social networking sites are a type of virtual community that has grown tremendously in popularity over the past few years. Social networking sites (e.g., Facebook, MySpace, Twitters, etc.) have attracted billions of users and the number of users is still fast increasing. When people join social networking sites, they begin by creating a profile, then make connections to existing friends as well as those they meet through the site.

Privacy protection is an important issue in social networking. As users publish their private information (e.g., name, birthday, hometown, religion, ethnicity, and personal interest) on social network websites, it is necessary to enforce appropriate protection on this sensitive information. Indeed, many social networking websites provide interfaces for users to configure their privacy settings. For instance, Facebook (facebook.com) supports privacy settings with various access levels, for sharing information with everyone, friends only, friends of friends, or a set of specified individuals. It also supports customizing of privacy setting rules on shared content, including photos, profiles, family and relationships, posts, etc., for users of various settings (everyone, friends of friends, friends only, specific individuals); Wordpress (wordpress.com) allows privacy settings of blog postings as visible to everyone, blocked from search engines but allow normal visitors, and visible to chosen users; Twitter (twitter.com) only allows two types of privacy settings, tweets will be available either publicly or to chosen visitors.

Previous study has shown that average users have difficulties in understanding privacy policies and reasoning holistically about privacy mechanisms [1, 2]. Therefore, in many cases, users may use the system default setup for their privacy configuration. However, the default privacy configuration may not be able to meet all users' needs. Indeed, it has been shown that many users (e.g., Facebook members) reveal a lot of information about themselves, without awareness of privacy options or who can actually view their profile [1]. Furthermore, the available privacy configuration interfaces do not allow users to easily specify their access control requirements, in that they are either too restrictive or too loose [3]. For example, as Facebook allows setting up privacy preference to friends by treating "all friends" as a single object, Facebook users cannot easily share some of their posts with a subset of their friends, while keeping some other postings private, and sharing the remaining to the public, except setting the privacy to every single friend one by one. On the other hand, it is tedious to construct and maintain privacy policies on very fine granularity for every single friend and data item (e.g., photos and blog postings). Such activity is not acceptable in practice for most SNS users who have tremendous online connections (e.g., Facebook user has 130 friends on average [4]).

A seemly straightforward approach is to let users predefine a few privacy configuration rules, so that the privacy settings of new friends will be automatically determined by following these rules. However, it is challenging since: (1) as many users lack sufficient understanding of privacy policies, asking users to define privacy rules is difficult, and (2) it is hard to define rules for the future friends, or the data contents that have not been produced and shared in the network yet.

To address these challenges, I design an intelligent privacy configuration system, named *SPAC*, for social networking sites. *SPAC* will learn users' privacy configuration patterns by utilizing machine learning techniques on users' profiles and privacy setting history. Based on the patterns, *SPAC* will recommend privacy configuration for either unlabeled existing friends or new friends.

To improve the accuracy of the predicted privacy settings, I take semantics of data items and user profiles into consideration. Introducing semantics into prediction provides additional clues about the underlying reasons for which a user may or may not allow access for particular items (something that is implicit and hidden to traditional methods without semantics awareness). This, in turn, allows *SPAC* to make inferences based on this additional source of knowledge, possibly improving the accuracy of predictions. In particular, I propose a novel semantics-enhanced k-Nearest Neighbors (k-NN) classification algorithm to predict the privacy settings for unlabeled or new friends based on the historical privacy setting data of the user. Our approach integrates the ontology knowledge hidden in the heterogeneous friends' profile data with the similarity calculation between these friends so as to provide more realistic similarities for the classification.

The paper is organized as follows. Section II discusses the related work. Section III introduces the system design and details of our semantics enhanced k-NN algorithm. Section IV presents our experimental results. Section V concludes the paper and lists possible future work.

CHAPTER 2. BACKGROUND AND RELATED WORK

In this chapter, I first briefly present the necessary background knowledge that is important to our work. Thereafter, I overview the related research as a comparison with our work.

2.1. Recommendation System and Classification

People rely on recommendations in their daily life. The traditional media that I can get recommendations vary from spoken words, reference letters, reports, surveys, travel guides, and so on. Nowadays, more and more recommender systems are created to assist and augment this social process and help people in an efficient, effective, and creative manner. The information overload in modern age and the customization trend coupled with E-commerce and online societies have led to an increasing need for recommender systems - a personalized information filtering technology used to either identify interesting items among a large number of choices or provide some automatic services for mitigating user effort. An excellent quality of personalized recommendations would guarantee customer satisfaction and loyalty.

There have been many existing techniques of building a recommender system. One successful and well-know approach of them is machine learning. In machine learning, classification is a widely applied method for assigning an unknown input data instance into one of a given number of categories or classes. Classification normally refers to a supervised procedure, i.e. a procedure that learns to classify new instances based on learning from a training set of instances that have been properly labeled by hand with the correct classes. An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. I will describe two classical classifiers that are related to our research as follows.

2.1.1. k Nearest Neighbors (k-NN) Classifier

k-NN Classifier [39, 40] is a simple machine learning algorithm. It is a method of classifying objects based on closest training examples. The training examples are normally vectors in a multidimensional feature space, each with a class label. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the classification phase, k is a user-defined constant, and an unlabeled object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. The typical process of k-NN algorithm includes the following steps:

- Select a suitable value for k;
- Determine a suitable “near” concept (a distance or similarity);
- Find the k nearest neighbor set of the unclassified sample;
- Find the plurality class in the nearest neighbor set;
- Assign the plurality class as the predicted class of the sample.

Usually, Euclidean distance is used as the distance metric; however the distance can be defined and calculated differently based on the specific properties and requirements of specific applications.

2.1.2. Decision Tree Classifier

A decision tree classifier [17, 18] is another simple yet widely used classification method. In a decision tree, each leaf node is assigned a class label and represents that class. Each non-leaf internal node contains a test condition based on one selected attribute of the data object. These tests will lead an unknown object to go through the decision tree from root to a leaf node which shows the final prediction of the class of this object.

The tree construction is also the classifier's learning phase. A tree can be constructed by splitting the source set into subsets based on the selected attribute with the highest information gain [38]. This process is repeated on each derived subset in a recursive manner. The recursive partitioning is completed when all the data items of a subset have the same class value, or when splitting no longer adds value to the predictions.

Classifying an unknown object is straightforward after the decision tree has been completely constructed. Starting from the root, the condition test of each internal node will be applied to the object. Based on the test result, the appropriate branch will be followed to form the path to a leaf node. The class label associated with the leaf node will be the predicted class for this object.

2.2. Semantics Ontology

Ontology is regarded as a key technology for enabling semantics driven knowledge processing. Although there is no common definition of the term "ontology", it is normally used in computer science to represent knowledge as an explicit specification of conceptualization [29] and the relationships between those concepts. In other words, ontology is a kind of dictionary containing a shared understanding of some domain of interest [29].

Ontologies are the structural frameworks for organizing information. They are widely used in computer science and information science as a form of knowledge representation. The specific areas of application include artificial intelligence, semantic web, biomedical informatics, information architecture and so on.

The main components of ontologies include:

- Individuals: the basic instances or objects in this world
- Classes: sets or collections of individuals
- Attributes: properties between individuals and their values
- Relations: ways in which classes and individuals can be related to one another

Figure 1 shows an example of a simple ontology. Each node in this figure defines a basic object, an individual of the "animal" world. These nodes are further categorized into different classes which are marked by different colors; their corresponding relationships are described by the edges between those entities.

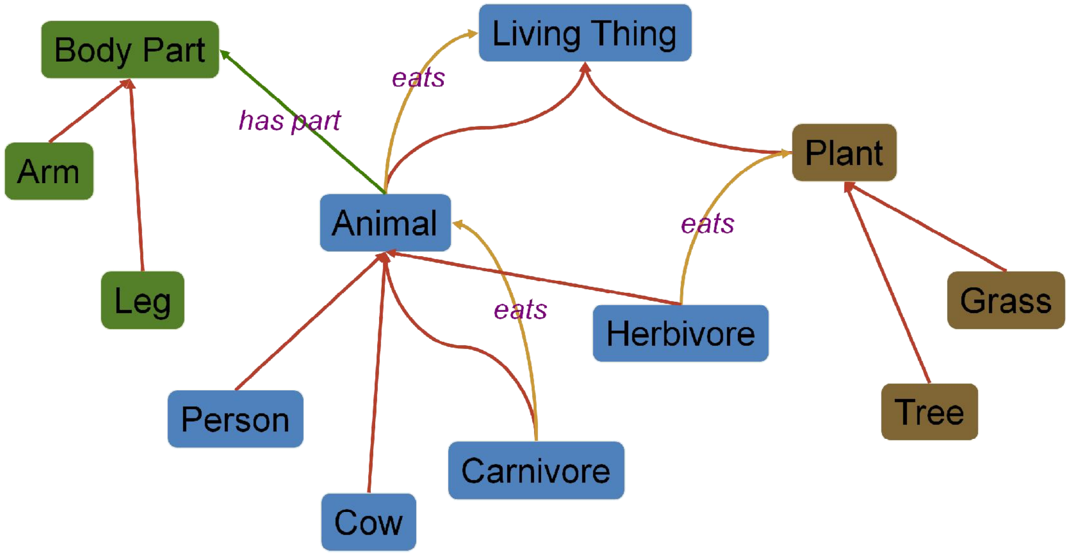

Figure 1 A simple ontology - Animals

2.3. Related Work

The development of usable tools for protecting personal data in social media is an emerging problem that caught much attention recently [1, 5, 6, 7, 8]. Several recent papers have proposed solutions to help users specify access control on social networking sites. Adu-Oppong et al. [9] and Danezis [10] tried to simplify the privacy policies by partitioning the user’s friends into lists based on automatically extracted network communities. However, neither of these works was well evaluated by experiments. Maximilie et al. [11] proposed a methodology for quantifying the risk posed by users' privacy settings. However, the quantified risk score does not help users in creating privacy setting rules. Kruk et al. [12] present an identity management solution based on social works. Each user has a control on his/her profile and social networking information. Each friend will be assigned a level value to indicate his/her closeness. User access is assigned to a resource when the friendship level and the distance bet ween the resource owner and the service requester meet required constraints. Ali et al. [13] proposed a social access control (SAC) strategy based on multi-level security model. SAC classified the data objects and subjects in hierarchical levels based on trust levels and then it could manage access controlled accordingly. Access to a data object is controlled using the trust values of subjects and objects. Carminati et al. [14] proposed a discretionary access control model for online social networks. The model allows the specification of access rules for online resources, where authorized users are denoted in terms of the relationship type, depth, and trust level existing between nodes in the network. In these work, semantics in the social networks is largely ignored.

Carminati et al. designed an access control system that uses semantic web technologies to represent much richer forms of relationships among users, resources and actions [3]. For example, by using OWL reasoning tools, a “very close” friend will be inferred as a “friend”. Therefore, anything that is accessible by friends could be also accessible by a “close friend”. Masoumzadeh et al. [15] proposed an access control ontology to capture the information semantics in an social networking site. The access control policies are defined as rules and enforced based on the access control ontology. In our work, I respect the semantics information in social networks too. Different from their work, I assume the user-specified access control rules are not sufficient to address users' privacy requirements. Therefore, the system has to infer hidden rules and perform automatic predictions based on users' access control history.

The work that is most related to ours is the work by Fang et al. [16]. They proposed a tool that can infer the model of users’ privacy preference by using machine learning techniques on users' specified input of some of their privacy preference. The preference model will then be used to configure the user’s privacy settings automatically. Our work shares the same goal of inferring user’s privacy preference models. In addition, I consider rich semantics in users’ profiles, and integrate the semantics into model inference. I explain the detailed differences between our system and their system and our advantages over theirs in the following sections.

CHAPTER 3. SYSTEM DESIGN

3.1. Preliminaries

Before presenting the detailed system design, I first introduce some preliminaries of the system:

User profiles: I assume every user has specified a profile. A user profile is a list of identifying information, such as name, birthday, hometown, religion, ethnicity, and personal interest.

Data items: Data items in social networks can be of various types; they can be user profile information (e.g., age and gender), photo images, blog entries, audio, and video files.

Privacy settings: A user's privacy setting describes her requirement to share data items with each of her friends. Suppose that a particular user has friend set *F*, and let *I* denote her data items. The users privacy settings can be expressed as a $|F|\times|I|$ matrix, where each entry is valued "*permit*" or "*deny*", corresponding to the setting as allowing and denying the access. Table 1 shows an example of user Dan's privacy settings.

Friends	Data items			
	Date of Birth	Diving Video	Blog Entry	. . .
A	Permit	Deny	Deny	. . .
B	Deny	Permit	Deny	. . .

TABLE 1 AN EXAMPLE OF PRIVACY SETTING

3.2. System Overview

SPAC is a classification system in nature. In the context of privacy settings, the classification process can be described as finding a function M: $F \rightarrow \{0, 1\}$, where F is the friend's feature vector that is related to the user's privacy setting, while *0* or *1* refers to the user's decision on whether permitting or denying the friend's access to the user's certain private data item. Finally, each friend of the user should be configured as "*permit*" or "*deny*" for each of the user's data items. Our goal is to predict the class labels for those friends whose privacy settings are undefined yet. *SPAC* predicts access settings for unlabeled friends by using the existing settings of labeled friends. A friend is represented as a vector of features extracted from the social networking site. I will describe the details of the feature extraction in Section C.

Although various classical classifiers, such as Naive Bayes, Decision Tree, and Nearest Neighbors, can be used to fulfill the classification functionality, I selected Nearest Neighbors as our classifier because of the following two reasons: (a) This simple and easy-to-implement method can yield competitive results even compared to the most sophisticated machine learning methods [33]. (b) The similarity based distance calculation used by the algorithm is a perfect point of penetration to combine the semantics ontology with the classifier.

Figure 2 depicts the architecture of the proposed *SPAC* system. The input of the predictor includes four parts: (1) user's shared data items, for which the user would like to grant access permissions to only partial friends. A tool like *SPAC* with automatic recommendation functionality will greatly reduce the user effort of configuration, (2) features of user's friends, which are extracted from the social networking site automatically, (3) limited amount of user's configuration effort/history, which will be used by the predictor as the training data, and (4) ontology (or ontology-like) knowledge of friends' features. The output of the predictor is a set of privacy settings recommended for the user's unlabeled friends. Table 2 shows an example of a user's friends list with extracted feature values and the corresponding class labels of the user's privacy settings for the item "*Relationship Status*". (Note: The "?" in the column "Class Label" means a friend is not yet labeled by the user).

The fundamental assumption of our SPAC system is that users tend to grant accordant access control to similar friends, no matter whether they conceive their privacy settings base on explicit or implicit rules. Also, I assume that the labels explicitly assigned to friend by the user are always correct.

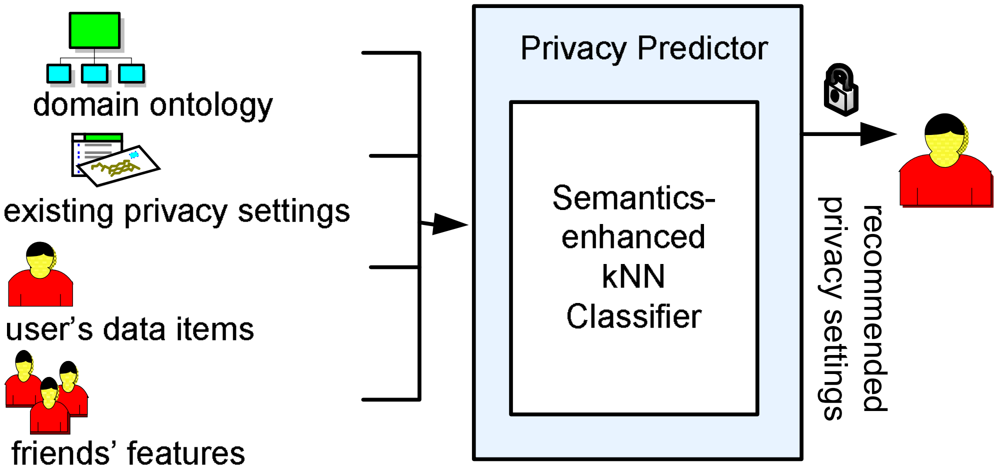
domain ontology
existing privacy settings
user's data items
friends' features
Privacy Predictor
Semantics-enhanced kNN Classifier
recommended privacy settings

Figure 2 SPAC system architecture

Friend	Features					Class Label (Relationship Status)
	Community	Age	Gender	Location	…	
A	C01	20	F	Fargo	…	Permit
B	C201	23	M	ND	…	Deny
C	C1	30	F	NYC	…	Permit
D	C21	50	M	North Dakota	…	?

TABLE 2 AN EXAMPLE OF A USER'S PRIVACY SETTINGS FOR "RELATIONSHIP STATUS"

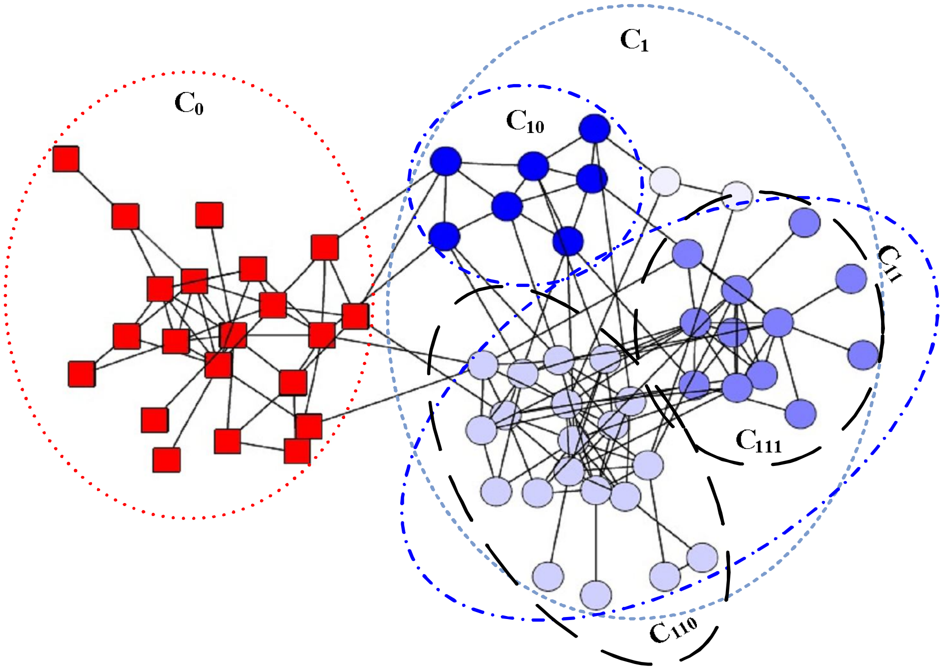

Figure 3 Example of community structure of a user's friend neighborhood

3.3. Friend's Features Used by the Classifier

To accurately classify a user's unlabeled friends, it is important to select a good set of friends' features. In social networks, the so-called community structures, i.e. relatively densely connected sub-networks [36], is fundamentally important for understanding the social relationships between social network participants [37]. An example of extracted community structure is shown in Figure 3. The community structure between people has been proved to be an effective feature for classifying social connections in SNS [16]. Therefore, in our work, I choose the community structure as an important feature. To discover communities, I adopted the hierarchical community discovery approach [16], which is based on the edge betweenness algorithm [28].

In their system, Fang et al. [16] extended the feature list with the community structure features. In particular, they added all the discovered community structures to the feature vector. However, this approach has two major shortcomings: First, the community structure may be large and complex. Then the system has to maintain a long list of community features. This will incur large overhead in terms of space and computation. Moreover, the communities discovered may have a hierarchical relationship. That is to say, community features are not independent with each other. Therefore, adding the discovered communities as independent features will miss the inherent relationships between them; and this also violates the generally admitted "minimum redundancy" principle in feature selection [20, 34].

To solve the aforementioned problems, I respect the inherent hierarchical relationships of the community structure with a predicting algorithm that supports complex features. Therefore, I only need to maintain one instead of multiple community features in the friend feature vector.

Besides the community structure, I also collect other friend's features, such as gender, age, location, hometown, education history, employment history, interests, religious view, and political views. I do not want to overlook any of these features, as they may be related to the hidden rules of a user's privacy settings. For example, some younger users might want to share certain photos with only friends of similar ages but not the elder generations, and some share their postings strictly within people who have the same religious or political views. I attach weight factors to individual features and give default values based on our experimental result. The above mentioned factors can be further configured by the user based on their personal preferences.

In practice, people tend to describe their features differently. For instance, two friends both working as "assistant professor" in the university, may describe their occupation differently: one user may use the term "faculty", while the other may describe herself as "professor". In another example, a friend interested in knowing more about "China", and another friend interested in knowing more about China's capital – Beijing. They may be assumed to have similar interests. Therefore, it is important for our system to overcome differences in vocabularies and support inference mechanisms. I will present our semantics-enhanced classification to support this functionality in the following section.

3.4. Semantics-enhanced k-NN Classification

3.4.1. Motivation

Studies have shown that general users have difficulty reasoning about privacy and security policies [1, 2]. Therefore, it is unrealistic to require the general users to define privacy rules for their data items in a SNS. This issue motivates researchers to study how to automatically learn privacy principles that users generally follow from their privacy setting history [16]. Fang et al. [16] have demonstrated that rule-based classification method, Decision Tree [17, 18], can be used as an effective tool for automatic privacy configuration. However, their approach has two major limitations:

- If there is no explicit rule corresponding to the privacy setting or if the rule is related to dynamically changed combinations of multiple features, then the rule-based decision tree, which uses the information gain on singular feature vectors, will not be appropriate to predict the settings.
- The collected friends' features may be heterogeneous in representations, even if they are semantically related. For example, the hometown "North Dakota" of one friend might be abbreviated as "ND" in another friend's profile. Or in another example, Interests might be configured by four friends as "soccer", "basketball", "guitar", and "saxophone", which are completely irrelevant to a normal classifier. However, the fact that the first two friends are sports lovers and the

last two friends are music lovers is likely to be the configuration rule for a certain data item of the user. Note that this limitation is not restricted in Decision Tree classification only, but for almost all classical classifiers without semantics awareness.

I propose a semantics-enhanced k-NN (s-k-NN) method to overcome the aforementioned two problems. The proposed k-NN method classifies according to the similarity measured based on a feature vector instead of one singular feature at each step. This overcomes the first problem of the decision tree-based classifier. Moreover, ontology or ontology-like semantic knowledge is used for the similarity calculation, so that the classifier is able to "perceive" hidden rules and give more accurate predictions. This will address the second problem of the decision tree-based classifier.

3.4.2. Methodology

The k-NN classification algorithm uses a majority vote based on the K nearest neighbors of the target object (i.e., the object to be classified) to determine the class label of the object. The performance of a k-NN classifier is primarily determined by the applied distance (or similarity) metric [21]. Various similarity measurement metrics (e.g., cosine similarity [22, 23], Pearson correlation [24, 25], Conditional Probability-Based Similarity [26, 27]) have been proposed to measure the similarity between items. However, I cannot directly apply any of these methods to our problem because (1) they cannot deal with heterogeneous feature types; while our feature vector contains both nominal (e.g. interest, location, college) and numerical (e.g. age, birthday) features. (2) They cannot measure the similarity of feature values which are semantically related but literally unrelated. In our work, I utilize domain ontologies and knowledge to facilitate the similarity measurement.

Equation (1) shows the definition of the similarity between two friends. I first measure the semantic similarity between the pair of values for each feature, and then linearly combine them to get the similarity between two friends. The result is normalized to a value between 0 and 1. Because features may have different significance when used by users to conceive their privacy preference [16], I assign different weight factors w to different features in the linear combination. For features that have significant influences, like the community structure feature, I assign them higher weight factors. Advanced users who have basic knowledge of the system can also configure the weight factors according to their own judgment.

$$SIM_{friend}(friend_x, friend_y) = \frac{\sum_{i=0}^{n} w_i SIM_{feature}(feature_i\ of\ x,\ feature_i\ of\ y)}{\sum_{i=0}^{n} w_i} \quad (1)$$

In Equation (1), function $SIM_{feature}$ is to measure the similarity between two feature values of the same feature. For example, *friend x* has interest "*NBA*", *friend y* has interest "*NBL*", what's the similarity between "*NBA*" and "*NBL*"? Another example, the value of community feature for *friend x* is C_{21}, the corresponding feature value for *friend y* is C_2, what is the similarity between this two values? Again, what's the similarity between the age values of *friend x* who is *23* and *friend y* who is *32*? I utilize the domain ontology (or ontology-like knowledge) to facilitate accurate similarity measurement. In particular, I divide the features into three main categories: community structure feature, nominal profile feature, numerical profile feature. The examples of the possible ontologies for these three feature types are shown in Figure 4.

Individual features are projected to nodes in the ontology graphs. I should note the ontology graph does not need to be a tree, although tree-like ontology is very important and common. Ontology graph can be any DAG representing all kinds of relationships between concepts. In the example ontology graphs shown in Figure 4, equivalent concepts are drawn in the same node (e.g., RolePlayGame and RPG are equivalent concepts). The similarity between values of the same feature then can be computed with a distance-based approach [19] over the ontology graph. The basic idea is to identify the shortest path between two concepts in terms of the number of edges and then translate that distance into semantic distance. I also consider the depth of the nodes in the ontology hierarchical graph to improve the accuracy. In particular, concept nodes sharing common ancestor (i.e., common more general concept) at the lower level should be more similar than those whose common ancestor is at a higher level. In other words, similar concepts should have long common path and deep common ancestor in the tree.

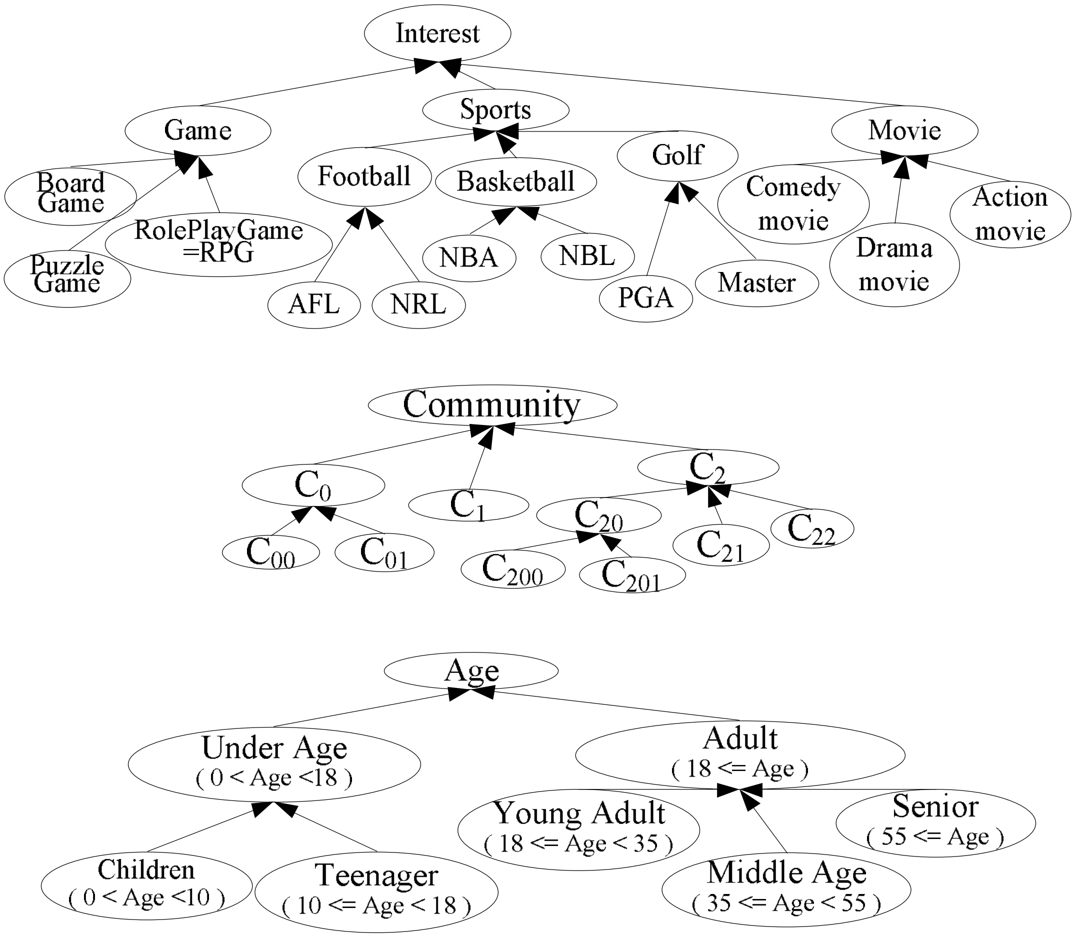

Figure 4 Example ontologies

Now go back to Equation (1), after projection, the similarity between two feature values of two friends, *x* and *y*, can be converted to the semantic similarity between two concepts C_x and C_y in an ontology graph, that is to say,

$$SIM_{feature}(feature_i\ of\ x,\ \ feature_i\ of\ y) = \ SIM_{sem}\big(C_x, C_y\big)$$

The similarity between two concepts C_x and C_y is defined in Equation (2).

$$SIM_{sem}(C_x, C_y) = \frac{1}{2}\left(\frac{\sum_{\mathrm{i}\in\mathrm{path}(\mathrm{C_{root}to\ C_p})} w_i dis(C_i, C_{i+1})}{\sum_{\mathrm{i}\in\mathrm{path}(\mathrm{C_{root}\ to\ C_x})} w_i dis(C_i, C_{i+1})} + \frac{\sum_{\mathrm{j}\in\mathrm{path}(\mathrm{C_{root}to\ C_p})} w_j dis(C_j, C_{j+1})}{\sum_{\mathrm{j}\in\mathrm{path}(\mathrm{C_{root}\ to\ C_y})} w_j dis(C_j, C_{j+1})}\right) \quad (2)$$

where C_p is the common ancestor of C_x and C_y in the hierarchical ontology graph, C_{root} is the root of the tree, C_i is C_{i+1}'s parent, and w_i is the weight of edge presented as a distance factor. It is easy to prove that SIM_{sem} is in the range of [0, 1].

Note that there is a special type of features which have binary values "0" or "1", indicating whether a friend belongs to a group of users. This type contains features such as Facebook group, events, and tagged photos. For this kind of binary features, the similarity between features can be converted to the similarity between two binary values V_x and V_y, i.e.,

$$SIM_{feature}(feature_i\ of\ x,\ feature_i\ of\ y) = SIM_{bin}(V_x, V_y)$$

The following simple formula is defined to compute the similarity between two values:

$$SIM_{bin}(V_x, V_y) = V_x \wedge V_y \qquad (3)$$

I can see from Equation (3) that if both values of the binary feature are 1, the result similarity is 1; otherwise it is 0. In other words, when both friends are the members of the same group, they are similar - if any of them does not belong to this group, they are irrelevant in terms of this feature.

Combining Equations (1), (2), and (3), I can compute the similarity for any two friends. Thus, for the target friend who needs to be labeled, the system will find the k-nearest friends who have been labeled with the user's access control decision. Based on the majority of the settings of the similar friends, the system will make final configuration recommendation for the unlabeled friend.

CHAPTER 4. EXPERIMENTS

A set of experiments were performed to verify the effectiveness of the proposed approach of automatic privacy configuration for social networking sites. I used one important metric, the accuracy-effort ratio, to illustrate how accurate the system can predict the user's privacy settings based on the same amount of existing settings (i.e. user effort). Moreover, I analyzed how different values of the parameters (including the value *k* of the k-NN algorithm and the set of weight factors in our similarity calculation formula) affect the performance of the *SPAC* predictor. The analysis assisted us providing reasonable default values for these parameters and will greatly improve the practicality of the proposed system.

4.1. Experimental Setup

Our experiments were performed based on self-generated data which were used to simulate relations and activities of real online social network users.

In particular, I created an artificial social network with 500 nodes, each stands for a social networking site user. I attached different user profiles for these users and randomly generated the network connections between them. Small groups were also set up and labeled to simulate the network activities such as tagged photos and fan groups. For 50 of the users, I specified their privacy settings by using various rules with different complexities. The number of friends for each of these users is calculated according to her social network connections. On average, each user has 116 friends here. For each data item of a certain user, I used part of the privacy setting data as training data and the rest of the data as testing data to evaluate our *SPAC* system based on the semantics-aware k-NN classifier proposed in Section III.D.

As mentioned in Section III.C, there are two main types of features in our system: community structure and profile data. The community feature is extracted by employing the implementation of iGraph library [30] based on edge-betweenness [28] algorithm. The profile features of the participants are extracted directly from Facebook upon their agreement. The candidate profile features include: gender, age, location, hometown, university, high school, employer, relationship status, religion, political view, interests, Facebook groups, events, and tagged photos.

4.2. Experiments and Results

4.2.1. Experiment 1: System Performance: Accuracy-Effort Tradeoff

Our first experiment was to evaluate the *accuracy-effort* relationship of our *SPAC* system. Here *accuracy* is defined as the average of all results obtained from experiments on all combinations of *users* and *privacy data items*; while *effort* is defined as the number of friends the user has labeled before the system starts to give recommended settings.

I used the well-known n-fold cross-validation [32] to conduct these tests. The n-fold cross-validation repeatedly partitions the given data into disjoint training and test datasets, and individual tests are executed on these combinations of datasets respectively to get the average accuracy or error rate. In this experiment, *n* was calculated based on the value of user effort (i.e. the labeled friends), which is used as training data of our classifier. Formally, for a certain privacy information item of user *X*, *n* is defined as:

$$n = \begin{cases} \dfrac{|Friends_X|}{|Effort_X|}, & |Effort_X| \leq \dfrac{|Friends_X|}{2} \\ \dfrac{|Friends_X|}{|Friends_X| - |Effort_X|}, & otherwise \end{cases} \qquad (4)$$

where $Friends_X$ is the set of user *X*'s friends and $Effort_X$ is the set of user *X*'s labeled friends. Note that in this experiment the training set might be smaller than the testing set, which is the reason I have to use different formulas to calculate *n* here.

In the experiment, the *k* value in our semantics enhanced k-NN classifier (s-k-NN) is decided by the following formula:

$$\text{Let } i = \lfloor \sqrt{|Effort_X|} \rfloor,$$

$$k = \begin{cases} i-2, & i-2 \geq 0 \quad and \quad i \; is \; odd \\ i-3, & i-3 \geq 0 \quad and \quad i \; is \; even \\ 1\,, & otherwise \end{cases} \qquad (5)$$

The reason of this setting will be further analyzed and explained in Experiment 2.

I compared the performance of our proposed s-k-NN approach with three other approaches including: (a) a baseline solution, in which the user labels some friends, and then the rest unlabeled friends will be labeled by using the majority type of the used labels as a default setting, (b) a Decision Tree approach as an alternative classification method based on generating explicit rules. I used a well-known implementation of Decision Tree, J48, from Weka [35] open source software, (c) classical (semantics-free) k-NN in which the similarity is measured based on exact match without using any semantic knowledge.

Figure 5 shows the results of Experiment 1, where the x-axis is the user effort in terms of number of friends labeled, and y-axis is the average accuracy of the method. I can see that the proposed s-k-NN strategy outperforms the other three solutions in accuracy. Moreover, the shape of the curves demonstrates that the learning ability of our solution surpasses other methods especially when user efforts are relatively low. This advantage is just what is required in this context to save user effort.

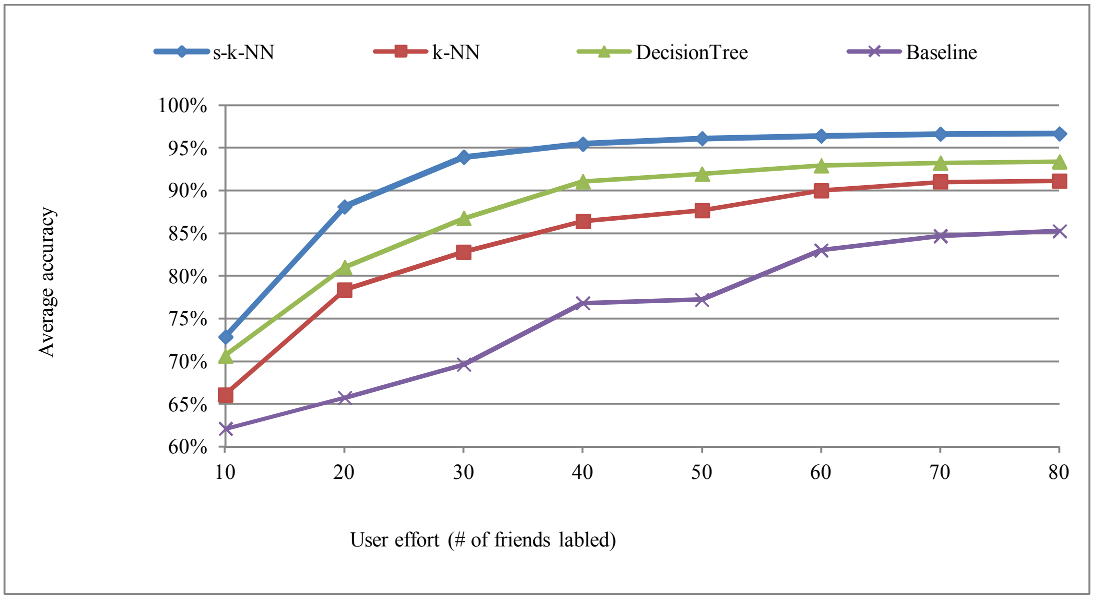

Figure 5 Comparison of accuracy-effort tradeoff

4.2.2. Experiment 2: The impact of *k* value in SPAC system

The value of *k* in k-NN algorithm is a key factor of the performance of k-NN algorithm [21]. Based on previous study, normally, the rule-of-thumb value of *k* is the square root (or less than the square root) of the total number of training data items [31]. I believe that the choice of *k* value is problem-dependent in many cases. This experiment aims to analyze the relation of *k* value and the performance of the proposed s-k-NN classifier.

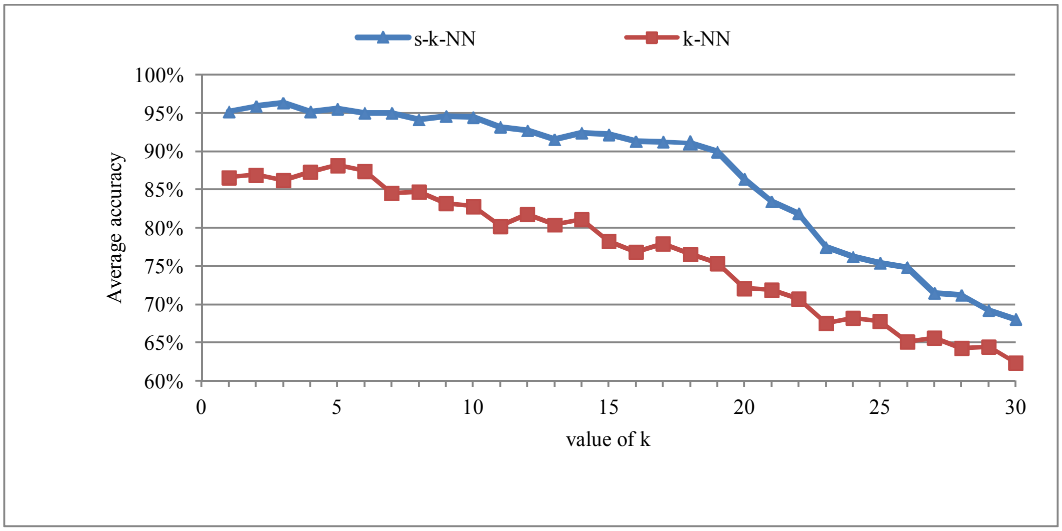

Figure 6 Impact of value *k*

In order to unify the size of data sets from different users, only data of 37 users who have more than 80 friends were used, and exactly 80 friends' data of each user were used. In the first part of this experiment, I applied 2-fold cross-validation for all of the tests. That is, the data sets were split into training sets and testing sets with a fixed proportion of 1:1 in all tests. All these settings were trying to take the average case of the collected data and simulate the reasonable user efforts in practice. Then I measured the accuracies of the system based on different settings of the *k* value. I compared s-k-NN with classical k-NN. Figure 6 summarizes the experimental results, which illustrates the following three points: (a) The s-k-NN algorithm performs better than generic k-NN consistently when the value of *k* changes. Although the performance of both methods degrade and the curves tend to converge when the value of *k* keeps increasing, the advantage of our method is obvious when *k* is relatively small. (b) The performance of s-k-NN is more stable in terms of the accuracy with the change of *k* value. (c) The value of *k* with best performance for our s-k-NN (k_{best} = 3) is smaller than the one in k-NN (k_{best} = 5).

Point (c) leads us to conduct the second part of this experiment, in which I divide the data set into training set and testing set with different ratios and perform the n-fold cross-validation tests. Table 3 summarizes the best *k* values with different training set sizes (i.e. different user efforts). The results in Table 3 show that our method requires a relatively smaller number of nearest neighbors to make a good prediction. This again demonstrates the advantageous effect of introducing the semantic knowledge into the system.

Training Set / Whole Data Set	User Effort (No. of labeled friends)	Best *k* value of s-k-NN	Best *k* value of k-NN
10%	8	1	3
20%	16	1	3
30%	24	3	3
40%	32	3	5
50%	40	3	5
60%	48	5	5
70%	56	5	5
80%	64	5	7
90%	72	5	7

TABLE 3 BEST *K* VALUES WITH DIFFERENT USER EFFORTS

Based on these results, I can recommend the default value (as shown in Equation 5) of *k* for the s-k-NN strategy in our *SPAC* system. Although this empirical formula does not guarantee it is the optimal setting every time, the result will not be far from the best based on the experimental data.

4.2.3. Experiment 3: Weights of features

This experiment is designed to evaluate the effectiveness of different features used by our classifier and to provide appropriate default settings for the weight factors of these features used in Equation (1). In this experiment, I separated the candidate features into 5 categories: (a) hierarchical *community* feature (F_{Co}), (b) important personal history including education and work experience such as *high school*, *university*, *employer*, and *hometown* (F_H), (c) culture related features like *religion* and *political view* (F_{Cu}), (d) other basic profile information such as *gender*, *age*, *location*, *interests* (F_P), and (e) binary features of user online activity including *Facebook groups*, *events*, and *tagged photos* (F_B).

Dominant Feature Category	Mean	STD	Feature Significance Order
F_{Co}	0.897	0.095	1
F_H	0.821	0.102	3
F_P	0.805	0.081	4
F_B	0.884	0.107	2
F_{Cu}	0.728	0.131	5

TABLE 4 EFFECTIVENESS OF INDIVIDUAL FEATURE CATEGORY

Proportions of weights (F_{Co}:F_H:F_P:F_B:F_{Cu})	Mean	STD	Combination Effectiveness Order
6:1:1:1:1	0.904	0.089	4
5:1:1:2:1	0.913	0.081	3
4:1:1:3:1	0.958	0.069	1
3:2:1:3:1	0.939	0.077	2
2:2:2:2:2	0.881	0.074	5

TABLE 5 EFFECTIVENESS OF COMBINATION OF FEATURE CATEGORIES

In the first part of this experiment, I assigned dominant weight factor (0.9) to each of these feature categories in turn and let the other categories equally share the remaining weights. The total sum of all features' weights is 1. Training-Testing data ratio is set as 1:3 in this experiment (Again, training set size is set smaller than testing set size to simulate practical user effort). The setting of k value uses the same formula (Equation 5) as in Experiment 1. Table 4 displays different weight combinations I took and the corresponding results of the mean and standard deviation of the accuracy.

Based on the significance order of the feature categories in the first part of this experiment, I tried a few heuristic weight combinations of these factors and performed the tests again. The corresponding test results are listed in Table 5. The result of this experiment reveals the fact that various features have different impact on the classification performance. The result confirms that community structure is an effective feature as found in the related work by Fang et al. [16]. The result also illustrates that a reasonable combination of different types of the features normally outperforms individual feature categories.

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

5.1. Conclusions

Privacy protection is indispensable for users of social networking sites. While sites like MySpace and Facebook allow users to configure their privacy settings, problems emerge when a typical user has hundreds or more online friends. On one hand, the general configuration tools provided by these sites cannot satisfy the users' specificity requirements; on the other hand, manual configuration for individual friends becomes a tedious or even impossible mission. Besides, it has shown that average users have difficulties in understanding and reasoning holistically about privacy policies.

In this paper, I proposed an intelligent semantics-based recommendation system for privacy configuration on social networking websites. I integrated semantics into nearest neighbor classification, so that various semantics information in users' profiles and hierarchical community structure can be effectively used to increase the accuracy of privacy recommendation. The results of the experiments conducted based on simulations with artificial social networking site users demonstrated the effectiveness of our approach.

I confirmed that the community structure feature has significant impact on classifying privacy setting data in social network environment. I also found out that a combined usage of different features with reasonable weights can be beneficial to the classification. As the feature effectiveness might be user or context dependant, it could be useful to maintain user-configurable weight factors for the features of classifiers in similar systems.

5.2. Future Work

In this paper, I integrated semantics knowledge with the k-NN classification method. Our experimental results have demonstrated that our semantics-enhanced approach is a correct and effective direction to solve the privacy recommendation problem in online social networks.

In the future, more work can be done to test and perfect the current system before it can be applied to real-life social network environment; furthermore, alternative classification methods other than k-NN can also be studied for semantics enhancement. Specifically, I list the possible future work as follows.

First, the processing speed and the corresponding required resources of the system can be further tested and compared with other alternative methods. While the effectiveness has been demonstrated by the experiments of this paper; the efficiency is another essential measurement of the system usability. The efficiency is normally reflected by the amount of time and resources that are expended in achieving the system throughput objective. In some scenarios that the recommendation task can be executed offline or in the background of the application, the processing speed might not be a critical issue. However, if the recommendations are expected to be used in a real-time interactive manner, the system response time will be a big concern.

Second, the weights of the classifier's features might be automatically adjusted to optimize the classification results. Our experimental results have already reflected the weights of the features have great impact on the classification accuracy. This interesting finding provided us an opportunity to further improve the *SPAC* system. I can further apply related machine learning techniques, such as genetic algorithm and neural networks, to automatically optimize these weight factors. In theory, this further update is expected to bring high prediction accuracies; but on the other hand, the potential high processing and response time might be the side effects. As mentioned previously, the system efficiency should be considered and studied when the system is to be used in practice.

Further, the user interface of the system can be improved to facilitate the usage of the system. User experience and satisfaction is another important measurement of the system usability, although this is not the focus of our current research work. Fang et al. [16] proposed their design of a social networking privacy wizard, which provided a visible interface of their Decision Tree based model. I can take it as a reference and improve the system visualization of our k-NN based model.

Besides these possible improvements of the current Nearest Neighbor based method, I will work on integrating semantics with other machine learning techniques, e.g., Decision Tree and SVM (Support Vector Machine), which are popularly used in recommendation systems.

REFERENCES

[1] A. Acquisti and R. Gross, "Imagined communities: Awareness, information sharing and privacy on the facebook," in the 6th Workshop on Privacy Enhancing Technologies, 2006.

[2] L. Church, J. Anderson, J. Bonneau, and F. Stajano., "Privacy stories: Confidence on privacy behaviors through end user programming," in Symposium on Usable Privacy and Security (SOUPS), 2009.

[3] B. Carminati, E. Ferrari, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham, "A semantic web based framework for social network access control," in the 14th ACM symposium on Access control models and technologies, 2009, pp.177–186.

[4] F. statistics, http://www.facebook.com/press/info.php?statistics.

[5] A. Carreras, E. Rodriguez, and J. Delgado, "Using xacml for access control in social networks," in W3C workshop on access control application scenarios, 2009.

[6] C. Gates, "Access control requirements for web 2.0 security and privacy," in Web 2.0 Security and Privacy Workshop, 2007.

[7] K. Gollu, S. Saroiu, and A.Wolman, "A social networking-based access control scheme for personal content," in The 21st ACM Symposium on Operating Systems Principles (SOSP), 2007.

[8] M. Hart, R. Johnson, and A. Stent, “More content – less control: Access control in the web 2.0,” In Web 2.0 Security and Privacy , 2007.

[9] F. Adu-Oppong, C. Gardiner, A. Kapadia, and P. Tsang, “Socialcircles: Tacking privacy in social networks,” in Symposium on Usable Privacy and Security (SOUPS), 2008.

[10] G. Danezis, “Inferring privacy policies for social networking services,” in Proceedings of the 2nd ACM workshop on Security and artificial intelligence (AISec), 2009.

[11] E. Maximilien, T. Grandison, T. Sun, D. Richardson, S. Guo, and K. Liu, “Privacy-as-aservice:models, algorithms, and results on the facebook platform,” in Web 2.0 Security and Privacy Workshop, 2009.

[12] S. Kruk, S. Grzonkowski and H. Cho, “D-foaf: Distributed identity management with access rights delegation,” in 1st Asian Semantic Web Conference, 2006, pp. 140—-154.

[13] B. Ali, W. Villegas, and M. Maheswaran, “A trust based approach for protecting user data in social networks,” in Proceedings of the 2007 conference of the center for advanced studies on Collaborative research, 2007, pp. 288–293.

[14] B. Carminati, E. Ferrari, and A. Perego, “Enforcing access control in web-based social networks,” ACM Trans. Inf. Syst. Secur., vol. 13, pp. 6:1–6:38, November 2009.

[15] A. Masoumzadeh and J. Joshi, “Osnac: An ontology-based access control model for social networking systems,” in IEEE Second International Conference on Social Computing, 2010, pp. 751 – 759.

[16] L. Fang and K. LeFevre, “Privacy wizards for social networking sites,” in Proc. of the 19th international conference on World wide web, 2010.

[17] S. Safavian and D. Landgrebe, “A survey of decision tree classifier methodology,” vol. 21, no. 3, pp. 660 – 674, 2002.

[18] Z. Fan-Zi and Q. Zheng-Ding, “A survey of classification learning algorithm,” in 7th International Conference on Signal Processing, 2004, pp. 1500 – 1504.

[19] J. Li, S.U. Khan: MobiSN: Semantics-Based Mobile Ad Hoc Social Network Framework. GLOBECOM 2009: 1-6.

[20] Auffarth, B., Lopez, M., Cerquides, J. (2010). Comparison of redundancy and relevance measures for feature selection in tissue classification of CT images. Advances in Data Mining. Applications and Theoretical Aspects. p. 248--262.

[21] Latourrette, M.: Toward an explanatory similarity measure for nearest-neighbor classification. Proceedings of the 11th European Conference on Machine Learning, London, UK, pp. 238‑245. Heidelberg (2000)

[22] G. Salton andM.McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York, NY, USA, 1983.

[23] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, “Analysis of recommendation algorithms for E-commerce,” in Proceedings of the ACM E-Commerce, pp. 158–167, Minneapolis, Minn, USA, 2000.

[24] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: an open architecture for collaborative filtering of netnews,” in Proceedings of the ACM Conference on Computer Supported Cooperative Work, pp. 175–186, New York, NY, USA, 1994.

[25] M. R. McLaughlin and J. L. Herlocker, “A collaborative filtering algorithm and evaluation metric that accurately model the user experience,” in 27th ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 329–336, 2004.

[26] M. Deshpande and G. Karypis, “Item-based top-N recommendation algorithms,” ACM Transactions on Information Systems, vol. 22, no. 1, pp. 143–177, 2004.

[27] G. Karypis, “Evaluation of item-based top-N recommendation algorithms,” in Proceedings of the International Conference on Information and KnowledgeManagement, pp. 247–254, 2001.

[28] M. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical Review, 69(2), 2004.

[29] Gruber, Thomas R. (June 1993). "A translation approach to portable ontology specifications". Knowledge Acquisition 5 (2): 199–220

[30] The igraph software package for complex network research. InterJournal Complex Systems, 2006.

[31] Lall, U. and A. Sharma (1996), A nearest neighbor bootstrap for resampling hydrologic time series, Water Resources Research,, 679-693.

[32] Schaffer, C. (1993) Selecting a classification methods by cross validation. Machine Learning, 13, 135-143.

[33] Peng, J., Heisterkamp, D.R., Dai, H.K.: LDA/SVM driven nearest neighbor classification.In: CVPR 2001, p. 58.

[34] Ding, Chris; Hanchuan Peng. "Minimum Redundancy Feature Selection and Extraction" (PDF Lecture). Ischool at Drexel. 2010.

[35] G. Holmes; A. Donkin and I.H. Witten (1994). "Weka: A machine learning workbench". Proc Second Australia and New Zealand Conference on Intelligent Information Systems.

[36] M. Newman. Modularity and community structure in networks. Proc Natl Acad Sci USA, 103:8577–82, 2006.

[37] J. Chen, O. R. Zaiane, and R. Goebel, “Detecting communities in social networks using max-min modularity,” SDM, 2009.

[38] Mitchell, Tom M., Machine Learning. The Mc-Graw-Hill Companies, Inc., 1997

[39] Cover TM, Hart PE (1967). "Nearest neighbor pattern classification". IEEE Transactions on Information Theory 13 (1): 21–27.

[40] Bremner D, Demaine E, Erickson J, Iacono J, Langerman S, Morin P, Toussaint G (2005). "Output-sensitive algorithms for computing nearest-neighbor decision boundaries". Discrete and Computational Geometry 33 (4): 593–604.