An Approach to Personalized Privacy Policy Recommendations on Online Social Networks

Ammar Abuelgasim and Anne Kayem
Dept. of Computer Science, University of Cape Town, Rondebosch 7701, Cape Town, South Africa

Keywords: Privacy Policies, Recommendation Systems, Social Networks.

Abstract: Most Online Social Networks (OSNs) implement privacy policies that enable users to protect their sensitive information against privacy violations. However, observations indicate that users find these privacy policies cumbersome and difficult to configure. Consequently, various approaches have been proposed to assist users with privacy policy configuration. These approaches are however, limited to either protecting only profile attributes, or only protecting user-generated content. This is problematic, because both profile attributes and user-generated content can contain sensitive information. Therefore, protecting one without the other, can still result in privacy violations. A further drawback of existing approaches is that most require considerable user input which is time consuming and inefficient in terms of privacy policy configuration. In order to address these problems, we propose an automated privacy policy recommender system. The system relies on the expertise of existing OSN users, in addition to the target user's privacy policy history to provide him/her with personalized privacy policy suggestions for profile attributes, as well as user-generated content. Results from our prototype implementation indicate that the proposed recommender system provides accurate privacy policy suggestions, with minimum user input.

1 INTRODUCTION

Online Social Networks (OSNs) have revolutionized the way that users interact and communicate online, by introducing new and innovative ways for self-expression, information sharing, and relationship formation. Popular OSNs nowadays attract a large number of users as attested by Facebook which has 1.49 billion monthly active users, of which 968 million users access Facebook daily (Facebook Inc., 2015). Likewise, Twitter has drawn about 316 million users since its creation in 2006 (Twitter, 2015).

Users of OSNs disclose large volumes of information during their daily interactions. For instance, everyday about 350 million photos are shared on Facebook (Fire et al., 2014), and 500 million tweets are sent on Twitter (Twitter, 2015). The majority of this information is sensitive in nature as Gross et al. (Gross et al., 2013) have pointed out.

The ease with which such sensitive information can be accessed, and by a large number of people, exposes OSNs' users to many privacy and security risks. Examples of such privacy and security risks include identity theft, financial fraud, cyberstalking, cyberbullying, insurance and employment discrimination, embarrassment and losing face with friends (Fire et al., 2014; Gao et al., 2011; Acquisti et al., 2007). As a result, in order to protect users against privacy violations, most OSNs have implemented privacy policies. Privacy policies are essentially a set of rules that enable users to control who can access their disclosed information. For example, OSNs like Google+ and Facebook have fine grained privacy policies that allow users to control access to individual profile attributes such as birthdate, and address, as well as to user-generated content such as posts, photos, and videos. However, it has been shown that numerous users fail to configure their privacy policies either because they are not aware of the existence of these policies, or because these policies are complex and time consuming to understand and configure correctly (Madejski et al., 2012; Liu et al., 2011).

Consequently, various automated approaches have been proposed to assist users with privacy policy configuration. (Fang and LeFevre, 2010; Shehab et al., 2010; Ghazinour et al., 2013b; Alsalibi and Zakaria, 2013; Sinha et al., 2013; Sánchez and Viejo, 2015). These approaches are however, limited to either configuring privacy policies for profile attributes, or configuring privacy policies for user-generated content.

Abuelgasim, A. and Kayem, A.
An Approach to Personalized Privacy Policy Recommendations on Online Social Networks.
DOI: 10.5220/0005689701260137
In *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016)*, pages 126-137
ISBN: 978-989-758-167-0

This is problematic, because both profile attributes and user-generated content can contain sensitive information. For instance, users have been documented to have put their real cellphone numbers on their profiles, and also disclosing their physical whereabouts in status updates (Gross and Acquisti, 2005; Gundecha et al., 2011; Toch et al., 2010; Ghazinour et al., 2013a). Therefore, protecting profile attributes without user-generated content (and vice-versa), is not a good privacy protection solution for OSNs. Furthermore, most of the proposed privacy policy configuration approaches require substantial user input, which is a time consuming process.

In order to address the aforementioned problems, in this paper we propose an automated privacy policy recommender system that enables users to protect both profile attributes and generated content, with minimal user input. In summary, the recommender system consists of two main components. The first component, the Profile Attributes Protector (PAP), utilizes privacy policies that existing experienced OSN users have specified for their profile attributes, to suggest to new target users[1], how to configure privacy policies for their profile attributes. The second component, which is the User Content Protector (UCP), utilizes the target user's privacy policy history, to suggest suitable privacy policies for content that the target user might generate and share in the future. We designed the recommender system as 'server-side' solution, with the understanding that this would be maintained by the OSN provider. The advantage of this solution is twofold, first we protect both profile attributes and user-generated content which is important in preventing privacy leaks and second, we alleviate the issue of manual privacy policy configurations by providing the users with automated personalized privacy policy suggestions.

The rest of the paper is structured as follows, in Section 2 we provide an overview of the related work on automated privacy policy specification in OSNs. We describe our proposed privacy policy recommender system in Section 3 and discuss experimental results on our prototype recommender system implementation in Section 4. Finally, we conclude in Section 5 with a summary and offer some ideas for future work.

2 RELATED WORK

Several studies have proposed helping OSNs' users with privacy policy configuration, by automating the process of privacy policy configuration. Fang and LeFevre (Fang and LeFevre, 2010) pioneered this area with a a mechanism for configuring privacy policies. Their privacy policy configuration mechanism follows an active learning approach, whereby for each profile attribute, the user is prompted to label a subset of his/her friends by stating whether or not the friends are allowed to access that profile attribute. Next, the wizard uses this subset of friends to train a classifier that predicts which of the user's remaining friends are allowed (or not allowed) to access that particular profile attribute. While this approach facilitates setting fine-grained privacy policies, users are required to provide considerable input to enable the system run efficiently. In fact, for every profile attribute (and one can have up to 27 attributes) the user is required to manually label a group of friends, in order to train the attribute's classifier. A further caveat of this solution is that it does not handle privacy policies for user-generated content, since the classifiers are trained to predict privacy policies for profile attributes only.

Shehab et al. (Shehab et al., 2010) proposed a solution similar to the one of (Fang and LeFevre, 2010), in which the user is required to label a selected subset of his/her friends as trusted or not trusted to access a particular profile object. This subset is then used to train a classifier that predicts which of the remaining friends are trusted (or not trusted) to access that particular object. In contrast to (Fang and LeFevre, 2010)'s scheme, the Shehab et al. scheme introduces an interesting concept, where the resulting classifier is merged with other neighboring users' classifiers to enhance the classifier's performance. However, the Shehab et al. scheme also requires substantial user input, because the user has to manually label a group of his/her friends for every profile object.

Ghazinour et al. (Ghazinour et al., 2013a; Ghazinour et al., 2013b) introduced a tool for recommending privacy policies called 'YourPrivacyProtector'. YourPrivacyProtector uses K-Nearest Neighbours algorithm to find the closest 3 profiles to the user's, and then it uses the privacy policies of the neighbour users to suggest to the user whether to disclose a particular profile attribute or not. This tool however only provides coarse-grained privacy policy suggestions. Furthermore, this approach cannot support privacy policies for user-generated content. Unlike profile attributes, users might generate different types of content. For instance, Bob's status updates are usually very different form Alice's status updates. Thus, it is not feasible to rely on other users content's policies for providing suggestions.

On the other hand, Alsalibi and Zakaria (Alsal-

[1] In the recommender systems' literature, the user that receives the recommendations (i.e. suggestions) is usually referred to as the 'target user'.

ibi and Zakaria, 2013) criticise the approach of (Fang and LeFevre, 2010), arguing that it incapable of providing recommendations for user's who do not have friends yet. Therefore, Alsalibi and Zakaria proposed a collaborative filtering privacy recommender system. In order to recommend privacy policies to a particular target user; this system first identifies a group of similar users (to the target), and then it uses the most frequently used privacy policies within this group, to make privacy policy recommendation to the target user. However, this system also fails to handle privacy policies for user-generated content for the same reason as that of Ghazinour et al.

Sinha et al. (Sinha et al., 2013) were the first to identify that non of the existing approaches, caters for user-generated content. Therefore, Sinha et al. propose an automated tool to help users configure privacy policies for text-based content. Sinha et al. used the Maximum Entropy classification algorithm (MaxEnt) to predict and recommend privacy policies for text-based content. However, despite being able to protect users' text-based content, this tool does not provide any protection for the profile attributes.

Along the same lines, Sánchez and Viejo (Sánchez and Viejo, 2015) proposed an automated mechanism to inform OSNs' users about the privacy risks inherent to their unstructured text-based content, to enable users to make more informed privacy policy choices. The proposed mechanism adopts an information theoretic approach, and it works by comparing the text-based content's 'sensitivity' against the content owner's 'privacy requirements' for all types of users in the OSN. However, this approach only warns users about potential privacy conflicts within their generated content, it does not suggest privacy policies directly to the users.

We note that existing schemes offer automated privacy policy configuration for either profile attributes or user generated content but not both. This is problematic, because as mentioned before, protecting one without the other can lead to privacy violations. Furthermore, requiring users to provide considerable content to support privacy policy configuration offers the advantage of high accuracy, but has the downside of being cumbersome to use for the average user. In the following section we discuss the details of our proposed privacy policy recommender system.

3 PRIVACY POLICY RECOMMENDER SYSTEM

In order to assist users with privacy policy configuration we propose a 'server-side' privacy policy recommender system that enables users to protect profile attributes and generated content, without having to deal with complex privacy policy configuration procedures. Our recommender system consists of two independent components, that work in parallel to protect the users' sensitive information. The first component, which we term the **Profile Attributes Protector** (PAP), relies on the privacy policies that existing users have specified for their profile attributes to suggest to presumably naïve target users, how to configure their profile attribute privacy policies. Specifically, the PAP extracts these policies from the profiles of existing experienced users, and uses this data to build several machine learning classifiers, which in-turn are used to suggest to the target users suitable privacy policies for profile attributes. The second component, which we term the **User Content Protector** (UCP). The UCP learns from the target user's privacy policy history and on the basis of this knowledge, suggests suitable privacy policies for the target user's future content. In other words, the UCP relies on the targets user's past privacy policy configurations for generated content to train a classifier, that is then used to suggest suitable privacy policies for the target user's future content.

An overview of the recommender is depicted in Figure 1 below, as shown, in the PAP the training data is extracted from the profiles of experienced users and used to train several decision tree classifiers. The classifiers then output privacy policy suggestions for the target user. The UCP works in a similar fashion, first, the training data is extracted from the target user's privacy policy history and used to train a naïve bayes classifier, which outputs privacy policy suggestions for user generated content.

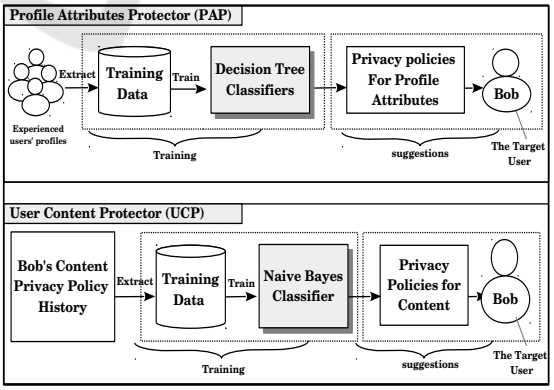


Figure 1: An overview of the proposed privacy policy recommender system.

In the following subsections we begin by first characterizing the theoretical structure of the OSN model, and then describe how both the PAP and UCP work to automate privacy policy configuration.

3.1 OSN Notations

We model an OSN as an undirected graph $G = (V, E)$, where V is the set of users represented by nodes in the graph, and E is the set of social relationships between the users, represented by edges in the graph. Every user $u \in V$ creates and maintains a profile P_u, which is a personal page (or space) that serves as a digital representation of the user. The user's profile consists of a fixed set of attributes $A_u = \{a_{u1}, a_{u2}, \ldots, a_{un}\}$ and a collection of pieces of user-generated content $C_u = \{c_{u1}, c_{u2}, \ldots, c_{uk}\}$. Profile attributes A_u describe the profile owner (i.e. the user). These profile attributes include demographic information such as: age, gender, political views/interests. User-generated content C_u is the materials created by users during OSN's interactions. Examples of user-generated content include: status updates, notes, posts, photos, and so on.

We assume that users can specify privacy policies, to control access to their profile attributes and content. A privacy policy is a user-defined rule in the form $< item, l >$, where $item$ can be any profile attribute $a_j \in A_u$ or any piece of content $c_j \in C_u$, while $l \subseteq V$ represents the audience, that is, the set of users allowed to access the $item$. Users are provided with a finite set of audiences $L = \{l_1, l_2, \ldots, l_k\}$ to choose from. An example of a privacy policy is $< Photo_{\#1}, Family >$, which means only Family members can access $Photo_{\#1}$.

3.2 Recommendation System

As mentioned earlier, the recommendation system is composed of two principal components namely, the Profile Attributes Protector (PAP) and the User Content Protector (UCP). The Profile Attributes Protector (PAP) is the component responsible for suggesting privacy polices for all the profile attributes in the target user's profile. While the User Content Protector (UCP) handles enforcing privacy on the target user's generated content.

3.2.1 Profile Attributes Protector (PAP)

In the PAP, we follow a demographic-based approach to recommender systems, where the motivating assumption is that: demographically similar people have similar privacy policies for their profile attributes. However, in order to get a deeper understanding of how the PAP works, we must first understand how it suggests privacy policies for individual profile attributes. The process by which PAP suggests privacy policies for an individual profile attribute, say $a_i \in A$ consists of three main phases.

The first phase is the data collection phase. In this phase, we collect our training data from the profiles of existing experienced users. Specifically, since the recommender is designed to be a server-side solution, we expect that it will have direct access to users' profiles' data. Therefore, from existing user's profile, we extract the user's demographic information, and the privacy policy that the user has set for attribute a_i. We then store the collected information in a dataset D, in which every record is of the form $(\vec{F}, l_j)$, where $\vec{F} = \langle f_1, f_2, \ldots, f_m \rangle$ is the user's demographic information, representing the features, and l_j is attribute a_i's privacy policy, and it represents the class label.

The second phase is the classifier training phase. we train a classifier that predicts the privacy policies of the a_i attribute. For this task we opted to use the Decision Tree algorithm, because it is capable of handling classification of categorical, noisy, and incomplete data (Mitchell, 1997), which are the characteristics of our training data D.

In order to build this classifier, we apply a decision tree learning algorithm on our dataset D, which builds the classifier as follows. First, the algorithm looks at the training dataset D and finds the feature f_* that tells us the most about users' choice of privacy policies. It does so by using a statistical measure called Information Gain (Gain for short), which is given by the following equation.

$$\boxed{Gain(D, f_*) = H(D) - H(D|f_*)}$$
where $H(x)$ is the entropy of X
and is given by $H(x) = -p(x)\log(p(x))$ (1)

The algorithm then uses the feature with the highest gain (i.e. f_*) to create the root node. Afterwards the training dataset D is partitioned into several partitions, such that, each partition contains records that have the same value for f_*. Next, for each partition p, the algorithm looks for the feature f_{*p} that has the highest gain in p and uses it (i.e. f_{*p}) to create a child node. The partition p itself is then re-partitioned and the process is repeated, until every partition has almost the same privacy policy, or the algorithm runs out of training records.

The final classifier will be represented as a decision tree, where each tree node represents a test for a specific feature $f_i \in \vec{F}$ (i.e. demographical trait), and each edge branching from the node corresponds to one of possible values of that feature. The leaves of the tree correspond to privacy policies (i.e. class labels). This decision tree predicts the privacy policies of the a_i attribute, by testing the target user's features at the root node, and moving down the tree to a child node that corresponds to the value of the tested fea-

ture. The target user's features are then retested at the child node, and moved down the tree progressively. The process is repeated until a leaf node is reached. The privacy policy associated with this leaf is the predicted privacy policy for the a_i attribute.

The third and final phase is the suggestion phase. This phase is triggered by the arrival of a target user. When a target user registers on the OSN, we extract demographical information form his/her newly created profile, and pass it to attribute a_i's classifier, which in-turn predicts a privacy policy for the a_i attribute. This privacy policy is then simply suggested to the target user. In Figure 2 below, we depict the process of suggesting privacy policies for individual profile attributes, that we have described above.

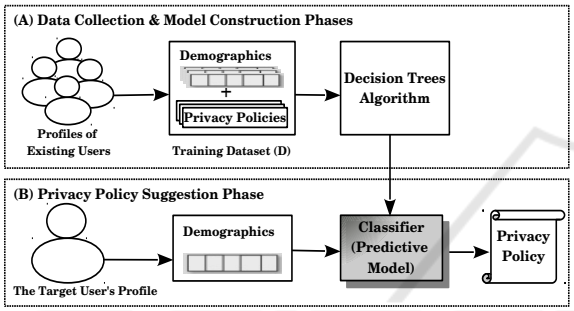


Figure 2: The process of suggesting privacy policies for an individual profile attribute.

So far, we have described how privacy policies are suggested for one profile attribute. In order to suggest privacy policies for all the attributes in the target user's profile, we repeat the previously described process for each one of the profile attributes. Specifically, we extract demographic information and privacy policies from the profiles of existing users. We then use this data to build a series of training datasets $\{D_1, D_2, \ldots, D_{|A|}\}$, one for each profile attribute $a_i \in A$. Next, we apply the decision tree learning algorithm on these training datasets, to train a series of classifiers $CL = \{cl_1, cl_2, \ldots, cl_{|A|}\}$. Each classifier $cl_i \in CL$ predicts the privacy policies that the target users would set for an attribute $a_i \in A$, $\forall i \in \{1, 2, \ldots, |A|\}$. Finally, each time a target user registers at the OSN. We pass his/her demographics to every classifier $cl_i \in CL$, to predict the privacy policies for each attribute in the target user's profile. These predicted policies are then suggested to the target user.

3.2.2 User Content Protector (UCP)

The UCP is activated after the target user joins the OSN, and starts generating different types of content as well as specifying privacy policies for some of the content he/she has generated.

In the UCP, we follow a content-based approach to recommender systems, were the motivating assumption is that: similar pieces of user-generated content have similar privacy policies. Accordingly, the process by which the UCP suggests privacy policies for content generated by a particular target user, say $u_t \in V$ consists of the following four phases.

The first phase is the data collection phase, in this phase we collect the privacy policy history of the target user's content, and arrange it into a dataset Ω, where every record is in the form (c_i, l_j), where c_i is a piece of content generated by the target user u_t, and l_j is the privacy policy that u_t has specified for c_i.

The second phase is the preprocessing phase (i.e. feature extraction phase), in this phase we transform every piece of content $c_i \in \Omega$ to a vector of features that characterize c_i, thereby transforming Ω from a collection of 'labelled' user-generated content, to a dataset of labelled feature vectors $\hat{\Omega}$, in which every record is in the form $(\vec{c_i}, l_j)$, where, $\vec{c_i} = \langle f_{i1}, f_{i2}, \ldots, f_{im} \rangle$ is c_i's feature vector.

The third phase is classifier training phase. Here, we build the classifier that predicts privacy policies for the target user's content. For this task we opted to use the Naïve Bayes algorithm, because it is quick to test and train, which is a necessity when it comes to predicting privacy policies for user-generated content.

Specifically, we apply the Naïve Bayes algorithm on our preprocessed dataset $\hat{\Omega}$. Naïve Bayes follows a probabilistic model, whereby in order to predict the privacy policy for a piece of user-generated content c_i. Naïve Bayes selects the privacy policy $l_* \in L$ that maximizes the probability $P(l_j|f_1, f_2, \ldots, f_m) \quad \forall l_j \in L$, which is the probability that the privacy policy l_j will be observed given that the features $\{f_1, f_2, \ldots, f_m\}$ that characterize content c_i are observed. l_* is determined as follows. First, according to Bayes theorem in equation (2)

$$\boxed{P(B|A) = \frac{P(A|B)P(B)}{P(A)}} \tag{2}$$

The probability that the privacy policy l_j will be observed given that c_i's features (i.e. $\{f_1, f_2, \ldots, f_m\}$) are observed, is calculated as follows.

$$P(l_j|f_1, f_2, \ldots, f_m) = \frac{P(f_1, f_2, \ldots, f_m|l_j)P(l_j)}{P(f_1, f_2, \ldots, f_m)} \tag{3}$$

By assuming that the c_i's features are conditionally independent, equation (3) can be rewritten as

$$P(l_j|f_1, f_2, \ldots, f_m) = \frac{\left(\prod_{i=1}^{m} P(f_i|l_j)\right)P(l_j)}{P(f_1, f_2, \ldots, f_m)} \tag{4}$$

Now to find l_* we select the privacy policy that

maximizes $P(l_j|f_1, f_2, \ldots, f_m)$, which is.

$$l_* = \arg\max_{l_j \in L} \left[\frac{\left(\prod_{i=1}^{m} P(f_i|l_j) \right) P(l_j)}{P(f_1, f_2, \ldots, f_m)} \right] \quad (5)$$

Since $P(f_1, f_2, \ldots, f_m)$ is constant for all $l_j \in L$, then

$$\boxed{l_* = \arg\max_{l_j \in L} \left[\left(\prod_{i=1}^{m} P(f_i|l_j) \right) P(l_j) \right]} \quad (6)$$

Where the probability $P(l_j)$ and each of the conditional probabilities $P(f_i|l_j)$ are estimated from $\hat{\Omega}$.

The fourth and the final phase is the suggestion phase, this phase is triggered when the target user u_t generates any new piece of content c_{new}. We first preprocess c_{new} to transform it into a vector of features. We then, pass this feature vector to u_t's classifier, which in-turn predicts c_{new}'s privacy policy. Then we simply suggest the predicted policy to u_t. In Figure 3 below, we depict the process of suggesting privacy policies for user-generated content.

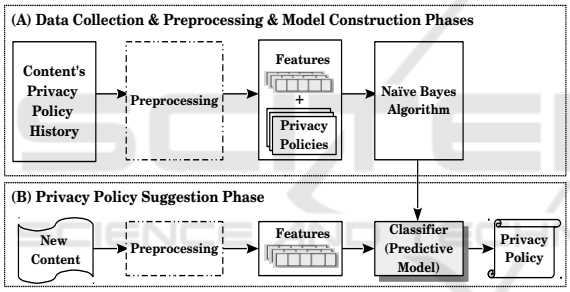


Figure 3: The process of suggesting privacy policies for user-generated content.

In the next section, we describe the experiments and implementation details of a basic proof-of-concept prototype of our recommender system.

4 EXPERIMENTS AND RESULTS

In order to implement our proposed privacy policy recommender system prototype, we carried-out several experiments which were primarily focused on building the classifiers that predict privacy policies for our target users. We generated the training data, using the Facebook Graph API, and NetLogo which is a widely used, Agent-Based Modeling and Simulation (ABMS) environment. We also used Weka, which is a popular machine learning/data mining package to train and validate our classifiers. Finally, all of our experiments were conducted on an Intel core i7-4790 3.60 GHz workstation, with 8GB of RAM.

4.1 Implementation and Experiments with the Profile Attribute Protector

In order to implement the Profile Attributes Protector (PAP); it's essential that we obtain rich OSNs datasets, to build the most important elements of this component, which are the classifiers that predict privacy policies of profile attributes. However, acquiring these datasets is difficult, due to the sensitive nature of OSNs' data, and the ethical issues associated with it (Zimmer, 2010). As an alternative, we opted to synthetically generate these datasets through simulating an OSN, the details of which we describe below.

4.1.1 The OSN Simulation Model

In order to simulate an OSN, we followed an agent-based approach to modelling and simulation, because it is suitable for simulating complex systems such as OSNs (Macal and North, 2011). We implemented our OSN simulation model using NetLogo.

Our OSN simulation model consists of a set of agents, representing OSN's users. For simplicity, we assigned each agent only eight profile attributes. In addition, each agent is characterized by three parameters namely, a friendship threshold $\lambda \in [0,1]$, which indicates how friendly an agent is; a maximum nodal degree δ, which represents the maximum number of friendships an agent can maintain (Ang and Zaphiris, 2009); and lastly, a privacy concern $\theta \in [0,1]$, which quantifies the agents level of privacy concern (Guo and Chen, 2012). We assume that λ, δ, and θ are normally distributed.

We rely on two concepts namely, homophily and triadic closures to simulate how a user's friendship graph is formed. Homophily is the love of the same (Bakshy et al., 2012), while triadic closures describe the tendency to make friendships with friends of friends (Klimek and Thurner, 2013). The idea of using these concepts is to ensure that our OSN follows the same pattern of friendship formations as is the case in typical real-life OSNs. The friendship graph is formed as follows, each agent u checks whether its current number of friends has reached its internal maximum degree δ_u. If this is not the case, then u selects another random agent v, and calculates the percentage of their identical profile attributes $S_{(u,v)} \in [0,1]$, and percentage of their mutual friends $mf_{(u,v)} \in [0,1]$. Next, if the friendship score, given by a linear combination of $S_{(u,v)}$ and $mf_{(u,v)}$, is greater than u's internal friendship threshold λ_u. Then a friendship is formed between u and v. This 'socialization' process is repeated until all agents reach their maximum degrees of friendship.

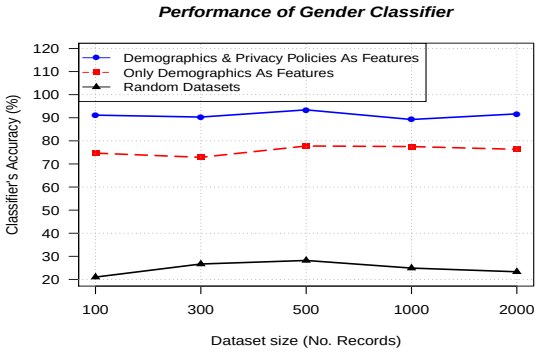
The privacy policy configuration process is inspired by the work of Guo and Chen (Guo and Chen, 2012), in which profile attributes are classified by sensitivity (that is privacy weights), and the privacy policies that are assigned for an attribute, are influenced by the user's level of privacy concern, and the attribute's privacy weight. Following this heuristic, we assigned each profile attribute $a_j \in A$, an arbitrary privacy weight $w_{a_j} \in [0,1]$, that indicates a_j's sensitivity. In order to set a privacy policy for an attribute, say a_i we modelled each agent u to first calculate a_i's privacy score, which is a linear combination of u's privacy concern θ_u, and a_i's privacy weight w_{a_i}. We then map the numerical privacy score to one of four possible audiences, namely: $L = \{$only me, friends, friends of friends, public$\}$.

Next, in order to generate our training data, we used the simulation model to run several simulations of OSNs of different sizes. By the end of each of these simulations, we collected every agent's demographic information and privacy policies, and stored it in a dataset. We also generated several control datasets, where the privacy policies are assigned randomly without going through the simulation model.

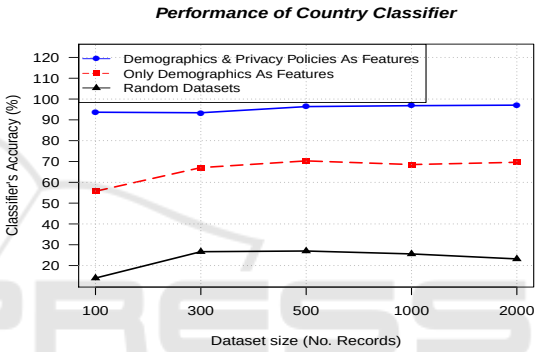
4.1.2 Training and Evaluating the Profile Attributes' Classifiers

For training the profile attributes' classifiers, we used the J48 algorithm, which is Weka's implementation of the C4.5 decision tree learning algorithm. Specifically, we applied the J48 algorithm on each of our datasets (both simulated and random) to train a classifier for each of our eight profile attributes. Next, in order to evaluate the performance of these classifiers, we performed a 10-fold cross-validation test, and recorded the accuracy of each model (that is the percentage of the records for which the classifier was successful). Cross-validation results show that the classifiers trained on simulated datasets achieved hight accuracy ranging from 60% to 80%. While those trained on random datasets were worse, as they achieved an accuracy ranging from 20% to 30%.

In an effort to further enhance the classifiers' accuracy, we extended the features used in training these classifiers. Particularly, instead of using only demographics as features, this time we used a combination of demographics and the privacy policies of other profile attributes. We then, retrained our classifiers on the new datasets, and measured their accuracy using the aforementioned methodologies. The results showed an average of 15% to 25% improvement in the accuracy of classifiers trained on simulated datasets. On the other hand, no significant improvement was noticed on the accuracy of classifiers trained on random datasets, as illustrated in Figure 4, which shows the accuracy of two selected profile attribute classifiers, plotted against the size of their training datasets.



(a)

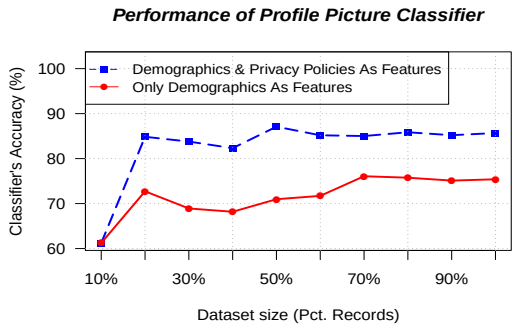


(b)

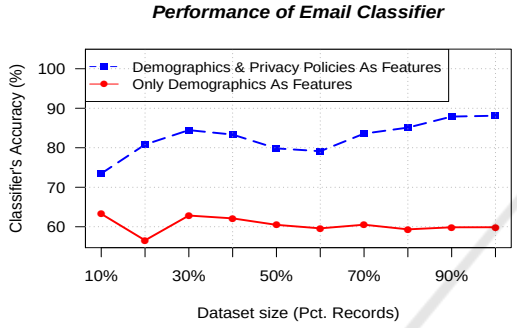
Figure 4: The performance of two profile attribute's classifiers when trained on both random and simulated datasets. The accuracy of each classifier is plotted against the size of the dataset used for training it.

At later stages of this work, we obtained a real dataset used in a similar study. This dataset however, is structurally different than our simulated datasets, as it has only three demographical features, compared to eight features in the simulated datasets. Nonetheless, this dataset can give an insight on how our system will behave on real OSN data.

Therefore, we applied the J48 algorithm on the real data, and trained a total of eleven, one for each profile attribute in the real dataset. Next, we performed a 10-fold cross-validation test, to evaluate the accuracy of these classifiers. The results showed that, the classifiers achieved a relatively high accuracy, ranging between 60% to 70%, but is a bit lower than the classifiers trained on the simulated datasets. Interestingly however, similar to what we observed with the simulated datasets; the classifiers' accuracy experienced an average of 17% improvement when we used a combination of demographics and privacy policies as features, as illustrated in Figure 5 below.
Next, in order to study and measure the influence of

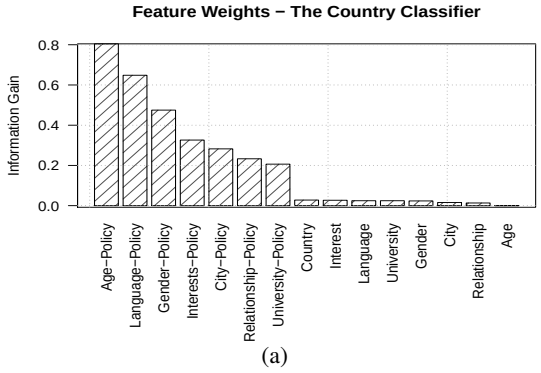


(a)

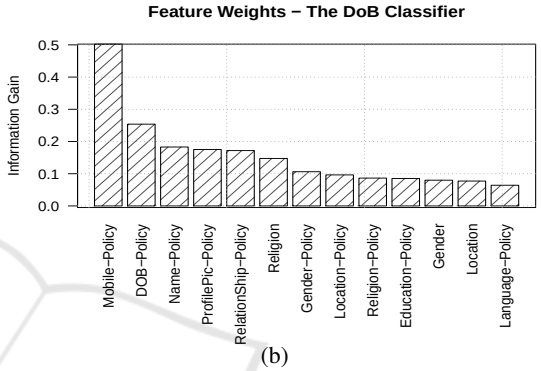


(b)

Figure 5: The performance of two profile attribute's classifiers when trained on the 'real' dataset. The accuracy of each classifier is plotted against the percentage of the dataset records used for training it.



(a)



(b)

Figure 6: Shows the features of the *Country* and *Email* attribute classifiers ranked by their information gain.

that each feature have on the classifiers accuracy; we calculated the information Gain of each feature. Information Gain is a statistical measure that quantifies the reduction of entropy when the value of that particular feature is known. In other words, it indicates how much that feature tells us about the target user's choice of privacy polices. We then ranked the features of each classifier based on the calculated information Gain. Initial results showed that demographic features generally have lower information Gain that privacy policy features. As illustrated in Figure 6, which shows the information Gain of the features of the *Country* and *Email* attribute classifiers.

4.2 Implementation and Experiments with the User Content Protector

In order to implement the User Content Protector; we focused on building the classifier that predicts privacy policies for the user-generated content.

Even though users can generate/share different types of content; however, for the purpose of our experiments we decided to work with text-based content, because it is availability. However, it is possible to do so with other content types like photos for instance (Squicciarini et al., 2011). Next, we collect the necessary data for building the classifier.

4.2.1 Data Collection

In order to collect that data necessary for building the UCP classifier; we used Facebook's graph Application Programming Interface (API) to download posts (i.e. user-generated content) from one of the author's Facebook accounts. Then we manually labelled each post with a suitable privacy policy. In total we downloaded 596 posts, 293 of them were written in English, and 290 were written in Arabic, while 13 posts were written in both languages. Next, we discuss how we transform our raw data into feature vectors.

4.2.2 Preprocessing the Data

In order to transform our recently harvested data Ω, from a 'corpus' of text, to a dataset of labelled feature vectors $\hat{\Omega}$, we used 'text classification' preprocessing methods. Specifically, we started the data-preprocessing with normalising each content (i.e. post) $c_i \in \Omega$ by lower-casing letters and removing diacritical marks. Then, we stemmed the data using

the *lovins stemmer* provided by Weka (Arabic content was left un-stemmed). Next, we used word tokenisation to break-down our text corpus Ω into small units called terms or words (i.e. blocks of consecutive characters). Then, out of these terms, we only kept 100 terms (per class) to form a *term vocabulary V* (which is Weka's default). Lastly, we vectorized Ω such that, each content $c_i \in \Omega$ is represented as a $|v|$-dimensional vector $< c_{i1}, c_{i2}, \ldots, c_{iv} >$, where c_{ij} is a binary value representing the existence of term j in c_i. Next, we discuss how we build the UCP classifier.

4.2.3 Building the Classifier

Before we start building the UCP content classifier, we first divided our preprocessed dataset $\hat{\Omega}$ into: a training & validation dataset $\hat{\Omega}_{TV}$ (contains 70% of the records in $\hat{\Omega}$); and a *test* dataset $\hat{\Omega}_{Te}$ (contains the remaining 30%). As advised by (Sebastiani, 2002).

Next, in order to build the UCP content classifier, we experimented with applying several algorithms on our training & validation dataset $\hat{\Omega}_{TV}$ namely: naïve bayes; multinomial naïve bayes; complement naïve bayes, in addition to Weka's SMO algorithm.This resulted in the construction of multiple classifiers. Since we only need one classifier, we performed a 10-fold cross-validation test with the intention of selecting the best performing classifier. Results showed that the classifiers performed poorly, were the best performing classifier is the one trained using complement naïve bayes, which achieved 50.11% accuracy.

In an effort to enhance the classifiers' accuracy, we returned to the data-preprocessing phase. This time, we increased the size of term vocabulary V to a maximum of 2000 terms, and removed stop-words such as *a*, *if*, and others. We also changed the vectorization such that, each content c_i is represented as $|v|$-dimensional vector $< c_{i1}, c_{i2}, \ldots, c_{iv} >$, where c_{ij} is a the $tf\text{-}idf$ [2] value. Furthermore, we also created another training set (using the same aforementioned preprocessing steps) but with an additional dimensionality reduction step, in which we removed all terms with *information gain* less than zero.

After re-training the classifiers on the new datasets, and measuring their accuracy, we noticed an average of 11.5% performance improvement, where the accuracy of the best performing classifier (i.e. the one trained using complement naïve bayes) is now reaching up-to 64%. As evident in Figure 7 below, which shows the content classifiers' accuracy after changing the preprocessing phase.

In order to further enhance the classifiers accuracy, we went back again to the preprocessing phase.

[2]Term frequency-inverse document frequency $tf\text{-}idf$.

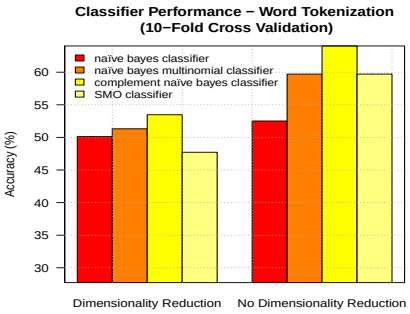


Figure 7: The accuracy of the UCP content's classifier when the training data (i.e. $\hat{\Omega}_{TV}$) is preprocessed using word tokenisation and $tf\text{-}idf$ vectorisation.

This time, we used the same preprocessing steps that we used previously, except for tokenisation, we used 3-gram tokenisation where terms can be sequence of words instead of individual words. The cross-validation results showed a small accuracy improvement, while still the highest accuracy is achieved by the classifier trained using complement naïve bayes, which was around 65.2%. As evident in Figure 8, which shows the content classifiers' accuracy, after last changes in the preprocessing phase.

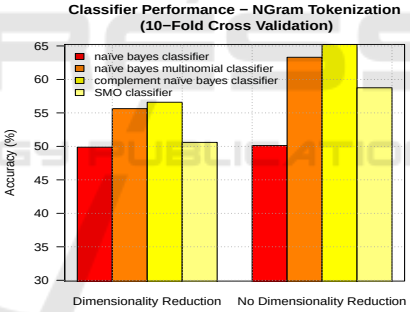


Figure 8: The accuracy of the UCP content's classifier when the training data (i.e. $\hat{\Omega}_{TV}$) is preprocessed using 3-gram tokenisation and $tf\text{-}idf$ vectorisation.

Since, the experimental results showed that the highest accuracy was achieved by the classifier trained using complement naïve bayes; on a dataset preprocessed using a combination of normalisation, stemming, 3-gram tokenisation, stop-words removal, and $tf\text{-}idf$ vectorisation. Therefore, we selected this classifier to be the UCPs main content classifier. We then validated this classifier on our 'untouched' testing data set $\hat{\Omega}_{Te}$. The results showed that the classifier's performance was relatively stable, achieving an accuracy around 64.2%.

Next, in order to study the affect of the size of the training corpus on the accuracy of the content classifier; we re-trained the 'selected' UCP content classifier several times. Such that, in each time we in-

creasing percentage of records from $\hat{\Omega}_{TV}$ to train the classifier. The results showed that the accuracy of the UCP content classifier starts to converge when 70% of the records in $\hat{\Omega}_{TV}$ are used to train the classifier. As evident in Figure 9.

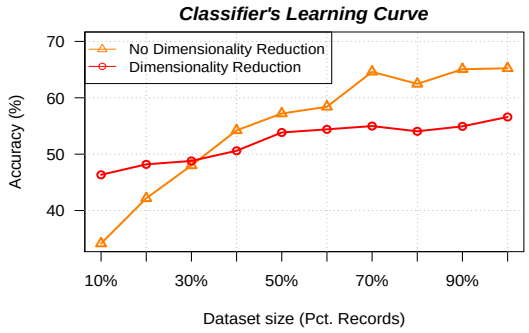


Figure 9: The learning curve of the UCP content classifier (i.e. the classifier accuracy plotted against the percentage of records from $\hat{\Omega}_{TV}$ that were used to train the classifier).

4.3 Privacy Analysis - A Discussion

Privacy policies enable users to protect their privacy by allowing users to specify boundaries, within which their sensitive information resides. However, since these boundaries might defer from one user to another; OSN providers tend to provide open (i.e. very permissive) default privacy policies, thus leaving the task of configuring (i.e. fine tuning) these policies for the users. As a result, new users by default have no, or at best very little privacy, unless they configure their privacy policies to suit their privacy needs.

However, for 'privacy-aware' users who invest the time and effort to configure their privacy policies, we assume that their policies are correct, and more privacy-preserving the OSN's default policies. By relying on the privacy policies of the existing OSN's users, our solution insures that our targets users' profile attribute privacy policies, are as privacy-preserving as the policies of existing 'privacy-aware' users, which is better the OSN's default, and by relying on the target user's privacy history, our solution insures that target user's content's privacy policies are as privacy-preserving as the user's own privacy history, which is also better the OSN's default.

In our experiments with our PAP implementation we used both simulated and real datasets. Our results indicate that, when we use simulated datasets, the PAP was able to suggest privacy policies for profile attributes with a high accuracy, ranging between 60% - 80%. Furthermore, this accuracy can be improved by up to 25% if we used a combination of demographics and privacy policies as features. When we used the real dataset however, the accuracy was slightly lower, ranging between 60% - 70%, but it is expected given the structural differences between the simulated and the real dataset. Interestingly however, we also noticed average of 17% improvement in the accuracy when we used a combination of demographics and privacy policies as features. This is indicative of the fact that the PAP's accuracy rate can be significantly increased, by requiring the user to provide some privacy policies for a few attributes.

In experiments with UCP implementation, we used a dataset that we collected from one of the authors Facebook accounts. After experimenting with several preprocessing techniques, and classification algorithms our results showed that UCP was able to predict suitable privacy policies for user-generated content with an accuracy reaching about 65.2%.

5 CONCLUSIONS

In order assist users with privacy policy configuration, in this paper we proposed an automated privacy policy recommender system, that caters for profile attributes, as well as user-generated content. The system consists of two components namely, the Profile Attribute Protector (PAP), which relies on knowledge and expertise of existing OSN's users, to suggest privacy policies for profile attributes, thus minimizing the input required from the target user; and the User Content Protector (UCP), which utilizes the target user's own privacy policy history, to suggest suitable privacy policies for his/her future generated content.

We experimented with implementing both of the recommender system's components, focusing mainly on the classifiers that predict privacy policies for users. The experimental results were promising, they showed that the recommender system was able to predict (thus suggest) suitable privacy policies with high accuracy, for both profile attributes and user-generated content.

For future work we plan to work on improving the accuracy of the system's classifiers, and conduct more extensive experiments including implementing the PAP using real datasets, and studying how the UCP will 'generalise' to other content-types such as photos for instance. Additionally, the current 'batch-learning' approach that we followed for building the UCP content classifier, does not handle the scenario where the target user does not have any generated content. Therefore, for future work we plan to follow an 'online learning' approach, where the content classifier is built incrementally as new pieces of content are generated by the target user. We also plan to work on narrowing down the number of pieces of

user-generated contents required for providing accurate privacy policy recommendations.

ACKNOWLEDGEMENTS

Funding for this research was provided by the National Research Foundation (NRF) of South Africa and the Norwegian National Research Council.

REFERENCES

Acquisti, A., Carrara, E., Stutzman, F., Callas, J., Schimmer, K., Nadjm, M., Gorge, M., Ellison, N., King, P., Gross, R., and Golder, S. (2007). Security Issues and Recommendations for Online Social Networks. Technical Report 1, European Network and Information Security Agency.

Alsalibi, B. and Zakaria, N. (2013). CFPRS : Collaborative Filtering Privacy Recommender System for Online Social Networks. *Journal of Engineering Research and Applications*, 3(5):1850–1858.

Ang, C. and Zaphiris, P. (2009). Simulating social networks of online communities: Simulation as a method for sociability design. In *Human-Computer Interaction*, volume 5727 of *Lecture Notes in Computer Science*, pages 443–456. Springer Berlin Heidelberg.

Bakshy, E., Eckles, D., Yan, R., and Rosenn, I. (2012). Social influence in social advertising: Evidence from field experiments. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, EC '12, pages 146–161, New York, NY, USA. ACM.

Facebook Inc. (09-2015). Statistics. https://newsroom.fb.com/company-info/.

Fang, L. and LeFevre, K. (2010). Privacy wizards for social networking sites. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 351–360, New York, NY, USA. ACM.

Fire, M., Goldschmidt, R., and Elovici, Y. (2014). Online social networks: Threats and solutions. *Communications Surveys Tutorials, IEEE*, 16(4):2019–2036.

Gao, H., Hu, J., Huang, T., Wang, J., and Chen, Y. (2011). Security issues in online social networks. *Internet Computing, IEEE*, 15(4):56–63.

Ghazinour, K., Matwin, S., and Sokolova, M. (2013a). Monitoring and recommending privacy settings in social networks. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, EDBT '13, pages 164–168, New York, NY, USA. ACM.

Ghazinour, K., Matwin, S., and Sokolova, M. (2013b). YourPrivacyProtector: A Recommender System for Privacy Settings in Social Networks. *International Journal of Security*, 2(4):11–25.

Gross, R. and Acquisti, A. (2005). Information revelation and privacy in online social networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, WPES '05, pages 71–80, New York, NY, USA. ACM.

Gross, R., Stutzman, F., and Acquisti, A. (2013). Silent Listeners: The Evolution of Privacy and Disclosure on Facebook. *Journal of Privacy and Confidentiality*, 4(2):7–41.

Gundecha, P., Barbier, G., and Liu, H. (2011). Exploiting vulnerability to secure user privacy on a social networking site. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 511–519, New York, NY, USA. ACM.

Guo, S. and Chen, K. (2012). Mining privacy settings to find optimal privacy-utility tradeoffs for social network services. In *International Confernece on Social Computing (SocialCom)*, pages 656–665. IEEE.

Klimek, P. and Thurner, S. (2013). Triadic closure dynamics drives scaling laws in social multiplex networks. *New Journal of Physics*, 15(6):063008.

Liu, Y., Gummadi, K. P., Krishnamurthy, B., and Mislove, A. (2011). Analyzing facebook privacy settings: User expectations vs. reality. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 61–70, New York, NY, USA. ACM.

Macal, C. M. and North, M. J. (2011). Introductory tutorial: Agent-based modeling and simulation. In *Proceedings of the 2011 Winter Simulation Conference (WSC)*, pages 1451–1464. IEEE.

Madejski, M., Johnson, M., and Bellovin, S. (2012). A study of privacy settings errors in an online social network. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 340–345. IEEE.

Mitchell, T. M. (1997). Decision Tree Learning. In Tucker, C. L. B., editor, *Machine Learning*, chapter 3, pages 52–80. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

Sánchez, D. and Viejo, A. (2015). Privacy risk assessment of textual publications in social networks. In *Proceedings of the International Conference on Agents and Artificial Intelligence*, pages 236–241.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47.

Shehab, M., Cheek, G., Touati, H., Squicciarini, A. C., and Cheng, P.-C. (2010). Learning based access control in online social networks. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 1179–1180, New York, NY, USA. ACM.

Sinha, A., Li, Y., and Bauer, L. (2013). What you want is not what you get: Predicting Sharing Policies for Text-based Content on Facebook. In *Proceedings of the 2013 ACM workshop on Artificial intelligence and security - AISec '13*, pages 13–24, New York, New York, USA. ACM Press.

Squicciarini, A. C., Sundareswaran, S., Lin, D., and Wede, J. (2011). A3p: Adaptive policy prediction for shared images over popular content sharing sites. In *Proceedings of the 22Nd ACM Conference on Hypertext and*

Hypermedia, HT '11, pages 261–270, New York, NY, USA. ACM.

Toch, E., Sadeh, N. M., and Hong, J. (2010). Generating default privacy policies for online social networks. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pages 4243–4248, New York, NY, USA. ACM.

Twitter (09-2015). Statistics. https://about.twitter.com/company.

Zimmer, M. (2010). "but the data is already public": On the ethics of research in facebook. *Ethics and Information Technology*, 12(4):313–325.