2016 IEEE 2nd International Conference on Collaboration and Internet Computing

Using Transfer Learning to Identify Privacy Leaks in Tweets

Saul Ricardo Medrano Castillo
Department of Information Systems
University of Maryland Baltimore County
Baltimore, MD 21250
Email: medrano1@umbc.edu

Zhiyuan Chen
Department of Information Systems
University of Maryland Baltimore County
Baltimore, MD 21250
Email: zhchen@umbc.edu

***Abstract*—Users of online social networks often disclose a lot of sensitive information intentionally or unintentionally, allowing different organizations such as the government, advertising companies, or criminals to exploit such information. In this paper, we focus on identifying privacy leaks such as being pregnant and being drunk in the content of tweets. This problem is non trivial for two reasons. First, we need to differentiate tweets that indeed contain privacy leaks from tweets that do not. E.g., a tweet may talk about a celebrity getting pregnant or selling products for pregnant women and thus is not privacy sensitive. Second, most existing solutions build a supervised learning model for each type of private leaks, but there could be many types of leaks so such solutions require labeling a large number of tweets for each type of leaks, which could be quite tedious and not easily generalizable. Our main contribution is that we apply transfer learning techniques such that we can use training data for one type of privacy leaks for another type of leaks which shares some common ground but is not exactly the same. This greatly reduces the labeling effort and makes our solution more generalizable. Experimental results validated the benefit of our approach: only 7% of data for the new type of leaks need to be labeled to achieve similar results as using 100% labeled data.**

***Keywords*-privacy; social network; transfer learning;**

I. Introduction

Social networks have grown exponentially in recent years. However, many users never take into account that some information they post on social network can be used in the future to harm them. Such information may include information about their health, location, age, religion sexual preferences [1], etc.

One solution to this problem is to develop an automatic tool that can detect privacy leaks from users' posts and warn them before a post containing leaks is published. Although this sounds easy, it is quite difficult in practice due to two key challenges. First, users are not just talking about themselves or their friends, they often talk about celebrities or forward news articles (e.g., retweet), or they could use social network to advertise their own business. For example, one can search for tweets containing the word pregnant but many such tweets are talking about celebrities being pregnant or advertising baby products rather than talking about the users themselves getting pregnant. Below is a sample tweet "Hollyoaks actress Ruby O'Donnell daunted by pregnancy storyline" . Clearly it does not talk about the author of the tweet but rather some actress.

Second, although more sophisticated data mining techniques can be used to automatically classify whether a tweet contains privacy leak, most such techniques require a large and labeled training data set, which is difficult to obtain. Further, a model trained for one type of leaks may not be easily generalized to another type of leaks.

There has been some studies on identifying privacy leaks in tweets. Mao et al. [2] proposed a classification based solution to identify three types of privacy leaks in tweets: revealing dates of vacation plans (which may lead to break-in), tweeting under the influence of alcohol, and revealing medical conditions. In their solution, one classifier is built for each type of privacy leaks and supervised learning methods are used. So their solution requires manual labeling of training data for each type of leaks.

Islam et al. [3] proposes solution to a slightly different problem. Rather than specifying whether a tweet contains private information, they try to classify twitter users into those who reveal a lot of private information from those who do not. They use topic modeling, sentiment analysis, and name entity recognition in their feature extraction process. However, their solution has two shortcomings: 1) their solution cannot identify privacy leaks at individual tweet level (their solution is at user level); 2) their solution still need a lot of labeling efforts (indeed they rely on crowd sourcing to label tweets but the quality of crowd sourcing is often questionable).

In summary, all these solutions require manual labeling of a large number of tweets to be effective and it is unclear how they can be easily generalized when a new type of privacy leaks is considered.

The main contribution of this paper is to apply transfer learning techniques such that we can use training data for one type of privacy leaks for another type of leaks which shares some common ground but is not exactly the same. This greatly reduces the labeling effort and makes our solution more generalizable. . Transfer learning is a hot topic in the field of AI and machine learning [4]. It studies how to use training instances from one domain (called *source domain*) in another domain (called *target domain*). Transfer

978-1-5090-4607-2/16 $31.00 © 2016 IEEE
DOI 10.1109/CIC.2016.76

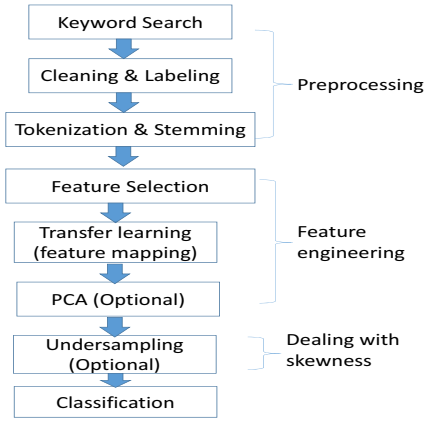


Figure 1. Architecture of Our Privacy Leak Identification System

learning has been used for sentiment analysis [5] and crowd selection on twitter [6]. However to the best our knowledge, it has not been used to identify privacy leaks.

We assume that we have sufficient labeled tweets as training data for one type of privacy leaks, but only have a few labeled tweets for another type of privacy leaks. We applied two transfer learning algorithms proposed in [7] and [8] to classify tweets with two types of privacy leaks: 1) those talking about the author of the tweet being pregnant, and 2) those talking about the author of the tweet being drunk. Although our solution applies to other types of privacy leaks, we choose these two types of privacy leaks because they share some similarity (both are about personal events), and also have some differences (words indicating being pregnant are often different from words indicating being drunk). For transfer learning to work, there need to be some commonality between the domains.

Tweets are also hard to classify because they are short (no more than 140 characters), contain a lot of slang, and have a skew distribution (usually there are far more tweets without privacy leaks than tweets containing leaks). Our solution uses several techniques to address these issues, including using polarity and subjectivity features generated by sentiment analysis in addition to word features, using Principle Component Analysis for dimension reduction, and using under sampling to address skewness issue.

The rest of the paper is organized as follows. In Section II, we describe our solution. In Section III we present experimental results. We conclude the paper in Section IV.

II. Our Approach

Figure 1 shows the architecture of our privacy leaks identification framework. Our framework contains four phases: 1) a preprocessing phase to collect and clean data; 2) a feature engineering phase to generate features for classification; 3) a undersampling phase to deal with skewness issue, 4) building a classification model.

In the preprocessing phase, the system first uses keyword search to collect tweets relevant to a certain type of privacy leaks (called a domain). It then cleans the data and users need to label some tweets. If the system already has labeled tweets for another domain, users only need to label a small number of tweets in the new domain. The system then applies tokenization and stemming to generate a list of words.

Next the system conducts a feature engineering process to generate features used for classification. First it selects both word features and features generated by sentiment analysis. Note that tweets in different domains (types of privacy leaks) usually have some common word features but also many features unique to a specific domain. The system uses a transfer learning method to map domain specific features from different domains to a common feature space. Finally, there could be still too many features. So the system uses a dimension reduction technique (we used Principal Component Analysis in this paper). We selected principal components that account for 80% total energy.

The training data often have far more tweets without privacy leaks (called negative instances) than those with privacy leaks (called positive instances). So the system uses undersampling to overcome the skewness issue. The system undersamples the negative training instances such that both classes have roughly the same number of training instances. Finally, a classification method is used to build a classification model to identify tweets with that type of privacy leaks.

Next we describe the details of preprocessing, feature selection, and transfer learning.

A. Preprocessing and Feature Selection

The preprocessing phase includes: 1) keyword search to retrieve relevant tweets; 2) cleaning and labeling; 3) tokenization and stemming. We will give more details for step 1 and 2.

For each type of privacy leaks, we conducted several keyword search using different keywords through the Twitter API to retrieve relevant tweets. We used several keyword searches because a single keyword search often cannot cover all relevant tweets. For example, for the privacy leaks related to being pregnant, a keyword search “pregnant” or “pregnancy” will return mostly tweets not about the author of the tweets being pregnant but about a celebrity being pregnant or advertisement of some baby products. So this search can be used to collect negative training instances. Another search “I am pregnant” can be used to collect mostly positive training instances (i.e., those indicating the author of a tweet is pregnant). However, not all positive tweets satisfy this search condition so we need to run other searches as well. We then union the results of these searches with duplicates removed. We then manually labeled each retrieved tweet as either positive (i.e., saying the author is pregnant)

or negative. We also cleaned the tweets by removing URLs and hash tags. We did add a feature indicating whether the tweet has a URL.

Feature selection includes selecting word features and sentiment related features. We use odd ratio [9] to select word features. For each word w, let n_{11} be the number of positive tweets that contain w, n_{10} be the number of negative tweets that contain w, n_{01} be the number of positive tweets that do not contain w and n_{00} be the number of negative tweets that do not contain w. The odd ratio of w equals $\frac{n_{11}n_{00}}{n_{01}n_{10}}$. Clearly, the higher the odd ratio, the more likely w will appear in positive tweets than in negative tweets. We set a threshold of odd ratio and only use the frequent words (we picked the top 200 words in terms of frequency) with odd ratio higher than the thread as word features. We also generate a small set of bigrams formed by two consecutive frequent words as features if the bigram's frequency is over a threshold and its odd ratio is over the above threshold.

We also used polarity and subjectivity [10] as sentiment related features. Subjectivity measures whether a tweet is factual or an expression of an opinion. Polarity measures whether a tweet expresses a positive or negative opinion of the subject matter. We used a sentiment analysis package to generate these two features.

B. Transfer Learning

Transfer learning considers the case when we have sufficient training data in a source domain but insufficient training data in a target domain. In our setting each domain is a type of privacy leaks. Note that tweets in different domains have different characteristics and contain different words. For example, suppose the source domain is privacy leaks about being pregnant and the target domain is privacy leaks about being drunk. Tweets in the source domain often contain words related to pregnancy such as "pregnant", "eat" (often in tweets talking about impact of pregnancy), and "month" (often in tweets talking about length of pregnancy). Tweets in the target domain often contain words related to drunk such as "drunk", "wasted" and "sleep" (often in tweets talking about impact of being drunk). So we cannot directly build a model using training data from the source domain for the target domain.

Transfer learning helps bridge the gap between the source and target domain. In this paper we focus on two transfer learning techniques: the Spectral Clustering method proposed in [7] and the SVD method proposed in [8]. Both techniques try to bridge the gap between features from the source and target domain (also called feature-based transfer learning). Next we present each algorithm and explain how they can be applied in our setting.

Spectral Clustering Method: This method was initially used for sentiment analysis over different types of products [7]. We adapt it to identification of privacy leaks. One key observation is that tweets with different types of privacy leaks often both share some common words (called *domain independent*) and have words specific for one type of leaks (called *domain specific*).

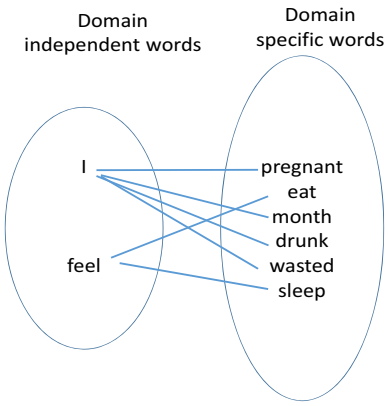


Figure 2. Spectral Clustering Example

Figure 2 shows an example. On the left hand side are domain independent words and the one right hand side are domain specific words. "I" is domain independent and "pregnant" and "wasted" are domain specific because "pregnant" only appears in the Pregnant domain and "wasted" only appear in the Drunk domain. Note that due to this mismatch in features, we cannot directly use training data from the Pregnant domain in the Drunk domain because test data in the Drunk domain will have different words (features) from the training data.

The spectral clustering method tries to link domain specific words through domain independent words. The observation is that domain independent words often co-occur with domain specific words in tweets. We can add an edge between a pair of such words if they co-occur often. This forms a bipartite graph. For example, in Figure 2, "I" and "pregnant" often co-occur in tweets in the Pregnant domain, and "I" and "wasted" often co-occur in tweets in the Drunk domain. So "pregnant" and "wasted" can be linked through "I" (or put into the same cluster by Spectral Clustering). Spectral clustering method tries to map domain specific features to a common feature space with lower dimensions. This common feature space is formed by a set of augmented features generated by applying spectral clustering on the bipartite graph.

Let D_s be the training data in source domain and D_t be the training data in the target domain. Suppose there are m features in $D_s \cup D_t$ and l of them are domain independent features and the remaining $m-l$ are domain specific features. Figure 3 shows the Spectral Clustering Algorithm.

The algorithm basically applies spectral clustering to the bipartite graph between domain independent and domain specific features. We do not need the actual clustering step because we just need to use the common feature space for

Input: Training data in source domain D_s and training data in target domain D_t (only a fraction of D_t is labeled).
$A_1, \ldots, A_{m-l}$ are domain specific features and $A_{m-l+1}, \ldots, A_m$ are domain independent.
Output: a new training set with domain independent features and augmented features $A'_1, \ldots A'_k$.
(1) Construct a similarity matrix $\mathcal{A}$ where $\mathcal{A}(i,j)$ represents the number of times feature A_i and A_j co-occur where A_i is domain independent and A_j is domain specific or vice versa. If A_i and A_j are both domain independent or specific we set the entry $\mathcal{A}(i,j)$ to 0.
(2) Find the largest k Eigen vectors of $\mathcal{D}^{-1/2}\mathcal{A}\mathcal{D}^{1/2}$ where $\mathcal{D}$ is a diagonal matrix with $\mathcal{D}(i,i) = \sum_j \mathcal{A}(i,j)$.
(3) For each selected Eigen vector $u_i, 1 \leq i \leq k$ and training instance x_j, take the first $m-l$ columns as $u_i[1:m-l]$ and $x_j[1:m-l]$ and compute augmented feature A'_i which equals $u_i[1:m-l]^T x_j[1:m-l]$

Figure 3. Spectral Clustering Algorithm

classification. At step 1 the algorithm creates a similarity matrix $\mathcal{A}$ between domain dependent and domain specific features based on co-occurrence. It then finds the k Eigen vectors associated with the largest Eigen values of the normalized version of $\mathcal{A}$. The first $m-l$ columns of selected Eigen vectors are used to map the $m-l$ domain specific features of training data into k augmented features. Now the training data consists of training instances from both source and target domain and have the same set of features (domain independent features plus augmented features).

SVD Method: This method was initially used for spam detection [8]. The algorithm is shown in Figure 4.

The algorithm first applies SVD to transpose of D_t (step 1) and computes similarity of two terms based on reduced matrix using Cosine similarity (step 2 and 3). So the basic idea is similar to spectral clustering: if two terms co-occur very often then they will be considered similar. Features only appearing in the target domain will be linked to the most similar domain independent feature. Note that the computation of similarity does not require labels so we can use unlabeled data in the target domain.

The main difference of this algorithm to Spectral clustering is that it does not add new features. Instead it maps training data in both source and target domain to the target domain's feature space.

For an instance in source domain, the method drops all features only in the source domain because they will not appear in test data in the target domain. The method keeps features common in both domains, and sets target domain only features to one if their most similar common feature is one in that tweet (step 4 to 7). For example, suppose the word "sleep" appears only in target domain and it is most similar to the word "feel" which is a common feature. If a tweet in the source domain contains the word "feel", then the feature for word "sleep" will be set to one even if this tweet does not contain the word "sleep". The rationale is that if a tweet talking about feeling in the source domain (Pregnant) is private, then a tweet talking about feeling in the target domain (Drunk) is also likely private. This allows us to use training data in the source domain to classify data in the target domain.

For an instance in the target domain, the SVD method keeps features only in the target domain and sets a common feature to one if the common feature's most similar non common feature is one (step 8 to 11). For example, if "sleep" appears in a tweet in the target domain then the most similar common feature "feel" will be set to one as well.

Table I
CHARACTERISTICS OF DATA SET

Data Set	Training or Testing	# of Positive Tweets	# of Negative Tweets
Pregnant	Training	328	1015
Pregnant	Testing	140	260
Drunk	Training	400	892
Drunk	Testing	140	260

The cost of both algorithms are dominated by the cost of SVD or spectral clustering. The computational complexity of spectral clustering is $O(nm^2+m^3)$ and the complexity of SVD is $O(\min(nm^2, mn^2))$, where n is number of training instances and m is number of features.

III. EXPERIMENTS

Datasets: Two types of privacy leaks are considered: being pregnant and being drunk. We collected tweets related to these two domains using the method described in Section II-A. For each data set we randomly split it into training and testing. Table I describes the two data sets used for the experiment.

Metrics: We used Area Under ROC Curve (AUC). Since we want to identify positive tweets (tweets with leaks), we use precision and recall for the positive class.

Implementation: The Pregnant domain is used as source and the Drunk domain is used as target. We tried a number of data mining algorithms, including Adaboost, KNN, SVM, Random Forest, J48, and K-means. For K-means, we first apply K-means clustering with $K = 2$, and then assign test tweets to the cluster with the nearest cluster center. We used Weka 3.8 to run these algorithms. We also tuned parameters for various mining algorithms (e.g., the number of trees in Random Forest) and used the values that gave the best mining quality. We implemented preprocessing, feature engineering, and undersampling in Python. NLTK package was used for text processing and TextBlob package was used

Input: Training data in source domain D_s and training data in target domain D_t
(only a fraction of D_t is labeled).
Let $\mathcal{A}_C$ be the set of common features in D_s and D_t and
$\mathcal{A}_{\bar{C}}$ be the set of features in D_t but not in D_s.
Output: a new training set with features in D_t.
(1) Apply SVD on transpose of target data D_t^T to get $D_t^T = U\Sigma V$
(2) Compute matrix $T = U_k\Sigma_k$ where U_k is first k columns of U and Σ_k is first k factors
(3) For each pair of features A_i and A_j where $A_i \in \mathcal{A}_C$ and $A_j \in \mathcal{A}_{\bar{C}}$ compute similarity $sim(A_i, A_j)$
as $cos(T_i, T_j)$ where T_i (T_j) is the i-th (j-ith) row in T and cos is cosine similarity.
(4) For each instance x_i in D_s
(5) drop features in source domain only and keep values of features in $\mathcal{A}_C$
(6) for each feature A_j only in target domain, find the most similar feature A_j* in $\mathcal{A}_C$ and
set A_j to one if A_j* is one.
(7) End For
(8) For each instance x_i in D_t
(9) keep values of features in $\mathcal{A}_{\bar{C}}$
(10) for each feature A_j in $\mathcal{A}_C$, find the most similar feature A_j* in $\mathcal{A}_{\bar{C}}$ and set A_j
to one if A_j* is one.
(11) End For

Figure 4. SVD Algorithm

for sentiment analysis. We set the threshold for odd ratio at 1.2.

We implemented the two transfer learning algorithms. Since existing work uses training data only in one domain [2], we compared our transfer learning algorithms with a No Transfer Learning algorithm that only uses training data in the target domain.

Results when using all training instances in each domain: We first report results of various data mining algorithms when all labeled training instances are used in each domain. This helps us determine which classification algorithm is most appropriate when there is sufficient training data. Figure 5 reports the results on Pregnant data and Figure 6 reports the results on Drunk data. AUC, precision, and recall are reported in separate subgraphs.

In terms of AUC, the results show that Adaboost has the best performance for both data sets. In terms of precision, Adaboost and SVM both have the best results. In terms of recall, J48 is the best for Pregnant and Random Forest is the best for Drunk. Adaboost has the best overall results. K-means clustering givens the worst results in both data sets. This is because K-means clustering often generate one cluster with most instances and most positive tweets in testing data are not classified correctly by K-means.

Benefits of Using Sentiment Analysis: Table II reports the results of Adaboost on both data sets using all training data. We tested two case: one case with sentiment features and the other without. The results show that sentiment analysis does improves mining quality slightly. The improvement is more significant on Drunk data, possibly because in Pregnant domain tweets are often quite emotional (so sentiment features are not very useful in distinguishing positive tweets from negative ones) but in Drunk domain only positive tweets are emotional.

We also conducted experiments on benefits of PCA and undersampling and we found that without sentiment features, these two methods do bring some benefits. But their benefits seem to become insignificant once sentiment features are used. So we do not report results for PCA and undersampling.

Table II
IMPACT OF USING SENTIMENT ANALYSIS (USING ADABOOST)

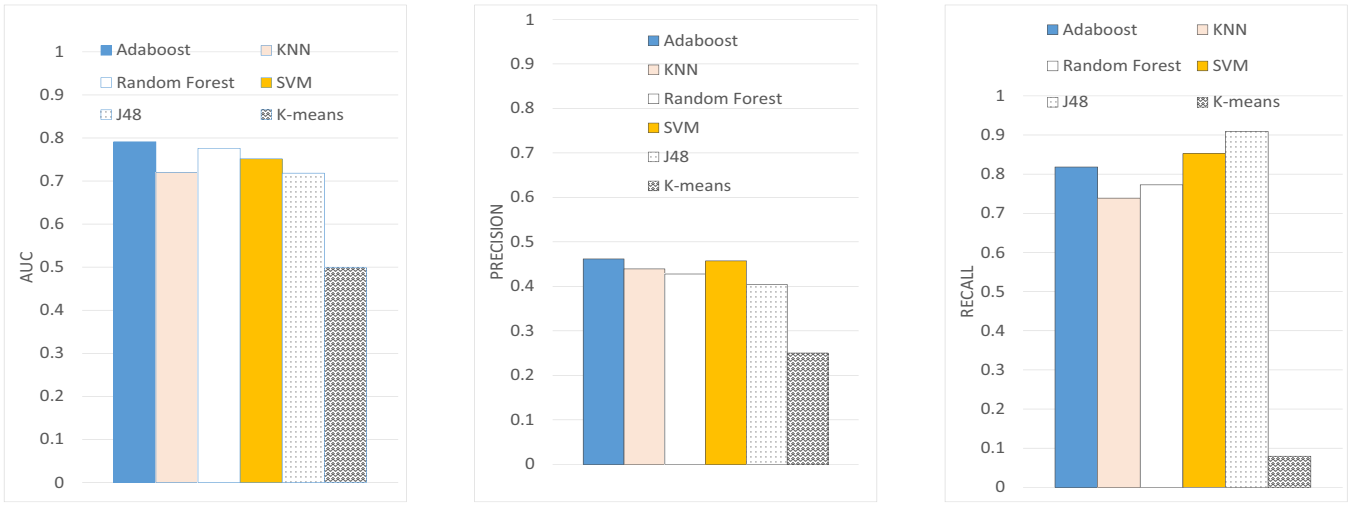
Data Set	Using sentiment features	AUC	Precision	Recall
Pregnant	No	0.78	0.45	0.82
Pregnant	Yes	0.79	0.46	0.82
Drunk	No	0.68	0.57	0.47
Drunk	Yes	0.73	0.58	0.58

Results for transfer learning: Next we study the benefits of transfer learning. We assume that users have sufficient labeled tweets in the Pregnant domain, but have very few labeled instances in the Drunk domain. So we only use a fraction of labeled training instances in the Drunk domain and use transfer learning to use training data from Pregnant.

Figure 7 reports the results of various mining algorithms using a fraction of training instances in Drunk domain and all training instances in Pregnant domain by SVD algorithm. Again, AUC, precision, and recall are shown in separate subgraphs. Figure 8 reports the results using Spectral Clustering.

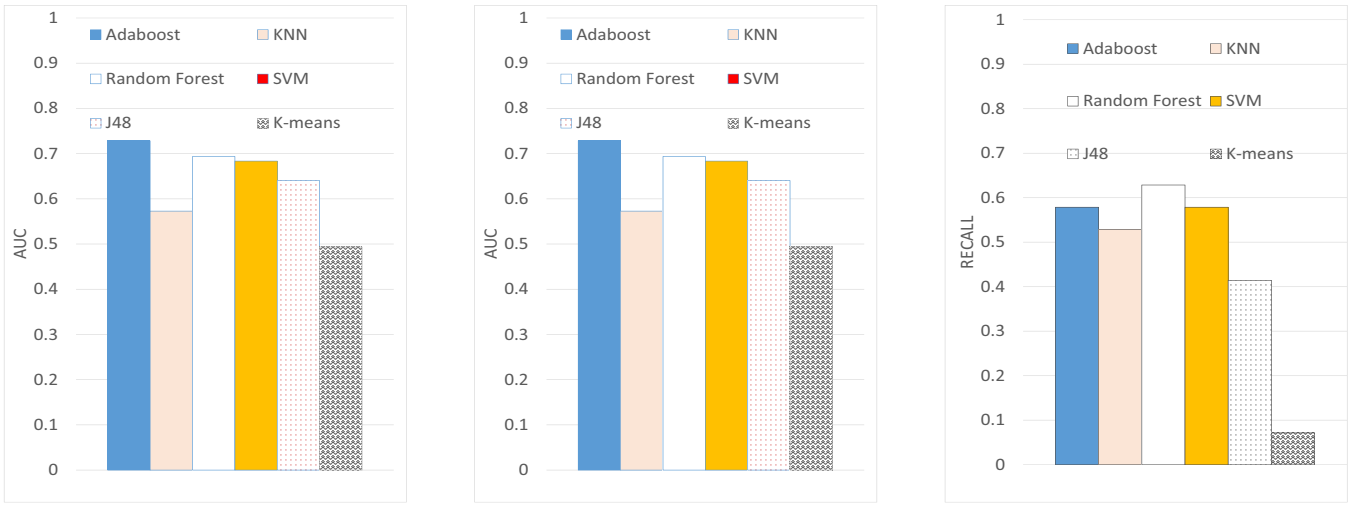
The results show that all algorithms have worse mining quality when a small fraction of training instances in Drunk domain are used. This is expected because to achieve good classification results we need sufficient training instances. However the results for Random Forest improve quite quickly as slightly more training instances are used due to transfer learning. Using SVD, the AUC for Random Forest using 7% of training instances from Drunk domain is about the same as the results for using 100% training instances.

Random Forest gives the best results in terms of AUC using both transfer learning algorithms. K-means gives the worst results. For precision, SVM gives the best results



(a) AUC (b) Precision (c) Recall

Figure 5. Results of various mining methods on Pregnant data using all labeled training data

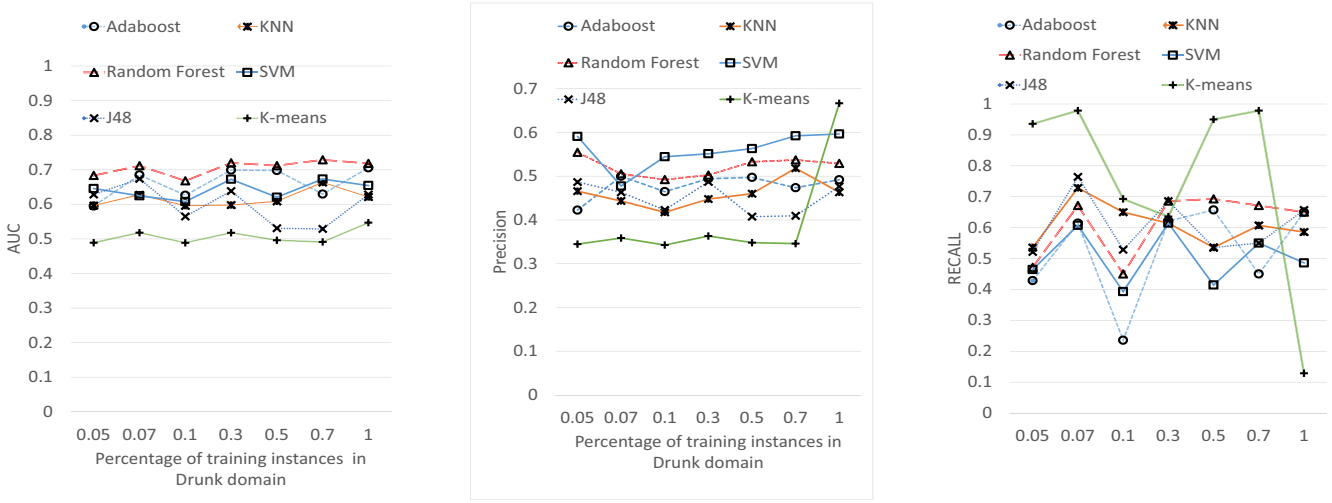


(a) AUC (b) Precision (c) Recall

Figure 6. Results of various mining methods on Drunk data using all labeled training data

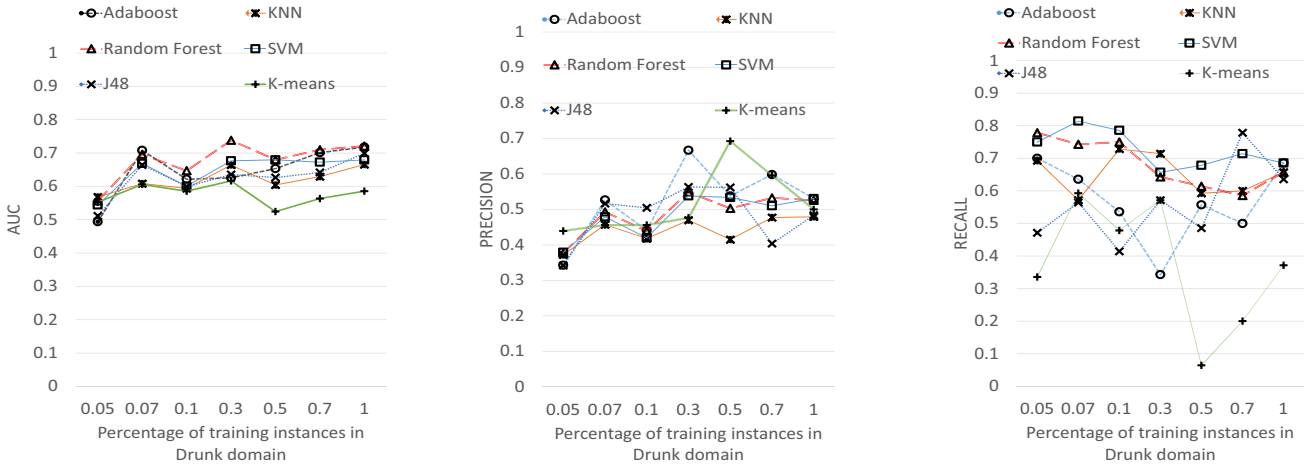
for SVD and Adaboost gives the best results for Spectral Clustering. Random Forest still give good results (it is the second best for SVD). For recall, the results for K-means fluctuates. We found that it often assigns most test cases to one class and as a result sometimes it has very high recall (when the class is correct) and sometimes it has extremely low recall (when the class is wrong). Random Forest gives good results for both algorithms. Overall, Random Forest gives the best overall performance when transfer learning is used.

Figure 9 reports the results for SVD, Spectral Clustering, and a No Transfer Learning algorithm that only uses training instances from the Drunk domain (not using instances from Pregnant domain). All three algorithms use Random Forest because it has the best results compared to other mining methods. The results show that in terms of AUC, SVD gives the best results. Spectral Clustering has results slightly better than No Transfer Learning when small fraction of training instances in Drunk domain are used. In terms precision, SVD is still the best and Spectral Clustering has the worst results. In terms of recall, Spectral Clustering has good results when small fraction of training data in Drunk domain are used, but



(a) AUC (b) Precision (c) Recall

Figure 7. Results of various mining methods on Drunk data using SVD transfer learning and a fraction of labeled Drunk data



(a) AUC (b) Precision (c) Recall

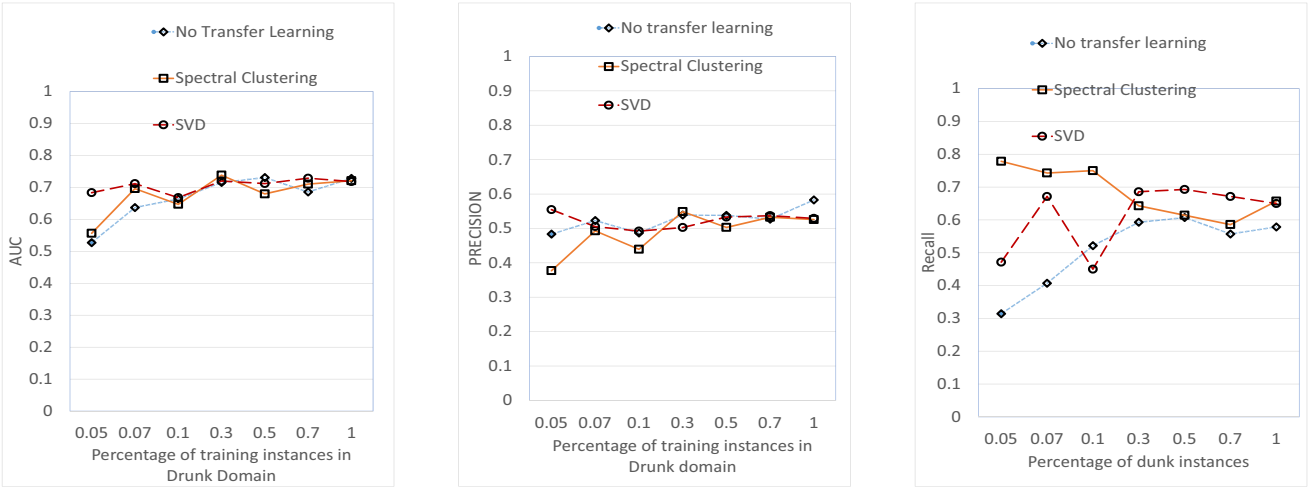
Figure 8. Results of various mining methods on Drunk data using Spectral Clustering transfer learning and a fraction of labeled Drunk data

deteriorate as more training data are used. Overall, SVD has the best results.

We tried to understand why Spectral Clustering does not work as well as SVD. One possible reason is that we used relatively small data sets and for Spectral Clustering to work a much larger data set may be needed. In addition, Spectral Clustering is more complex than SVD. SVD tries to identify features in target domain that are most similar to a common feature and uses that for transfer learning. We looked at these features and found that they indeed co-occur frequently in tweets. Spectral Clustering instead comes up with a set of augmented features, which is hard to verify. In addition, it does not differentiate features unique to source domain from features unique to target domain. So it may link two features both unique to source domain (e.g., "eat" and "pregnant" in Figure 2), which has little use for transfer learning because they do not appear in the target domain.

IV. Conclusion

In this paper we studied the problem of identifying different types of privacy leaks in tweets. One major challenge of this problem is the lack of labeled training data. We applied two transfer learning algorithms to address this challenge. One of the algorithms, SVD, only requires a small number of labeled training data (only about 90 tweets in our experiments) by reusing training data from other types of privacy leaks. This greatly reduces the labeling



(a) AUC (b) Precision (c) Recall

Figure 9. Results of two transfer learning methods vs. not using transfer learning on Drunk data. Random Forest is used.

efforts and makes it easy to extend the solution to new types of privacy leaks. The success of the algorithm relies on linking domain specific features to features common in multiple domains. Interestingly, we found that the other transfer learning algorithm (Spectral Clustering) does not work well, possibly due to its complexity and need for more training data.

As future work, we will consider whether transfer learning can be used to identify other types of privacy leaks on twitter. We do expect our method will work as long as two types of privacy leaks share some common ground. The precision and recall of our methods are also not very high. We will investigate ways to improve them, including using larger data sets and using other feature engineering techniques such as deep learning.

REFERENCES

[1] S. Livingstone, "Taking risky opportunities in youthful content creation: teenagers' use of social networking sites for intimacy, privacy and self-expression," *New media & society*, vol. 10, no. 3, pp. 393–411, 2008.

[2] H. Mao, X. Shuai, and A. Kapadia, "Loose tweets: An analysis of privacy leaks on twitter," in *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, ser. WPES '11. New York, NY, USA: ACM, 2011, pp. 1–12. [Online]. Available: http://doi.acm.org/10.1145/2046556.2046558

[3] A. Caliskan Islam, J. Walsh, and R. Greenstadt, "Privacy detective: Detecting private information and collective privacy behavior in a large social network," in *Proceedings of the 13th Workshop on Privacy in the Electronic Society*, ser. WPES '14. New York, NY, USA: ACM, 2014, pp. 35–46. [Online]. Available: http://doi.acm.org/10.1145/2665943.2665958

[4] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. on Knowl. and Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2009.191

[5] M. Kaya, G. Fidan, and I. H. Toroslu, "Transfer learning using twitter data for improving sentiment classification of turkish political news," in *Information Sciences and Systems 2013*. Springer, 2013, pp. 139–148.

[6] Z. Zhao, D. Yan, W. Ng, and S. Gao, "A transfer learning based framework of crowd-selection on twitter," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 1514–1517.

[7] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, "Cross-domain sentiment classification via spectral feature alignment," in *Proceedings of the 19th International Conference on World Wide Web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 751–760. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772767

[8] J.-n. Meng, H.-f. Lin, and Y.-h. Yu, "Transfer learning based on svd for spam filtering," in *Intelligent Computing and Cognitive Informatics (ICICCI), 2010 International Conference on*. IEEE, 2010, pp. 491–494.

[9] F. Mosteller, "Association and estimation in contingency tables," *Journal of the American Statistical Association*, vol. 63, no. 321, pp. 1–28, 1968.

[10] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining text data*. Springer, 2012, pp. 415–463.