User Comment Analysis for Android apps and CSPI Detection with Comment Expansion

Lei Cen, Luo Si, Ninghui Li
Computer Science Department
Purdue University
West Lafayette, IN 47907, USA
{lcen, lsi, lin}@purdue.edu

Hongxia Jin
Samsung Information
Systems America
San Jose, CA 95134, USA
hongxia.jin@sisa.samsung.com

ABSTRACT

Along with the exponential growth on markets of mobile apps, comes the serious public concern about the security and privacy issues. User comments serves as a valuable source of information for evaluating a mobile app, for both new users and developers. However, for the purpose of evaluation on the security/privacy aspects of an app, user comments are not always directly useful. Most of the comments are about issues like functionality, missing feature or just pure emotional expression. Therefore, further efforts are required in order to identify those Comments with Security/Privacy Issues (CSPI) for future evaluation. In this paper, a dataset of comments is collected from Google Play, and a two dimensional label system is proposed to describe those CSPI within it. A supervised multi-label learning method utilizing comment expansion is adopted to detect different types of CSPI described by this label system. Experiments on the collected dataset shows that the proposed method outperforms the method without the comment expansion.

1. INTRODUCTION

New challenges come with the exponentially growing markets of mobile apps. On one side, comparing to traditional software markets, markets like Google Play and Apple Store have lower entering threshold for developers and faster financial payback, hence greatly encourage more and more developers to invest in this thriving business. One result out of this is the huge amount of mobile apps with great diversity. Therefore, to control the quality of apps, especially the security risk of them across the whole markets becomes a important issue to all that involved. On the other side, public concerns about privacy issues with on-line activity and mobile phones are also elevating, demanding a mobile environment with more respect to users' privacy.

The infection rate of real malicious mobile apps over a market can only be estimated. It is reported to be about 0.28% in [10]. The rest of them are assumed benign apps but are not free of security issues. Many misbehaviors of a mobile app may lead to real security/privacy issues. For example, adding too much or inappropriate advertisement may lead to phishing and scareware; unresolved In-App-Purchase (IAP) may lead to fraud; unnecessary running in background may be connected with unauthorized access to personal data.

User comments are valuable feedback for both new users and developers. Many security/privacy issues can be revealed by user comments, including those which may not be so easy to be discovered from other sources (i.e. unresolved IAP issue). However, most of the negative comments are not necessarily Comments with Security/Privacy Issues (CSPI). Some of them are non-informative comments [4], including vague statement and pure emotional expressions. Others may be complaining about attractiveness, quality or even cost of the app [5], which normally have nothing to do with security/privacy. In order to make use of the comments to reveal the issues that are really related to security/privacy of an app, CSPI need to be detected first to avoid all those irrelevant comments. This paper provides a novel method to deal with this problem. A set of comments are first collected and investigated. Based on observations from the comments, a two dimensional label system is designed to describe CSPI from two different perspectives. With this label system, a CSPI Detection with Comment Expansion (CDCE) method is proposed to first use keyword-based filtering method to narrow down the scale of comments in concern, then applies a supervised multi-label learning method to identify different types of CSPI.

The contribution of this paper are listed as following:

- Instead of using a list of label for different issues, this paper presents a two-dimensional label system, picturing the "What" and "When" of the occurrence of a reported CSPI, providing an additional dimension to better understand the relationship between different issues.
- A supervised learning method is proposed to solve this multi-label problem. Comment expansion is adopted to utilize the relationship between comments and a post-process for the relationship between labels. Both relationships are proven to be useful in improving the CSPI detection performance based on the collected dataset.

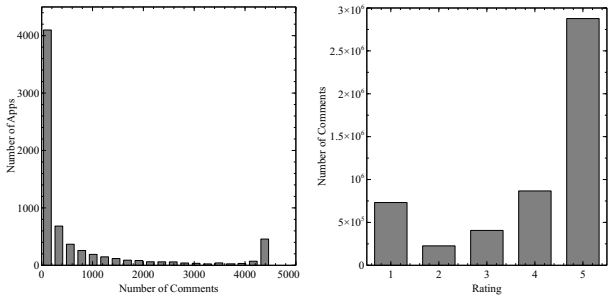
The rest of the paper proceeds as follows. Section 2 describes related works. Section 3 discuss the collection of data and the design of the label system. Section 4 presents the method for CSPI detection. Section 5 demonstrates the evaluation process of the proposed method. Section 6 lists the limitation of this paper with future work and Section 7 concludes the paper.

2. RELATED WORKS

User comment are utilized by some works [5, 6, 4] to evaluate mobile apps. Some researchers [6] aim at the task to extract new/changed requirement for new version of the app. It proposes Aspect and Sentiment Unification Model (ASUM) to extract the topics of comments. ASUM is an extension of Latent Dirichlet Allocation (LDA) [2] by incorporating both topic modeling and sentiment analysis. Another work [5] uses LDA model to identify different topics from those negative comments, in order to provide insight about why an app is getting low ratings. Our work focus on a finer level to investigate only CSPI, which is part of the negative comments. Also, some researchers [4] propose a novel method for extracting informative review topics from user comments. A filtering process is applied first to filter out non-informative comments, then LDA is adopted for generating topics from the informative ones. Our work also requires a filtering process, but not from the perspective of the quality of information, but whether the comment is related to security/privacy. All these works mentioned above make use of topic models (LDA), which is an unsupervised method. For our work, the task is not to generate general summarization of the topics of comments of an app, but to identify a specific part (CSPI) of all the comments and further distinguish different types of it. Hence unsupervised method like LDA may not be able to generate the expected and specified types of topics. Therefore, supervised method is adopted in our work, and a label system is manually designed to provide precise task for the learning process.

3. DATA COLLECTION

The data used in this work is collected by crawling the Google Play website. Information about $6,938$ free apps in Google play are downloaded during September through December in 2013. For user comments, the total number of comments collected for these $6,938$ apps is $5,108,538$. The average number of comments for an app in this dataset is 736 and the maximum number is 4500. Figure 1a shows the distribution of number of comments among the apps.



(a) Histogram for the number of apps against the number of its comments in the collected dataset.

(b) Histogram for the number of comments with different ratings in the collected dataset.

Figure 1: A simple view of the collected dataset.

The rating from user comments are highly skewed as shown in Figure 1b (this is previously also reported in [5]), indicating that most of the comments are not complaining about an issue. In addition, many of the comments with poor ratings (< 4.0) are not really about security/privacy issues. As observed from the dataset, complains about the functionality and quality (or attractiveness for Game app) take the majority. Therefore it would be necessary to narrow down the huge set of comment to a more feasible size for further analysis and annotation. In order to do that, a coarse filtering is applied first to create a set of suspect comments. This filtering is keyword-based, as any comment that contains at least one of the predefined keywords will be put into the suspect set. These keywords are manually picked in an iterative way. The initial set of keywords is just $\{security, privacy\}$. New keywords are picked from those that have high co-occurrence probability with current keywords. Comments with the co-occurrence are investigated manually to see how often they are related to security/privacy issues. The final keywords set includes *security, privacy, permission, money, spam, steal, phish*, etc.. To avoid mismatch, different forms of the keywords are also considered, i.g. *unsecure* and *insecure* for *security*, *stole* and *stolen* for *steal* etc. A rating filtering is also applied to only include those comments with poor ratings(< 4.0). The resulting suspect set contains $36,464$ comments from $3,174$ apps.

3.1 CSPI Annotation

It is worth noting that the suspect set still contains not only CSPI. Many of them are not CSPI, due to the simplicity of the keyword-based coarse filtering. Hence the suspect set only serves as a base candidate set of comments which are suspect for CSPI. Further effort need to be done to actually detect CSPI. And different types of CSPI need to be distinguished and treated accordingly.

Various different issues are found from the comments in the suspect set. It would be exhausting to distinguish each of them without abstract concepts by considering the relationship between them. Also, it would be too vague to just distinguish CSPI from non-CSPI comments without further insight of the actual issues. To make a trade-off between the complexity and functionality of the annotation system for describing CSPI, a label *General* is firstly defined for comments that is in general CSPI. In addition, a two dimensional (*Nature*, *Scenario*) label set is defined as shown in Table 1 to provide finer level identification for different CSPI topics. Among these labels, label *Execution* is a super label for *Foreground* and *Background*. And all are sub-label of label *General*.

The two dimensions of label set serve as "What" (*Nature*) and "When" (*Scenario*) of the underlining issues. The dimension *Nature* is used to identify the nature of the security/privacy related misbehavior of the app complained by the comments. These behaviors are normally described explicitly in the comments. On the other hand, the dimension *Scenario* is used to identify the scenario when the issues occur. This is sometimes expressed implicitly in the comments. System, Privacy and Finance are clearly necessary as the direct issues. The label Spam is assigned to the widely used advertising behavior among apps. As a popular and common behavior, however, spamming is closely related some security issues like phishing, scareware etc., hence is also included as one type of issue mentioned in CSPI. The label *Others* is used for other issues not included in *System*, *Privacy*, *Spam* or *Finance*. For example the topic about some apps that can not be normally uninstalled, it is usually an app pre-installed by the vendor without any real issues, but

	Label	Definition	Issues
Nature	*System*	Issues causing negative effect to the system	Freezing the phone, unauthorized downloading
	Privacy	Issues about getting unauthorized access to user info.	Stealing phone number, accounts, emails
	Spam	Issues about unpleasant ads and related.	Annoying ads, spam shortcut on home screen.
	Finance	Issues about suspect money stealing.	unresolved purchase.
	Others	Security / privacy issues not included above.	uninstall issues
Scenario	*Before*	Issues occur before installation of the app.	Requiring too much permissions.
	Execution	Issues occur when the app is executing on the phone.	General complains about spam
	Foreground	Issues occur when the app takes the foreground screen.	Ads occupies too much screen, phishing
	Background	Issues occur when the app is running background.	Notification bar spam
	After	Issues occur after uninstall of the app.	email/SMS spam after uninstall

Table 1: Two dimensional label set

it clearly violates the users' control over the phone and is complained about by many users. Another example for *Others* labeled comments would be those claims that the app is reported by other security apps for some reason. Since these are not direct opinions from the users but only a suspects, they are put with *Others*, instead of with the other four ones according to the reasons said to be reported by other security apps, which may not be true or even explicitly expressed in the comment. The label *Before* is mostly assigned together with *Privacy* to indicating the complains about the permission required by the app, which is available to the user before the installation. And the label *After* is mostly for the email/SMS spam after uninstall. The label *Execution* stands for the most common scenario, and it is further divided into two sub-labels: *Foreground* for the issues occur when the app is running in foreground (occupying the screen) and *Background* for running in background. If the issue in the comment is not specific about foreground or background, only the label *Execution* is applied, otherwise both *Foreground* (*Background*) and *Execution* will be applied.

The suspect set is then annotated manually with these labels. Each label is validated by the agreement of two people. Some statistics of the suspect set with respect to the labels are shown in Table 2. About 30% of the comment in the suspect set are labeled as CSPI. This also means many of the comments, while mentioning some keywords, are not really talking about security/privacy issues. The sample comment for *Whole Set* is one of them. It mentions *stolen* not to claim a financial issue but a issue about stolen idea, which is not about security or privacy. Some comments talk about "money", but only to express that the app is not worth it (Although all the apps we collected are free apps, people can still talk about In-App-Purchase and the paid version of the app.). Another example would be the comments for the apps about security (Anti-virus apps, password manager, etc.). A lot of keywords are mentioned in these comments since the apps are exactly about the issues, but the comments are not necessarily reporting an issue of the apps. Similar things happen with Bank apps. Label *Others* has a very low percentage, indicating the coverage of the four main natures of issues is good.

It is worth noting that misspelling may hinder the performance of detection. The "baught" in the sample for *Finance* serves as an example. Also of important is the fact that a CSPI usually has two or more labels besides *General*. For example, the sample comment for *Spam* is also annotated with *Background*, since it mentions that the spam appears in the notification bar. And the one for *Before* is also annotated with *Privacy*.

4. CSPI DETECTION

The purpose of CSPI detection is to detect certain types of CSPI in the suspect set. This is naturally a multi-label classification problem. Independent Logistic Regression (ILR) is used as a baseline. The proposed CDCE method adds two improvements upon it: Comment Expansion to embed similarity between comments and Post-Process with label correlation to utilize label correlation.

4.1 Feature Extraction

To represent comments for detection, Bag Of Word (BOW) feature is extracted from comment text. Proper pre-processes are applied to the text before feature extraction, including removing stop words and stemming. The dimension of the BOW feature is tuned by removing the rarest and most popular words. With the BOW feature and the annotation with the 11 labels, the suspect set can be represented as X and Y. $X = \{X_1, X_2, \cdots, X_N\}$ is the set of the BOW feature vectors for each comments with N the number of comments. And $Y = \{Y_1, Y_2, \cdots, Y_N\}$ is the set of label vectors for each comments with $Y_i = \{Y_i^1, \cdots, Y_i^{11}\}$ and $Y_i^j \in \{-1, 1\}$ with $Y_i^j = 1$ indicating the presence of the jth label for comment i.

4.2 Independent Logistic Regression (ILR)

There are many works that have been done on multi-label learning from different perspective. G. Madjarov[7] made an extensive empirical comparison of different multi-label learning methods. In the comparison, Binary Relevance (BR) [11] performs similar to classifier chaining (CC) method [9] . Although not the best method among all the competitors, BR shows robust performance and has the advantage of simplicity. For the task of CSPI detection, this paper does not seek to investigate and improve the multi-label learning algorithm in general. Therefore, ILR is chosen as the baseline, which is just BR framework with LR [1] as the base classifier. LR is a widely adopted linear classifier due to its robustness and simplicity. To apply LR to the multi-label problem considering independence between labels, one LR classifier is trained for each of the labels. The objective function of ILR with 2-norm is as following:

$$\min_{w_j, b_j} NLL(X, w_j, b_j) + \lambda(|w_j|_2^2 + |b_j|_2^2)$$

	#	%	Sample Comment
Whole Set	36,464	100%	"Idea *stolen* from ***, and it feels unfinished, or unprofessional"
General	10,636	29.17%	all of the below.
System	1,304	3.58%	"It keeps crashing the phone. It becomes a device administrator."
Privacy	2,513	6.89%	"*Leak*ing GPS location to dvertisers=Top 2 fastest ways to get me to hate your app/guts."
Spam	6,164	16.90%	"Since installing this app I get *spam* in my notifications constantly."
Finance	1,191	3.27%	"I baught $2.00 in *** and I never got them. In really mad about it."
Others	191	0.52%	"I buy my own phone with my own *money* and I cant delete this app from my phone?"
Before	1,611	4.40%	"New versions *permissions* can *steal* all my data and contacts!!!"
Execution	8,456	23.19%	"No need for them to *invade* my *privacy*."
Foreground	2,011	5.52%	"game wont even load due to the pop ups before game starts..splash sceen pop ups suck"
Background	4,171	11.44%	"*spam* emails that I did not send were going out from my email address."
After	36	0.10%	"six months after uninstall, *** *Spam* just keeps on coming"

Table 2: Statistics and sample comment pieces on suspect set

with

$$NLL(X, w_j, b_j) = \sum_{i=1}^{N} ln(1+exp^{-y_i^j(w_j^T X_i + b_j)}), j = 1, \cdots, 11$$

where the $NLL(X, w_j, b_j)$ is the negative log-likelihood function. λ is the regularization parameter which can be obtained by cross validation in training set. This optimization problem can be solved using gradient descent.

4.3 Comment Expansion

User comments has some properties distinguishing them from properly compiled documents. They are normally short, with wrongly spelled words (as mentioned in Section 3.1) and made-up words (i.g. "spamspamspamspam"). Also, different words or phrases may be used for the same opinion. These properties of user comments may harm the performance of CSPI detection since the features may not fully represent the opinion of the comments. This is in a similar situation with the user query in Information Retrieval (IR) problem, where a query submitted to search engine could also be short, misspelled and vary in the choice of words. Due to this motivation, a traditional IR technique called query expansion with pseudo relevant feedback [12] is borrowed here to make comment expansion. Originally, this technique uses the top documents in the retrieval result rank list with respect to the original query as "relevant" documents, and use these document to expand the original query, generating a new query resembling the "relevant" documents. This query expansion is reported to almost always have a positive effect on the retrieval performance [12]. A similar process can be adopted for comment expansion as following. The comment expansion would have two steps:

1. Find "relevant" comments via retrieval.
2. Expand original comment with "relevant" comments.

The relevance between comments is hereby evaluated using the retrieval model. An interesting question would be the scope of the retrieval. Should the candidate "relevant" comments picked from all other comments? or just from the comments from the same app or same category of apps? Besides, the "relevant" comments should only be those posted before the underlining comment. To avoid making complicated query to the retrieval engine, the query contains only the underlining comment with no constrain. A sufficiently long rank list from the retrieval engine will be returned and the scopes and restrictions will be applied to this list to pick "relevant" comments afterwards.

With a set of "relevant" comments, the comment expansion can be conducted by making a convex sum of the original comment and the mean of the set of "relevant" comments on feature level as following:

$$f_{new} = (1-\alpha) f_{old} + \alpha \cdot \frac{1}{|R|} \sum_{f \in R} f$$

where f_{old} is the BOW feature vector of the original comment, f_{new} the feature vector of the expanded comment, and R the set of "relevant" comments with respect to the underlining comment. α serves as a tunable parameter for the degree of effect of the expansion, and will be tuned in experiment for the best performance, so is the size of R.

Comment expansion is an efficient way to utilize the relationship between comments. Other choices like kernel method may require the similarity computation between each pair of comments, the amount of computation grows exponentially with the number of comments. Comment expansion, on the other hand, relies on retrieval engine, and the cost of retrieval for one comment is normally constant, hence the total cost of time is linear to comment number.

4.4 Post-Process with Label Correlation

Label correlation is used in various ways in multi-label learning [7]. As mentioned in Section 3.1, labels hardly appear alone. Hence the correlation between labels could be a valuable source of information. For the CSPI detection task, a simple post-process is applied to embed this information. The post-process uses a second round of ILR with different feature for the comments. The probability output $\hat{Y}$ of the first round ILR classifiers are used as feature in this second round of ILR to predict Y. And the correlation matrix of the labels are utilized into a Laplacian norm [8] in these LR problems. Let $\hat{Y} = \{\hat{Y_i}, \cdots, \hat{Y_N}\}$ the estimated label vector set from the ILR. The objective function of the second round LR is as following:

$$\min_{\hat{w_j}, \hat{b_j}} NLL(\hat{Y}, \hat{w_j}, \hat{b_j}) + \hat{\lambda}(\hat{w_j}^T L \hat{w_j} + |\hat{b_j}|_2^2)$$

with

$$NLL(\hat{Y}, \hat{w_j}, \hat{b_j}) = \sum_{i=1}^{N} ln(1+exp^{-y_i^j(\hat{w_j}^T \hat{Y_i} + \hat{b_j})}), j = 1, \cdots, 11$$

where $L = A - D$ is the Laplacian matrix with A the correlation matrix of the labels Y in training set, and D a diagonal matrix with its diagonal elements the sum of each row of A. These optimization problems can be solved similarly using gradient descent.

5. EXPERIMENTS

5.1 Experiment setting

The evaluation of the proposed CSPI detection method is conducted under the suspect set of comments. As a supervised method, a training set is required for training the model. The suspect set is split in a 50/50 manner into a training set and a testing set. This splitting is based on app level, so the comments for the same app can only be in training or testing set altogether. The BOW feature vector is of length $13,135$ by removing those words with less than 100 or more than $1,000,000$ appearance (in number of comments) in training set. Considering the hierarchy in label set, a comment labeled with a sub-label will be considered a positive sample for a super-label as well. And a sample labeled with a super-label will not be considered either a positive or negative sample for a sub-label. For the baseline ILR, 5-fold cross validation is adopted for finding λs in training set and L-BFGS quasi-Newton method [3] is applied for solving the optimization problems. For comment expansion, tf-idf feature with cosine similarity is adopted for the retrieval model and Lemur[1] as the actual tool for the retrieval. Time constrain is enforced to prevent comment expansion with "future relevant" comments. Set constrain is applied to only allow expansion within training/testing set respectively, and the indexes of retrieval model are built for the two sets separately so that the model parameter like document number and IDF values won't interfere between sets. Three types of scope: All, app, Category are tested with each label and a 5-fold cross validation is used to pick the best of the three in training set for each label. Also, the mixture ratio α and the size of "relevant" document set $|R|$ for the expansion are also fixed by the 5-fold cross validation in training set for each label along with the scope. The size of $|R|$ is selected from $\{1, 3, 5, 10\}$. For post-process with label correlation, the problem solving method is similar to the baseline ILR. The metric for evaluation is micro F1 value. F1 value is the harmonic mean of Precision and Recall. And micro F1 is computed across all sample and all label at once.

5.2 Results and Analysis

The comparison between the proposed CDCE method and the baseline in F1 value is shown in Table 3.

The CDCE$^-$ method indicates the method using only comment expansion without the post-process. The CDCE$^+$ method indicates the method using comment expansion and post-process on all labels. And the CDCE* method indicates the method that pick the models between CDCE$^-$ and CDCE$^+$ by cross-validation for each labels. This evaluation is based on 10 different training/testing sets splitting, and the reported micro F1 values are the means over these ten settings.

[1]http://www.lemurproject.org/

method	micro F1			
	General	*Scenario*	*Nature*	All
ILR	0.7962	0.6713	0.7032	0.7153
CDCE$^-$	0.8037†	0.6740†	0.7159†	0.7223†
CDCE$^+$	0.8004†	0.6814†	0.7096†	0.7225†
CDCE*	**0.8037**†	**0.6836**†	**0.7159**†	**0.7263**†

Table 3: Experiment results on suspect set. † shows the statistical significance based on ILR. It is computed over ten different random splits of the training/testing sets, using one-tailed pair-wise t test with $\alpha = 0.05$.

By comparing CDCE$^-$ and ILR, the general improvement from using comment expansion is obvious. The expansion makes comments "smoother" among similar comments in feature level, and improves the performance of the classifiers against short comments and those with misspelled words. For example, "ads" may sometimes be misspelled to be "add", the expansion adds similar comments' feature into the underlining feature and the feature dimension with respect to "ads" may not be zero anymore, hence the classifier can capture this feature and tend to put the label "Spam" on the comment. On the other side, the effect of expansion is not always positive for all samples. For example, for comments about a Game app, many ones may be talking about "money" because they paid for some item in the game but never got it. These comments should be labeled "Finance". But one comments for the same app may be just talking about how expensive that item is, hence not worth the "money". This one however, is not a CSPI. After expansion, this comment will look much like the others hence be labeled "Finance" as well. Therefore a negative effect of comment expansion is that it may silence some different voice that are making different points while using similar words like others. Nevertheless, the general effect of comment expansion is obviously positive.

The difference between CDCE$^-$ and CDCE$^+$ shows that the post-process does not guarantee an improvement for all labels. Hence CDCE* method is propose to pick a better model between CDCD$^-$ and CDCE$^+$ for each label from training set. CDCD* provides the best performance among others.

A label level comparison between CDCE* and ILR can be found in Table 4, where CDCE* appears to outperform ILR for all labels except *After*. The model behavior for label *After* is different from others mostly due to lack of sample. There are only 36 samples to be split into training and testing set, which practically makes the training of model insufficient and the comment expansion mostly uses comments with different labels as "relevant" comments. Besides *After*, three labels:*Before*, *System* and *Others* do not pass the statistical significant test at α level 0.05, with p-value 8.2%, 5.5% and 8.1% respectively. Other than these, the highest p-value is 0.9% from *Finance*.

The results of picking better model between CDCE$^-$ and CDCE$^+$ are also shown in the P.P. column of Table 4. It

Label	F1		P.P.	Type
	ILR	CDCE*		
General	0.7962	**0.8037**†	No	App
Before	0.6965	**0.7012**	Yes	Cat.
Execution	0.7277	**0.7358**†	Yes	App
Foreground	0.4347	**0.4689**†	Yes	Cat.
Background	0.6674	**0.6750**†	No	App
After	**0.1327**	0.0882	No	App
System	0.3991	**0.4012**	No	App
Privacy	0.7264	**0.7350**†	No	Cat.
Spam	0.8181	**0.8304**†	No	Cat.
Finance	0.5238	**0.5320**†	No	Cat.
Others	0.0235	**0.0384**	No	App

Table 4: Detailed comparison in label level. The † is computed at $\alpha = 0.05$ with one-tailed t test among 10 different training/testing set splits.

appears that all *Nature* labels are not suitable for the post-process, but the *Scenario* ones get a boost based on the results in Table 3. The post-process makes use of the ILR result as feature to predict a label. This prediction may be improved by getting correlation information from other labels, but also suffer from poorly predicted results from ILR. Important words (features) for the *Nature* labels are not so diverse as those in *Scenario*. For example, for label *Spam*, no matter what *Scenario* label come with it, the comment would probably still use the words like "spam" or "ads". Similarly for *Privacy* there are "privacy", "invade", or "permission". To distinguish *Scenario* labels, however, different words are of importance under the condition of different *Nature* label. If given *Spam*, *Foreground* is related to over-sized on screen ads, and *Background* most likely to notification bar spamming. On the other hand, if given *Privacy*, *Foreground* may be related to phishing and *Background* to personal information stealing or abuse. Hence the correlation information may be much more helpful for *Scenario* labels than *Nature* ones.

The scopes of retrieval in comment expansion for each label are listed in the Type column of Table 4. None of classifiers choose to use All comments as candidates for "relevant" comments, and the scope of using the comments of the same app or same Category (Cat.) of apps are both popular. The scope of All comments makes the "relevant" comments too diverse and noisy and the expansion normally lead to some unexpected result. One the other side, app and Cat. scope serves well, providing much high probability of getting "relevant" comments with both similar text content but also similar topic of issues.

6. LIMITATION & FUTURE WORK

As a comment level analysis, one limitation of this work is that it does not provide a risk assessment on the app level. How to evaluate the app's security/privacy risk base on the identification of CSPI would be an interesting work in the future. Another limitation comes from the coarse filtering, where CSPI that do not contain any keywords could be left out by the method. This may due to the variety of language itself or simply misspelling or using made-up words instead of the keywords. Hence further research may lie on how to expand the suspect set as a trade-off between computation overload and detection performance.

7. CONCLUSION

In this paper, a supervised learning method is proposed to detect CSPI for Android apps. This task is formalized as a multi-label learning problem with a two dimensional label system with respect to "What" and "When" of issues reported in CSPI. A coarse filtering is first applied to narrow down the set of comments as suspect. Then comment expansion is adopted to improve the representativity of the feature by making convex combination of the original feature with those of "relevant" comments. Finally, a post-process is used upon some of the labels to make use of the label correlation for further improvement. Experiment results on the collected dataset shows statistical significant improvement in general against ILR as a baseline method.

8. REFERENCES

[1] C. M. Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[3] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] N. Chen, J. Lin, S. C. Hoi, X. Xiao, and B. Zhang. Ar-miner: Mining informative reviews for developers from mobile app marketplace. *International Conference on Software Engineering*, 2014.

[5] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh. Why people hate your app: making sense of user feedback in a mobile app store. In *Proceedings of the 19th ACM SIGKDD*, pages 1276–1284. ACM, 2013.

[6] L. V. Galvis Carreño and K. Winbladh. Analysis of user comments: an approach for software requirements evolution. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 582–591. IEEE Press, 2013.

[7] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, 2012.

[8] B. Quanz and J. Huan. Aligned graph classification with regularized logistic regression. In *SDM*, pages 353–364. SIAM, 2009.

[9] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine learning*, 85(3):333–359, 2011.

[10] H. T. T. Truong, E. Lagerspetz, P. Nurmi, A. J. Oliner, S. Tarkoma, N. Asokan, and S. Bhattacharya. The company you keep: Mobile malware infection rates and inexpensive risk indicators. 2013.

[11] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13, 2007.

[12] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '96, pages 4–11, New York, NY, USA, 1996. ACM.