Mobile App Recommendations with Security and Privacy Awareness

Hengshu Zhu[1], Hui Xiong[2], Yong Ge[3], Enhong Chen[1]

[1]University of Science and Technology of China,

[2]Rutgers-The State University of New Jersey,

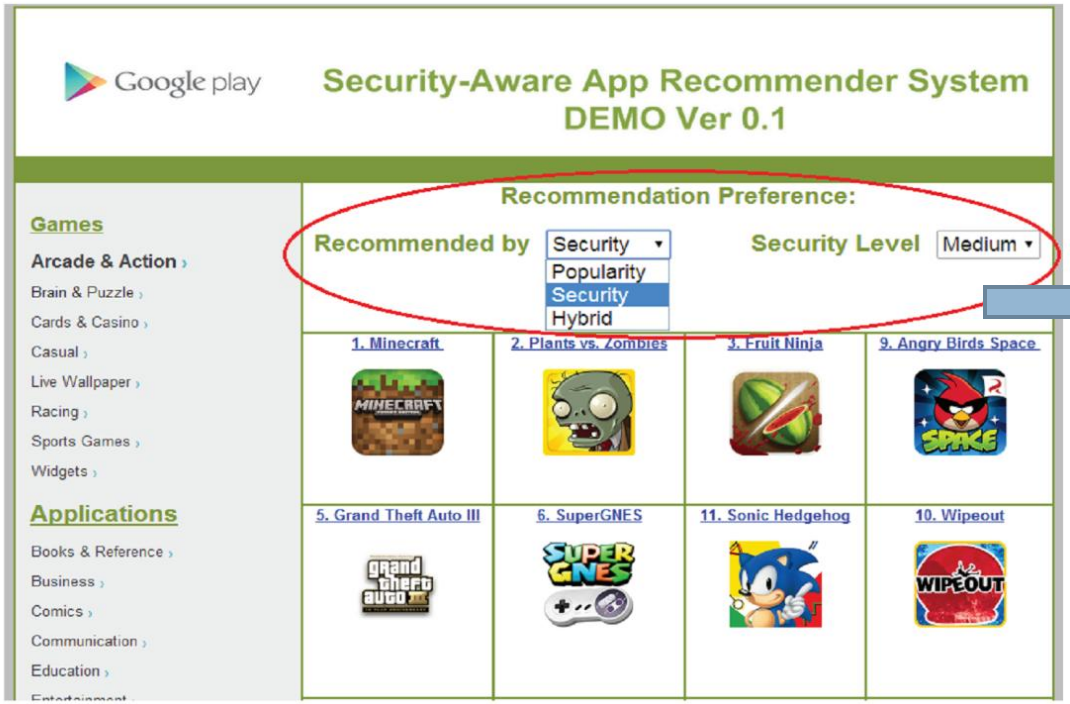
[3]UNC Charlotte

Background

- With the rapid prevalence of smart mobile devices, the number of mobile Apps available has exploded over the past few years.
  - 2M+ Apps, 120 billion+ cumulative downloads in Apple and Google' App stores.
- Benefit from the prospering mobile App industry, the functionalities of smart devices have been intensely extended to meet diversified user needs.
  - E.g., Location based Service, Social Network based Service.
  - Data Accessing Permissions: your locations, contact books, SMS etc.
- More and more mobile users are reluctant to adopt mobile Apps due to the risk of privacy invasion and other security concerns.

Motivation

- Building a **mobile App recommender system with security and privacy awareness** is critical for the healthy development of the mobile App industry.
- Existing mobile App recommender systems only consider user preferences about the Apps' popularity (e.g., ratings, downloads), but not the security and privacy risks inherent in the mobile Apps.



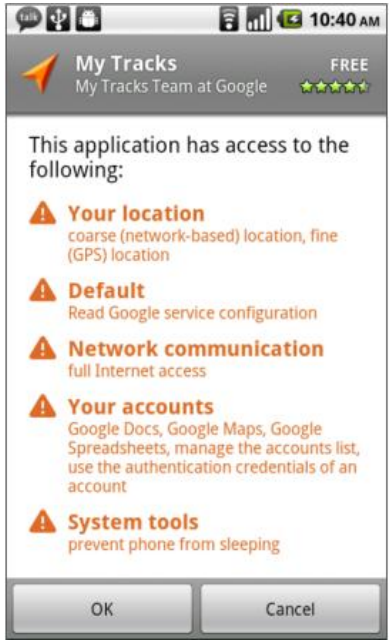
The design goal is to equip the recommender system with the functionality which allows to automatically detect and evaluate the security risk of mobile Apps.

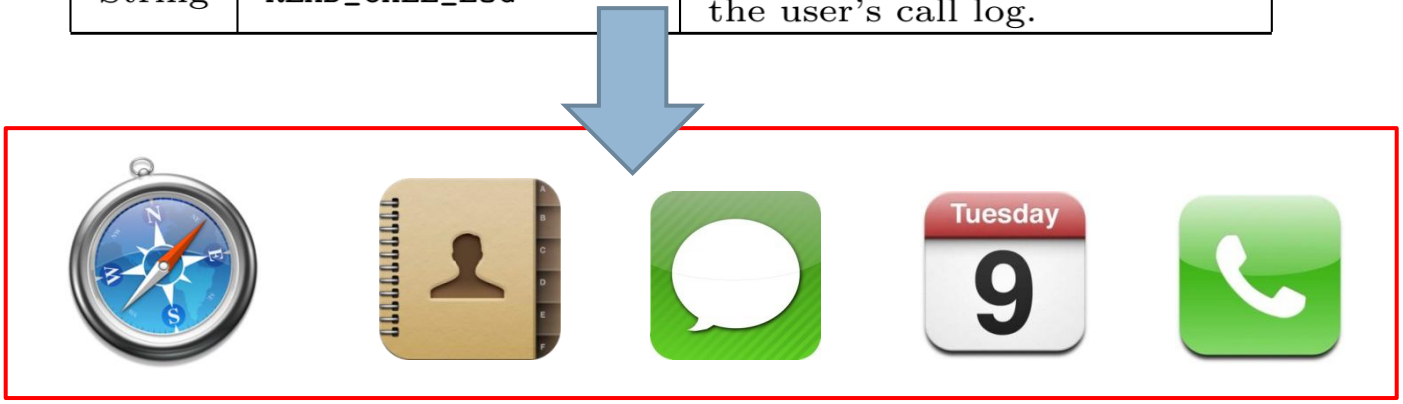
Preliminaries

- Indeed, the most advanced mobile operating systems, such as Apple IOS and Google Android, have different permission settings for 3rd party mobile Apps to visit users' personal data in their devices.

Table 1: Examples of data access permissions.

Type	Permission ID	Description
String	ACCESS_FINE_LOCATION	Allows an application to access fine (e.g., GPS) location.
String	READ_CONTACTS	Allows an application to read the user's contacts data.
String	READ_SMS	Allows an application to read the user's SMS messages.
String	READ_CALENDAR	Allows an application to read the user's calendar data.
String	READ_CALL_LOG	Allows an application to read the user's call log.

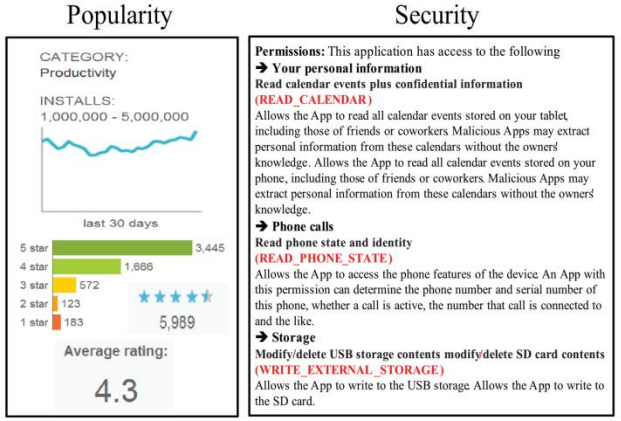




Problem Definition

DEFINITION 1 (PROBLEM STATEMENT). *Given a category label c, and a set of Apps $A = \{a\}$, each of which contains a set of data access permissions $\{p_i\}$, profile information (e.g., category, popularity), the goal of mobile App recommendation with security and privacy awareness is to build an optimal ranked list of Apps in category c based on both the Apps' popularity and users' security preferences.*

- How to mine the security risks of Apps and produce a ranked list $\Lambda^{(Risk)} = \{a|a \in c\}$ according to their **risk scores** $Risk(a)$, where a is ranked higher than a^* if and only if $Risk(a) > Risk(a^*)$.
- How to combine the risk based ranked list $\Lambda^{(Risk)}$ with the popularity based ranked list $\Lambda^{(Pop)}$ to produce final ranking so as to meet various expectations of users, who have different security and privacy concerns.



The Framework

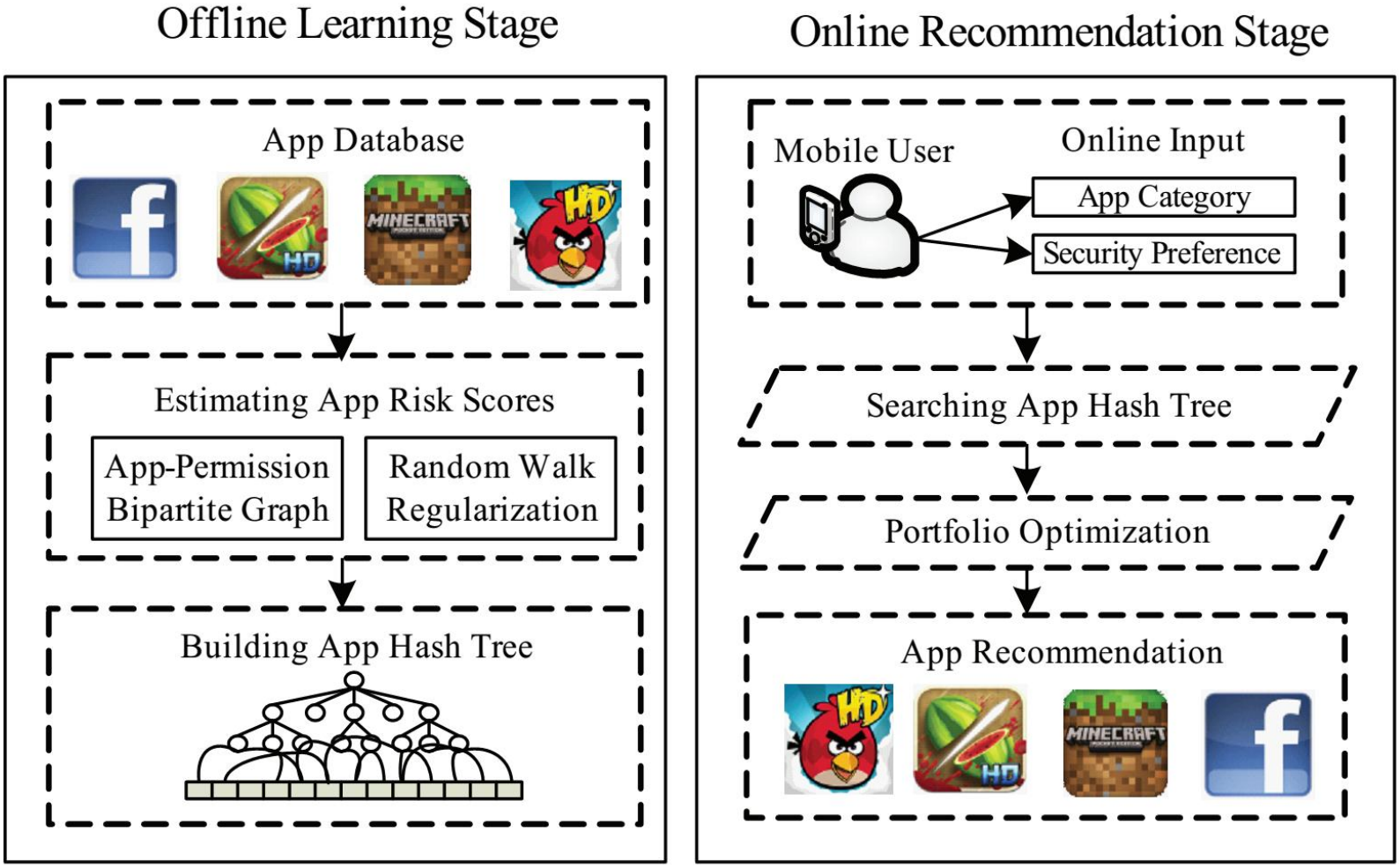
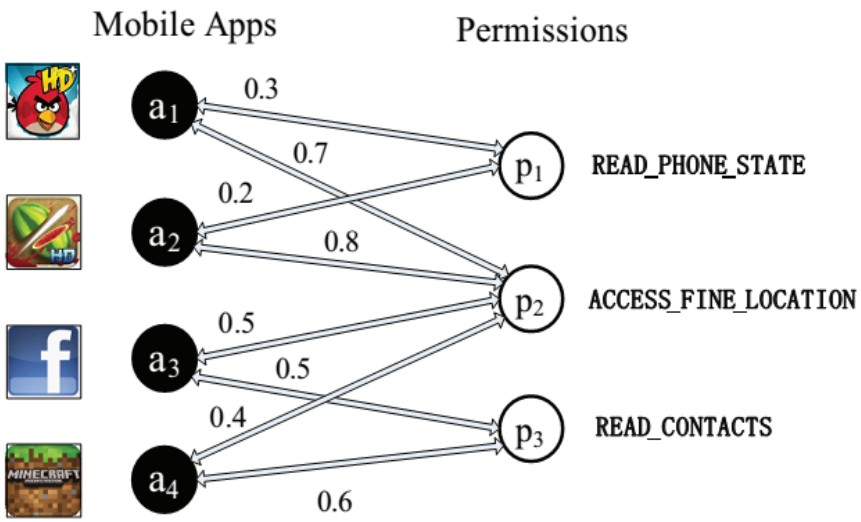


Figure 3: The recommendation framework.

Estimating Risk Scores for Mobile Apps

- Challenges:
  - It is hard to explicitly define a risk function with respect to different permissions for evaluating the potential risks of mobile Apps, since the permissions are often very ambiguous and poorly understood.
  - The latent relationships between Apps and permissions should be taken into consideration, since similar Apps (permissions) should have similar risk scores.
  - We should develop a scalable approach to refine risk scores, since rich external knowledge can be leveraged for evaluating potential risks of Apps.

A Novel Bipartite Graph based Estimation Approach.



$$w_{ij} = \frac{f_{ij}}{\sum_{p_j \in V^P} f_{ij}}.$$

$$\overrightarrow{a_i} = \{w_{i1}, \cdots, w_{iN}\}$$

$$\overrightarrow{p_j} = \{w_{1j}, \cdots, w_{Mj}\}$$

Estimating Risk Scores for Mobile Apps

The Optimization Framework:

1. The risk score and prior risk of Apps: $R_i^a \quad \widetilde{R}_i^a$
2. The risk score and prior risk of Permissions: $R_j^p \quad \widetilde{R}_j^p$

$$\mathcal{Q}(a,p) = \frac{\lambda}{2} \cdot \Big\{ \sum_i \left\| R_i^a - \widetilde{R}_i^a \right\|^2 + \sum_j \left\| R_j^p - \widetilde{R}_j^p \right\|^2 \Big\} +$$ ⇒ The prior consistency

$$\frac{\mu}{2} \cdot \Big\{ \sum_{i,j} s_{ij}^a \left\| R_i^a - R_j^a \right\|^2 + \sum_{i,j} s_{ij}^p \left\| R_i^p - R_j^p \right\|^2 \Big\} +$$ ⇒ The global consistency

$$\frac{1}{2} \cdot \sum_{i,j} w_{ij} \left\| R_i^a - R_j^p \right\|^2,$$ ⇒ The global Smoothness

$$s_{ij}^a = Cos(\overrightarrow{a_i}, \overrightarrow{a_j}) = \frac{\overrightarrow{a_i} \cdot \overrightarrow{a_j}}{\|\overrightarrow{a_i}\| \cdot \|\overrightarrow{a_j}\|}.$$

Prior Risks: PNB (*Peng et al. CCS-2012*)

Learn a generative model with parameter θ

$$\widetilde{R}_j^p = -\ln P(p_j|\theta)$$

$$\widetilde{R}_i^a = -\ln P(p_1, \cdots, p_k|\theta), \text{ where each } p_k \in a_i$$

Estimating Risk Scores for Mobile Apps

The Optimization Framework:

1. The risk score and prior risk of Apps: $R_i^a \quad \widetilde{R}_i^a$
2. The risk score and prior risk of Permissions: $R_j^p \quad \widetilde{R}_j^p$

Initialization:

$$R_i^a = 1/M \text{ and } R_i^p = 1/N$$

For each iteration:

$$\frac{\partial \mathcal{Q}}{\partial a_i} = \lambda(R_i^a - \widetilde{R}_i^a) + \mu \sum_j s_{ij}^a (R_i^a - R_j^a) + \sum_j w_{ij}(R_i^a - R_j^p),$$

$$R_i^a = \frac{\lambda \widetilde{R}_i^a + \mu \sum_j s_{ij}^a R_j^a + \sum_j w_{ij} R_j^p}{\lambda + \mu \sum_j s_{ij}^a + \sum_j w_{ij}}. \quad (4)$$

$$\frac{\partial \mathcal{Q}}{\partial p_j} = \lambda(R_j^p - \widetilde{R}_j^p) + \mu \sum_i s_{ij}^p (R_j^p - R_i^p) + \sum_i w_{ij}(R_j^p - R_i^a),$$

$$R_j^p = \frac{\lambda \widetilde{R}_j^p + \mu \sum_i s_{ij}^p R_i^p + \sum_i w_{ij} R_i^a}{\lambda + \mu \sum_i s_{ij}^p + \sum_i w_{ij}}. \quad (5)$$

Until converge

Estimating Risk Scores for Mobile Apps

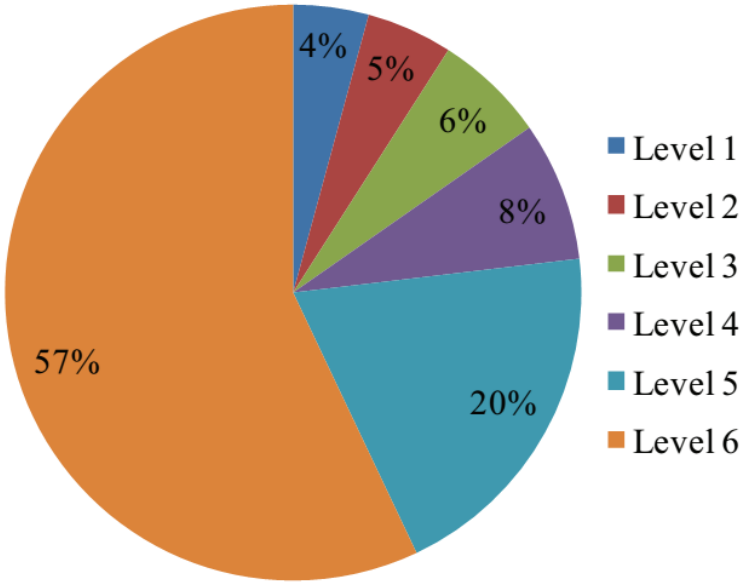
- For real-world App recommendation services, users may have difficulties to get clear perception about the risks of ranked Apps.
- A promising way to help users understand the different risks of Apps is to categorize the risks into discrete levels (e.g., Low, Medium, High).

Coefficient of Variation based Risk Segmentation

Algorithm 1 Automatic Detection of Security Levels

Input: The set of Apps $A = \{a_i\}$; Parameter δ;
Output: The set of security levels Ψ;

1: Rank A in descending order according to $Risk(a)$;
2: $L = \emptyset$;
3: **for each** $i \in [1, |A|]$ **do**
4: $A^* = L \cup \{A[i]\}$;
5: calculate $CV(A^*)$ in terms of $Risk(a)$ $(a \in A^*)$;
6: **if** $(CV(A^*) > \delta)$ **then**
7: $\Psi \cup = L$; $L = \emptyset$ is a new level;
8: **else**
9: $L \cup = \{A[i]\}$;
10: **end if**
11: **end for**
12: **return** Ψ



Ranking for Mobile App Recommendation

- Given a specific security level L* and a category c, we can treat all the Apps in category c with security L ≥ L*as candidates.
- Two basic Ranking Principles:

> - ***Security Principle***: We first rank App candidates in ascending order by their risk scores, and Apps have the same scores will be further ranked by popularity scores (e.g., overall rating).
> - ***Popularity Principle***: We first rank App candidates in descending order by their popularity scores (e.g., overall rating), and Apps have the same popularity scores will be further ranked by risk scores.

How to strike a balance?

Ranking for Mobile App Recommendation

- Portfolio based hybrid ranking principle:
  - Stock portfolio → App portfolio (App Candidates)
  - Future return → App popularity (e.g., Rating)
  - Future risk → App security risk

1. App portfolio: $\Upsilon = \{(a_i, w_i)\}, \quad s.t. \quad \sum_i w_i = 1.$

2. Popularity of a given App portfolio: $\mathbb{E}[\Upsilon] = \sum_i w_i \cdot \Delta_i^{-1},$

3. Risk of a given App portfolio:

$$\mathbb{R}[\Upsilon] = \sum_i \sum_j risk(a_i, a_j) w_i w_j, \quad risk(a_i, a_j) = \nabla_i^{-1} \nabla_j^{-1} J_{ij}.$$

$$\mathbb{R}[\Upsilon] = \sum_i^n (w_i^2 \nabla_i^{-2} + 2 \sum_{j=i+1}^n w_i w_j \nabla_i^{-1} \nabla_j^{-1} J_{ij}),$$

Ranking for Mobile App Recommendation

- **Objective:** Maximize the popularity and minimize the security risk of App portfolio.

$$\arg\max_{\mathbf{w}} \quad \mathbb{E}[\Upsilon] - b \cdot \mathbb{R}[\Upsilon],$$

$$s.t. \quad \sum_i w_i = 1.$$

$$\mathbf{w}^* = \frac{\begin{vmatrix} 1 & \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{E} \\ E^* & \mathbf{E}^T\boldsymbol{\Sigma}^{-1}\mathbf{E} \end{vmatrix} \boldsymbol{\Sigma}^{-1}\mathbf{1} + \begin{vmatrix} \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{1} & 1 \\ \mathbf{E}^T\boldsymbol{\Sigma}^{-1}\mathbf{1} & E^* \end{vmatrix} \boldsymbol{\Sigma}^{-1}\mathbf{E}}{\begin{vmatrix} \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{1} & \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{E} \\ \mathbf{E}^T\boldsymbol{\Sigma}^{-1}\mathbf{1} & \mathbf{E}^T\boldsymbol{\Sigma}^{-1}\mathbf{E} \end{vmatrix}}, \quad (12)$$

where $\boldsymbol{\Sigma}_{ij} = \nabla_i^{-1}\nabla_j^{-1}J_{ij}$, $\mathbf{E} = (\Delta_1^{-1}, \cdots, \Delta_n^{-1})^T$, and E^* can be computed by

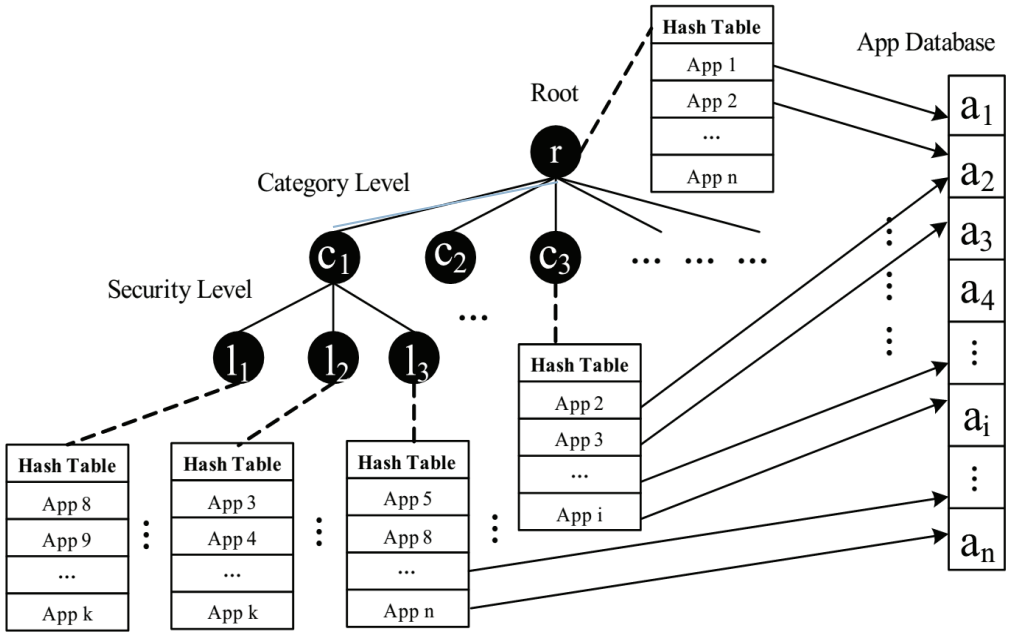
$$E^* = \frac{(xz - y^2)^2 - 2b(x\mathbf{E} - y\mathbf{1})^T\boldsymbol{\Sigma}^{-1}(z\mathbf{1} - y\mathbf{E})}{2b(x\mathbf{E} - y\mathbf{1})^T\boldsymbol{\Sigma}^{-1}(x\mathbf{E} - y\mathbf{1})}, \quad (13)$$

where $x = \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{1}$, $y = \mathbf{1}^T\boldsymbol{\Sigma}^{-1}\mathbf{E}$, and $z = \mathbf{E}^T\boldsymbol{\Sigma}^{-1}\mathbf{E}$.

Efficient frontier based solution (Zhang et al. RecSys 2013)

Ranking for Mobile App Recommendation

- In an online App recommender system, it is necessary to quickly response users' requests and efficiently manage Apps in its back-end servers.



App Hash Tree

1. The category level is used for traditional popularity based App recommendation.
2. The security level is used for our novel security-aware App recommendation.
3. All ranked lists can be pre-stored in the nodes of the tree.

Experiments

- The experimental data were collected from Google Play in 2012. This real-world data set includes **170,753** Apps in 30 App categories, and the Apps have 173 unique data access permissions.
- Particularly, the data set includes more than **25%** Apps available at the Android Market, which totally includes **675,000** Apps as of the end of September 2012.

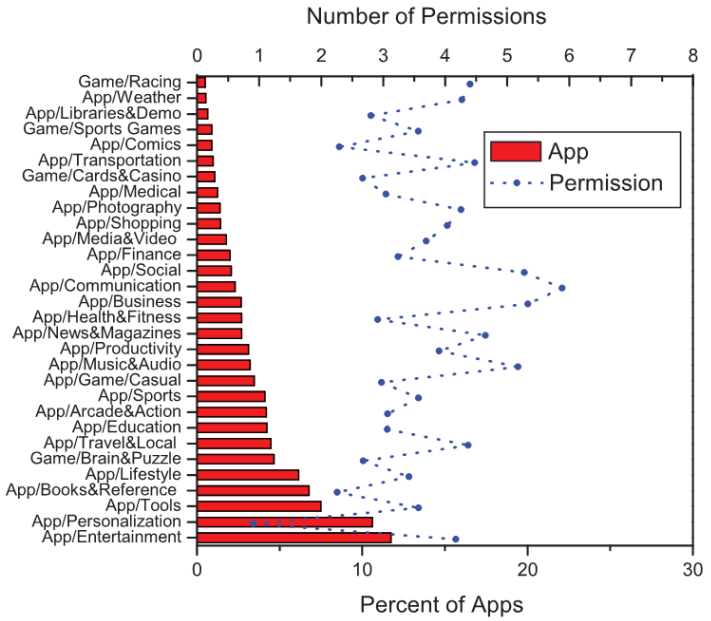


Figure 6: The percent of Apps and the average number of requested permissions by each App in different categories.

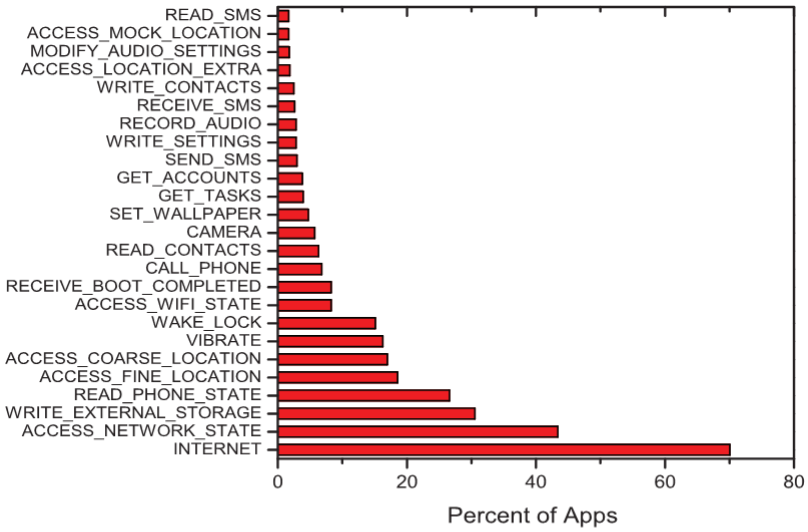


Figure 7: The top 25 most used permissions in our data set and the percent of Apps that request those permissions.

Experiments

Evaluation of Ranking App Risks

- **Three senior Android users for labeling:** Insecure (2), Not Sure (1), Secure (0)
- **Baseline 1:** Naive Bayes with information Priors (PNB)
- **Baseline 2:** RankSVM
- **Metrics:** NDCG@K, Precision@K, Recall@K, F@K

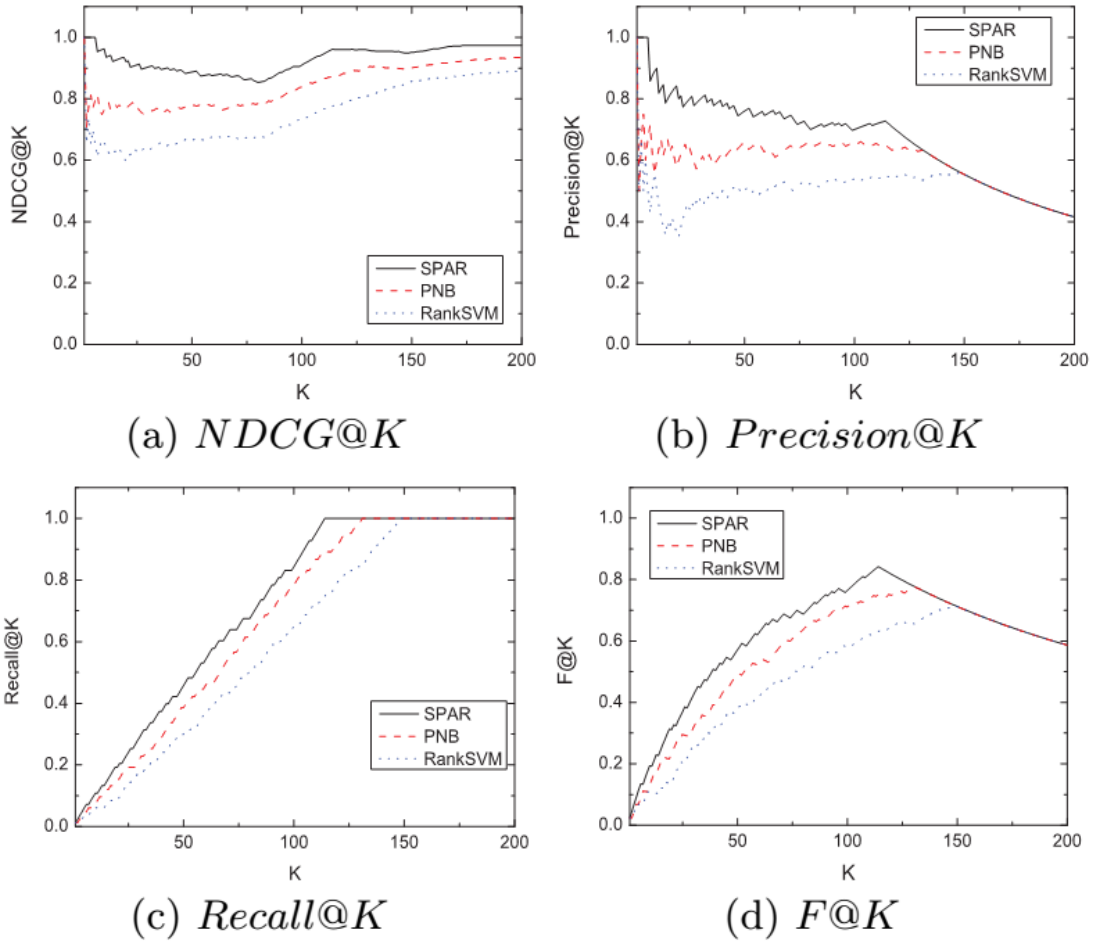
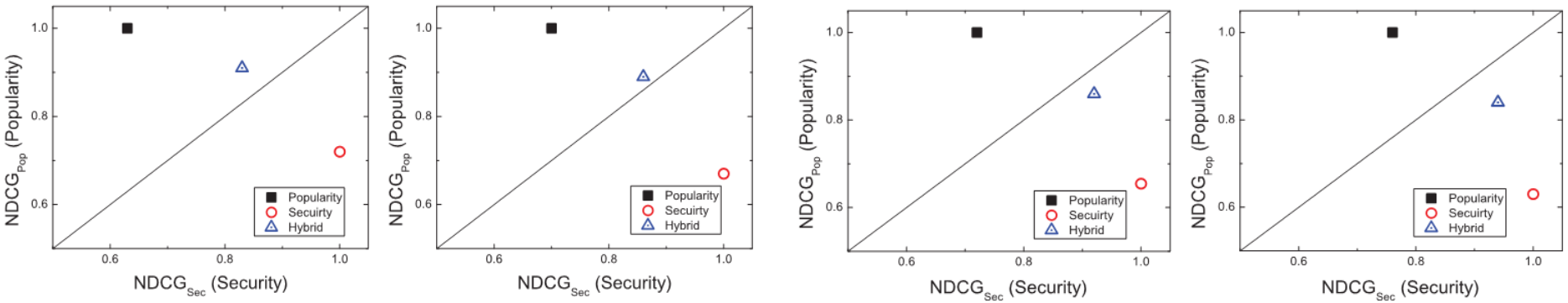


Figure 10: The performance of each approach w.r.t different metrics based on user judgment.

Experiments

Evaluation of App Recommendation



(a) Level 1 (b) Level 3 (c) Level 5 (d) Level 6

Figure 11: The recommendation performances of different ranking principles.

Table 3: The case study of App recommendation.

	Recommendation
SEC	SimplyNoise,Moment Diary,Bedside,BeNaughty,Weterago
POP	Weterago,Bedside,Moment Diary,BeNaughty,SimplyNoise
H-1	Bedside,Moment Diary,Weterago,BeNaughty,SimplyNoise
H-3	Moment Diary,Bedside,BeNaughty,SimplyNoise
H-5	Moment Diary,SimplyNoise,Beside
H-6	SimplyNoise,Moment Diary

Conclusions

- We developed a mobile App recommender system with security and privacy awareness.
- We designed a scalable and automatic approach for estimating the security risks of Mobile Apps.
- We introduced a flexible App recommendation method based on the modern portfolio theory.
- We conducted extensive experiments on a large-scale real-world data set, which clearly validated the effectiveness of the proposed recommendation framework.

Q&A

Thank You!

zhuhengshu@gmail.com