Mobile App Privacy and Security Recommendation Systems

Dillon V Stenberg

Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267

December 5, 2015

- Over 1.5 million Apps (2015) and Over 50 billion downloads (2013)
- Each App on Android devices have data permissions
- Both Google Play and Apple have their own recommendation method
- Two recommendation methods
  - SPAR
  - Privacy-Respect App

Figure: Android Smartphone

Source: gadget-report.eu

What are some potential risks?

- Downloading an App allows access to your device and information
- Apps can access information based on their data permissions
- Apps can send and store your private information
- Survey Says! (International Data Group)
  - 54% wouldn't download an App
  - 30% removed an App

Outline

1. Background
2. SPAR
3. Privacy-Respect App Recommendation Model
4. Conclusion

1. Background
   - Data Permissions
   - Latent Matrix Factorization
2. SPAR
3. Privacy-Respect App Recommendation Model
4. Conclusion

Data Permissions

Android OS has a data permission framework

- What is a data permission?
- `READ_CONTACTS`, `ACCESS_FINE_LOCATION`

What makes a data permission dangerous?

- Depends on the user
- Depends on how the permission is used
- Some consider these to be dangerous
    - `READ_SMS`
    - `READ_CALL_LOG`

Matrix Factorization

- What is a latent matrix factorization model?
- Latent Variables: a variable that is not directly observed (directly measured), but are inferred from other variables that are observed

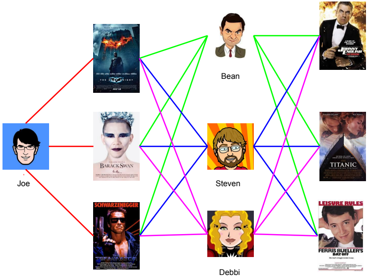


Figure: Example of recommending movies to users

Security and Privacy aware mobile App Recommendation (SPAR)

- Estimates risk scores for each App
- Ranks each App by risk score and popularity
  - Modern Portfolio Theory approach used to balance popularity and user privacy preferences
- Create an *App Hash Tree* to efficiently recommend Apps

Evaluating Risk Scores

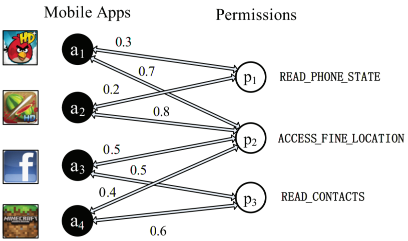
- Risk score is used to reflect the security level
- The lower the score, the safer it is to use the App

Challenges

- Difficult to assign risk scores based on permissions because some are ambiguous and are understood poorly
- Considering latent relationship between Apps and permissions
- Develop a scalable approach to refine the risk scores: Bipartite Graph

Bipartite Graph

- The bipartite graph is used to build the connections between Apps and their permissions
- Learns the security risk of each App by learning the probability of an App requesting a permission



Assigning Risk Score to Apps

- A probabilistic approach called Naive Bayes with information Priors (PNB) is used for assigning the risk scores to each App
- A technique to construct classifiers: models that assign class labels to problem instances

Naive Bayes with information Priors

- *Bayes theorem*: describes the probability of an event based on conditions related to that event
- The Beta Distribution is used as an information prior to describe probability

Coefficient of Variance (CV)

- Algorithm constructed to appropriately divide the Apps in their security levels
- Risk scores of each App compared to parameter δ
  - If the risk score is greater than δ, then that App is placed in a new security level
  - If not, then the App is put into the appropriate security level

App Hash Tree

- *Hash Tree*: a tree that has labeled nodes of values that have child nodes in different hierarchies
- Two hierarchies: Security and Category
- Hash tree is used for efficiently recommending Apps to users

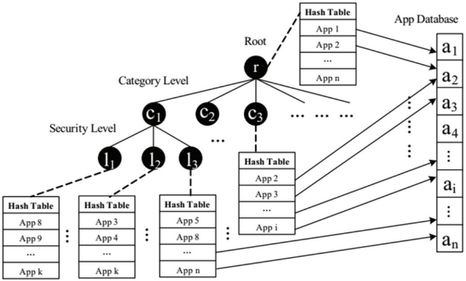
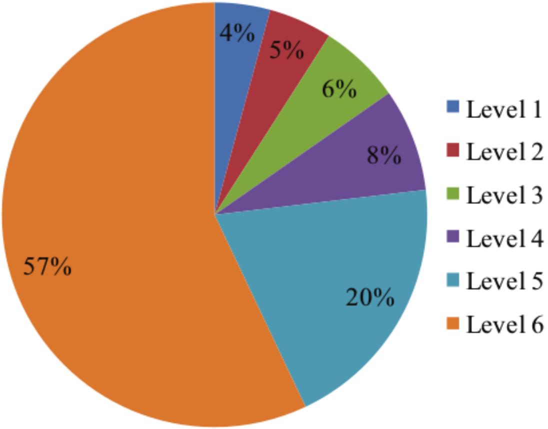


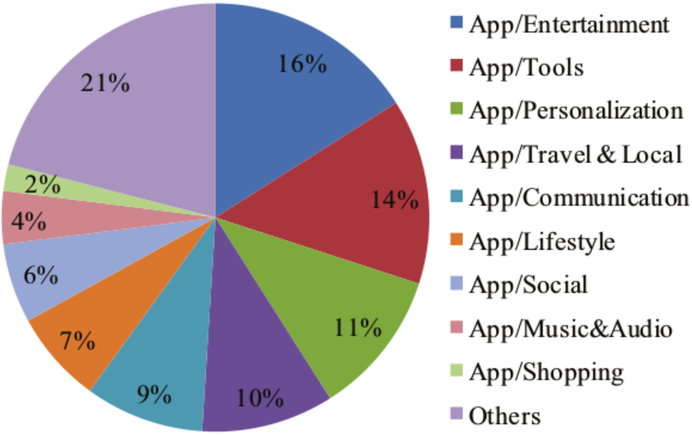
Figure: Example of the App hash tree

Data Collected

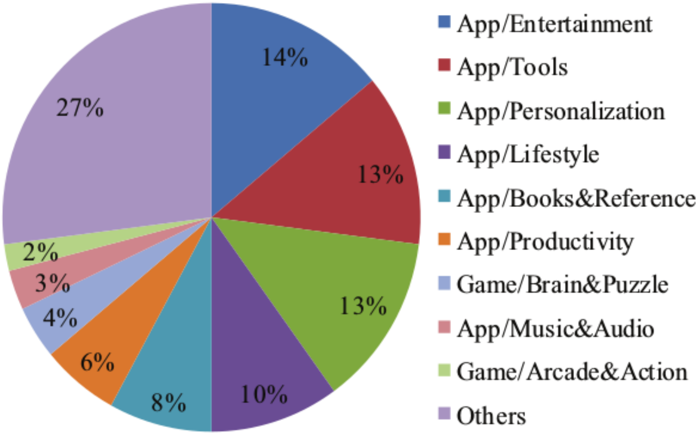
- 170,753 Apps in 30 different categories, with 173 unique security permissions
    - Took top 100 and bottom 100 ranked Apps from SPAR
- Zhu et el. manually labeled 200 secure Apps and 200 insecure Apps
- Merged Apps into a pool which includes 496 unique Apps
- Had each App judged by at least 3 users
    - Gave each App a score of 2 (insecure), 1 (not sure), or 0 (secure)
    - Gave label based on the App profile, their own experience, and other users



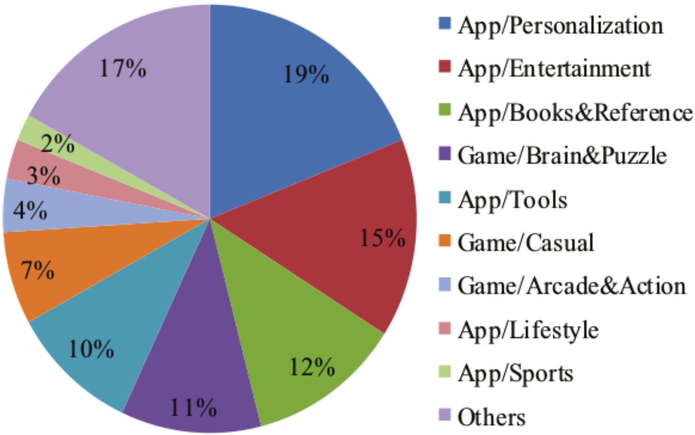
(a) All Levels


16%
14%
11%
10%
9%
7%
6%
4%
2%
21%
App/Entertainment
App/Tools
App/Personalization
App/Travel & Local
App/Communication
App/Lifestyle
App/Social
App/Music&Audio
App/Shopping
Others

(b) Level 1


14%
13%
13%
10%
8%
6%
4%
3%
2%
27%
App/Entertainment
App/Tools
App/Personalization
App/Lifestyle
App/Books&Reference
App/Productivity
Game/Brain&Puzzle
App/Music&Audio
Game/Arcade&Action
Others

(c) Level 3


19%
15%
12%
11%
10%
7%
4%
3%
2%
17%
App/Personalization
App/Entertainment
App/Books&Reference
Game/Brain&Puzzle
App/Tools
Game/Casual
Game/Arcade&Action
App/Lifestyle
App/Sports
Others

(d) Level 6

Three Methods Tested

- SPAR
- PNB
- RankSVM: a learning-to-rank approach that analyzes data and recognizes patterns by the relationship of a specific query

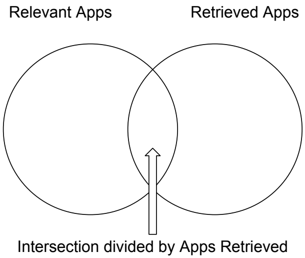
Four Evaluation Metrics

- Normalized Discounted Cumulative Gain (NDCG)
- $\text{Precision@K} = \frac{|\{\text{Relevant Apps} \cap \text{Retrieved Apps}\}|}{|\{\text{Retrieved Apps}\}|}$
- $\text{Recall@K} = \frac{|\{\text{Relevant Apps} \cap \text{Retrieved Apps}\}|}{|\{\text{Relevant Apps}\}|}$
- $F@K$ is the balance of precision and recall.

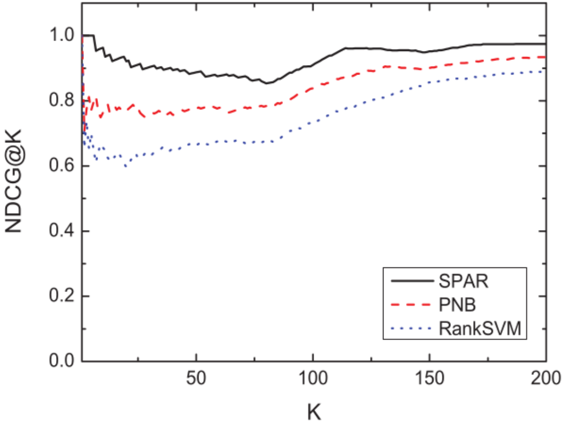
NDCG

- $\mathrm{NDCG}_p = \frac{\mathrm{DCG}_p}{\mathrm{IDCG}_p}$
- $\mathrm{DCG}_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$
- Highly relevant Apps appearing lower in a search result list should be penalized as the graded relevance value is reduced logarithmically proportional to the position of the result
- Being normalized means ordering the relevance ranks of the users most relevant to not relevant
- IDCG is the max sum of normalized DCG ordering

Venn Diagram of Precision@K

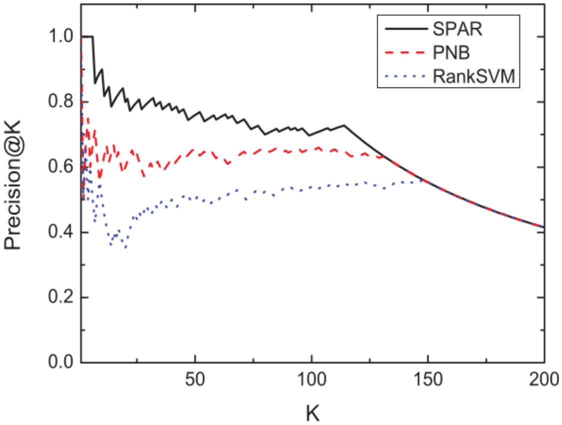


Overall Performance



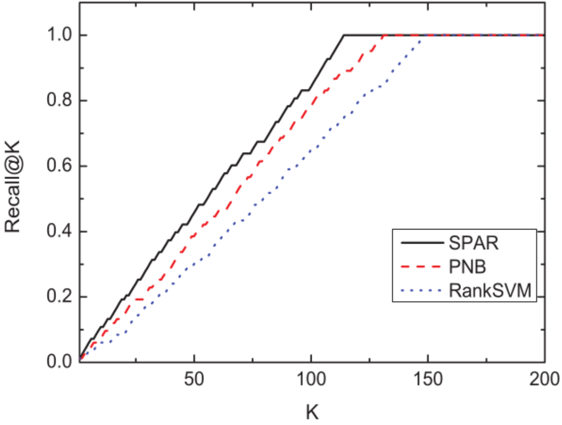
(a) $NDCG@K$

Overall Performance



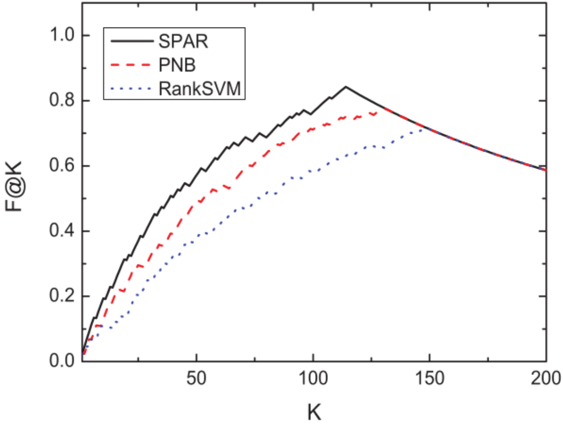
(b) $Precision@K$

Overall Performance



(c) $Recall@K$

Overall Performance



(d) $F@K$

Overview

- Construct a new latent factorization model to capture the trade-off between App functionality and user privacy preference
- Define a hierarchy of three levels of privacy used to characterize users' privacy preferences
- A crawler is created to crawl through a real world dataset

Privacy Levels

- Level I: 10 resources
- Level II: 10 resources from Level I and 23 security permissions
- Level III: Resources and permissions from Level II as well as resources Internet and Bluetooth. 72 total security permissions

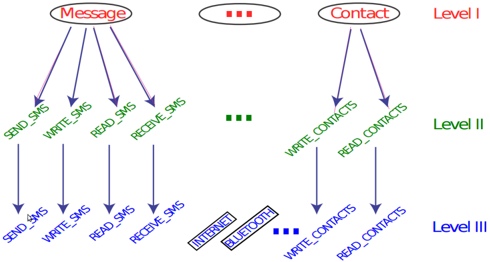


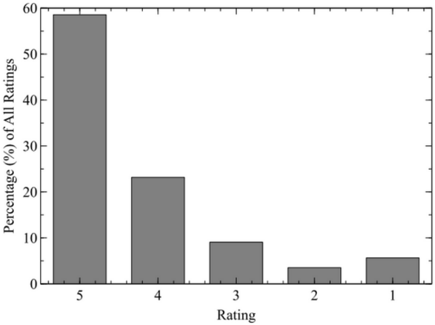
Figure: Illustration of the three levels of privacy

Constructing a Latent Factorization Model

- Modeling functionality match: Model based on standard latent factorization from other models (SVD)
- Modeling privacy respect: Modeled based on the privacy levels, describes user privacy preference and App's privacy information
- Trade-off between privacy and functionality: User's overall preference

User Profile Latent Factor

- Combine the privacy preference and user interest vectors into one vector
  - Combining them can reduce parameters to learn and will improve computational efficiency
- Poisson Distribution used to help model User Profile by user ratings data
  - It can better rank the preference order



Crawling Through the Dataset

- Obtained Google ID of a user
- Crawler is created to retrieve all the Apps that a user rated
- 16,344 users, 6,157 Apps, and 263,054 rating observations

Three Privacy Variants

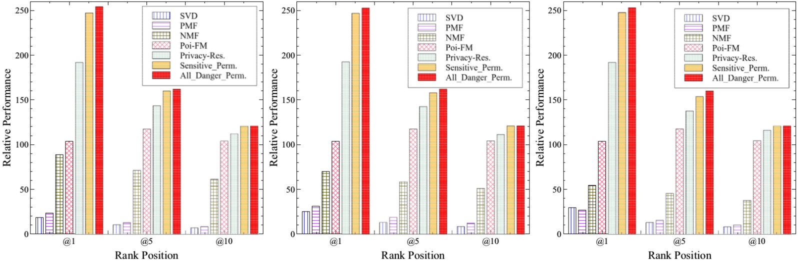
- `Privacy_Res`: Privacy-respect App recommendation with Level I privacy level.
- `Sensitive_Perm`: Privacy-respect App recommendation with Level II privacy level.
- `All_Danger_Perm`: Privacy-respect App recommendation with Level III privacy level.

Relative Performance

- A user is presented with a list of recommendations of top-N ranked Apps that have the highest predicted values
- Evaluate each approach on the Apps that were adopted by the user
- rPrecision@N $= \frac{\text{Precision@}N}{|C_{\text{adopted}}|/|C|}$
- rRecall@N $=$ rPrecision@N: Called Relative Performance
- C denotes the candidate Apps

Overall Performance

K is the latent dimension or cut-off of recommended Apps



(a) $K = 20$ (b) $K = 30$ (c) $K = 50$

Figure: Relatvie performance @N with different latent dimensions K

Conclusions

- Results show that both recommendation methods perform better over previous methods
    - SPAR was shown to be more effective and efficient
    - Privacy-Respect App privacy variants were shown to outperform other methods
- Implying that considering user privacy preference on personalized App recommendations is important

Acknowledgements

I would like to thank Kristin Lamberty, Nic McPhee, Elena Machasova, Peter Dolan, and everyone that had taken the time to provide me with feedback.

References

Liu, Bin and Kong, Deguang and Cen, Lei and Gong, Neil Zhenqiang and Jin, Hongxia and Xiong, Hui. *Personalized Mobile App Recommendation: Reconciling App Functionality and User Privacy Preference*. 2015.

Zhu, Hengshu and Xiong, Hui and Ge, Yong and Chen, Enhong. *Mobile App Recommendations with Security and Privacy Awareness*. 2014.

Thank you for your time

Contact Info

- stenb061@morris.umn.edu

Questions?