# Recommending Functions in Spreadsheets from the Fuse Corpus

Shaown Sarker, Matthew Neal, Nisarg Vinchhi

# Why Spreadsheets?

- End-user programmers[1]
  - People who are not professional software developers
  - Make use of tools and processes that lets them perform tasks similar to programming
- Spreadsheet users are the largest demographic within the end-user programmers.
  - According to a study in 2005[2], nearly 23 million Americans use spreadsheets (31% of the total workforce).

# Why Recommend Functions?

- There are more than 350 distinct functions in spreadsheets[3].
- But only a handful are user most frequently.
- We want to make personalized function recommendations for users for ‘functional awareness’
    - Helps accomplishing new task.
    - Improves the performance of known task.

# Inspiration

- Recommending commands in large and complex software systems
  - CommunityCommands in AutoCAD[4]
  - Improving developer fluency in Eclipse[5]
- Both utilizes the collaborative filtering based algorithms to recommend personalized commands.

# Project Goals

- Recommend functions from Fuse by applying user-based and item-based collaborative filtering.
- Our work involves
  - Feature extraction
  - Applying the algorithms
  - Cross validation
- Baseline:
  - Most popular algorithm[6]

# Methodology

Feature Extraction:

- 7000 distinct user vectors from 250K spreadsheets
  - Distinct by: created by, last modified by, domain name
- Features in the vectors:
  - Function use count (eg number of times SUM used)


```
('Greenfield, Laura#Greenfield, Laura#www.cde.state.co.us', {'SUM': 8, 'Plus': 179, 'Divide': 179})
```

# Item Based Collaborative Filtering

- Uses same algorithm used by “CommunityCommands”[5]
  - Recommends commands in Autocad.
- Generates a similarity matrix from the user vectors
  - Similarity Function: cosine similarity
- Recommendations (variable number):
  - Functions not used by the input user
  - List generated from matrix, sorted by mean similarity score.
- Tuning parameter:
  - Number of recommendations returned

# User Based Collaborative Filtering

- Calculates weighted vectors using command frequency inverse user frequency (cf-iuf) from user vectors.
- Finds most similar users based on cosine distance from the input user
- Recommendations:
  - List of functions used by similar users but not by the input user
  - Sorted by expected frequency in the similar vectors
- Tuning parameters:
  - Number of similar users
  - Number of recommendations returned

# Baseline

- Most Popular Algorithm[6]:
  - Create a global ordered list of most used functions for all training vectors
  - Recommend functions based on the top functions not used by the input user
  - Used by Owl Recommendation System in MS Word.

# Cross Validation Strategy

- Used 14 Fold Cross Validation
- For each test subject we randomly remove one of their functions
- Then we generate a set of recommendations. If one of the recommended functions is the removed function, count a success
- Our ratio is calculated across all 14 folds
- Item Based method: Tune no. of recs - 1, 3, 5, 10
- User Based method: (a) Tune no. of recs - 1, 3, 5, 10. (b) Tune no of similar users - 10, 20
    a. Below 10 similar users tend to not fill 10 recommendations
- Baseline: Tune  number of recommendations: 1, 3, 5, 10

# Results

<insert graph of results>

# Discussion

# Future Work

- Limitations of existing dataset
  - Function usage rate and diversity
  - Lack of temporal or sequential information
  - Method of evaluation cannot replace real life spreadsheet users

- Future Work
  - Incorporate more datasets to increase diversity, like Enron[7] and Euses[8]
  - Applying function discovery sequence information in in algorithms[5]
  - Possibility to conduct a study

# References


1. Ko, Andrew J., et al. "The state of the art in end-user software engineering."*ACM Computing Surveys (CSUR)* 43.3 (2011): 21.
2. Scaffidi, Christopher, Mary Shaw, and Brad Myers. "Estimating the numbers of end users and end user programmers." *Visual Languages and Human-Centric Computing, 2005 IEEE Symposium on*. IEEE, 2005.
3. https://support.office.com/en-us/article/Excel-functions-alphabetical-b3944572-255d-4efb-bb96-c6d90033e188
4. Matejka, Justin, et al. "CommunityCommands: command recommendations for software applications." Proceedings of the 22nd annual ACM symposium on User interface software and technology. ACM, 2009.
5. Murphy-Hill, Emerson, Rahul Jiresal, and Gail C. Murphy. "Improving software developers' fluency by recommending development environment commands." Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering. ACM, 2012.
6. Linton, Frank, et al. "OWL: A recommender system for organization-wide learning." *Educational Technology & Society* 3.1 (2000): 62-76.
7. Felienne Hermans and Emerson Murphy-Hill. Enrons spreadsheets and related emails: A dataset and analysis. Technical report, Delft University of Technology, Software Engineering Research Group, 2014.
8. Marc Fisher and Gregg Rothermel. The euses spreadsheet corpus: a shared resource for supporting experimentation with spreadsheet dependability mechanisms. In ACM SIGSOFT Software Engineering Notes, volume 30, pages 1–5. ACM, 2005.