Improving Relevance in a Recommendation System to Suggest Charities without Explicit User Profiles Using Dual-Autoencoders

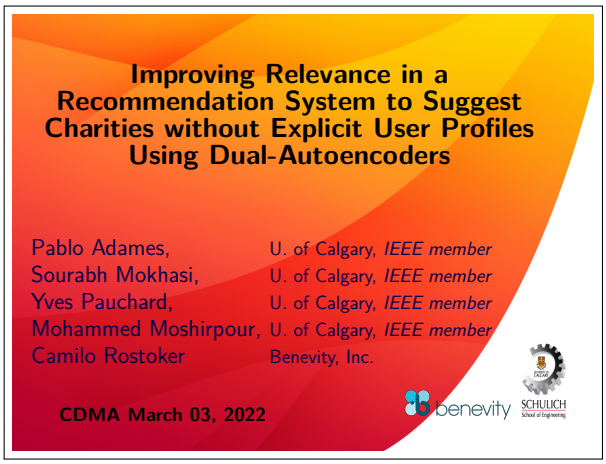
Pablo Adames, Sourabh Mokhasi,
Yves Pauchard, Mohammed Monshipour,
Camilo Rostoker

March 03, 2022

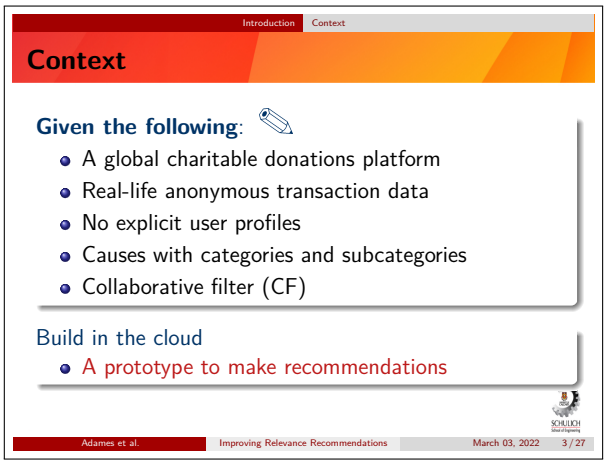
1 Introduction

This presentation was prepared for the paper of the same title accepted for publication at the 7th International Conference on Data Science and Machine Learning Applications (CDMA 2022) held at the Prince Sultan University, Riyadh, Saudi Arabia, from March 1st to 3rd, 2022.

1.1 Context



(a) ***Presentation title and authors slide***



(b) ***Context for this work***

Figure 1: Title and context

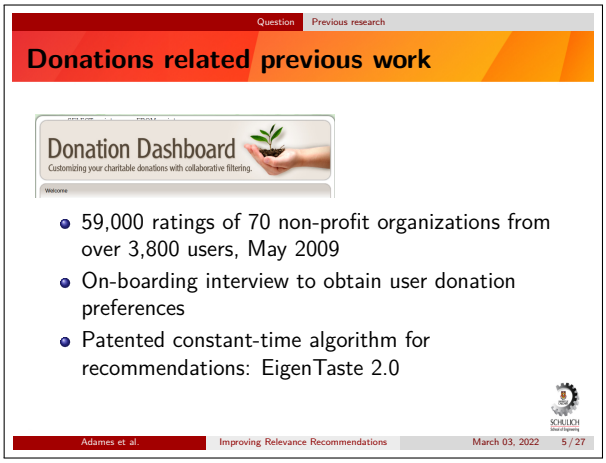
A recommendation system is based on an item-user matrix containing ratings given by users to the items. Furthermore the users and the items have explicit profiles that are used to find similarity of them as needed. Figure 1b mentions the limitation of the data used for this system, because in order

to generate the user-cause matrix, one has a set of anonymous transactions without explicit user profiles, no ratings, and two levels of classification for the charitable causes.

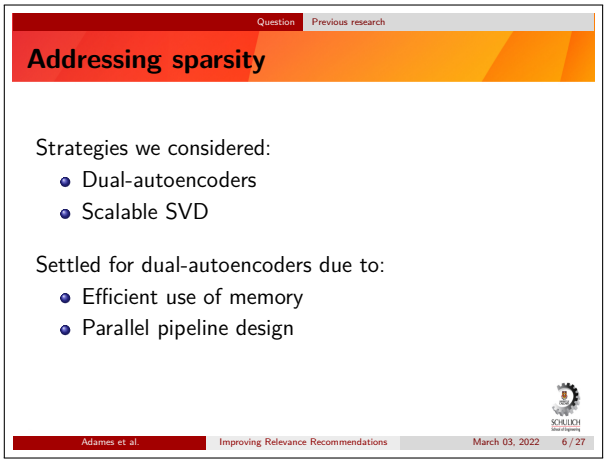
2 Question

Figure 3b highlights the main research question given the constraints discussed in other slides included in this section below.

2.1 Previous research



(a) ***Previous recommender for donations.***



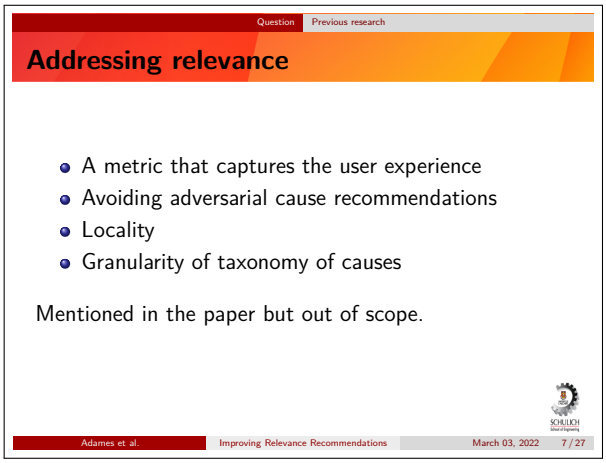
(b) ***Strategies to address rating sparsity***

Figure 2: Previous work and computational strategies

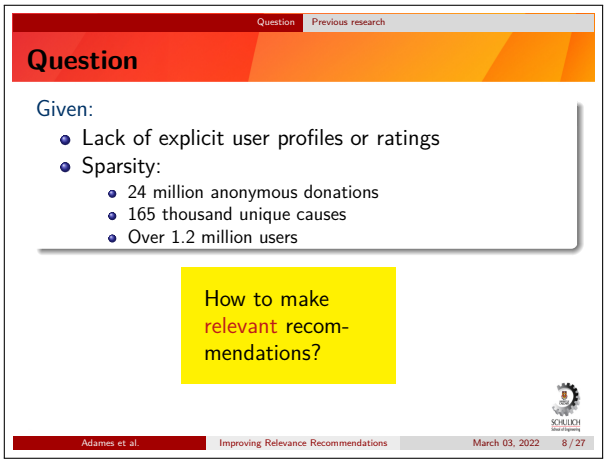
A web portal called Donations Dashboard was created in 2009 as an application of a patented, general-purpose, scalable recommendation algorithm called EigenTaste, figure 2a. This was part of a research project at Stanford University. In this application, users were on-boarded via a survey to generate their preference for certain types of causes along several classification criteria. This marked a strong difference with the current work where there are no explicit user preferences for the hundreds of thousands of users of the platform.

Another observation was the great sparsity in the user-cause matrix due to the great number of causes available for users. The computational strategies offered by a managed machine learning environment available for this research were dual-autoencoders and scalable SVD, figure 2b. The first option was chosen due to efficient memory use and the ability to parallelize the training components in two independent streams. No further attempt at comparing the strategies was done in order to stay within the scope of this research.

The questions around how to measure the user experience, how to craft formulations that avoid adversarial cause recommendations, address locality, or are impacted by the granularity fo the classification of causes were mentioned but not addressed in this paper as they would have also gone beyond the original scope, figure 3a.



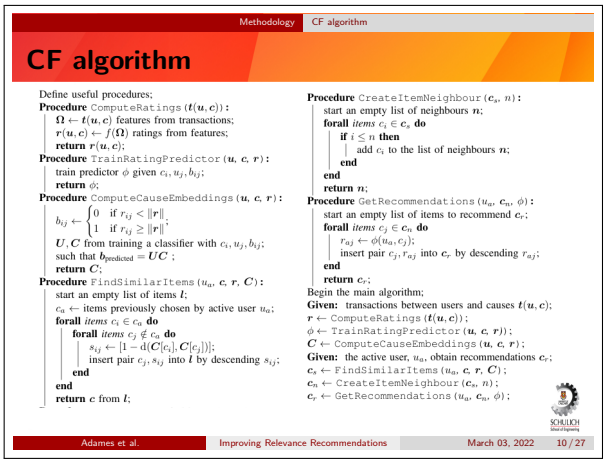
(a) ***How to measure relevance***



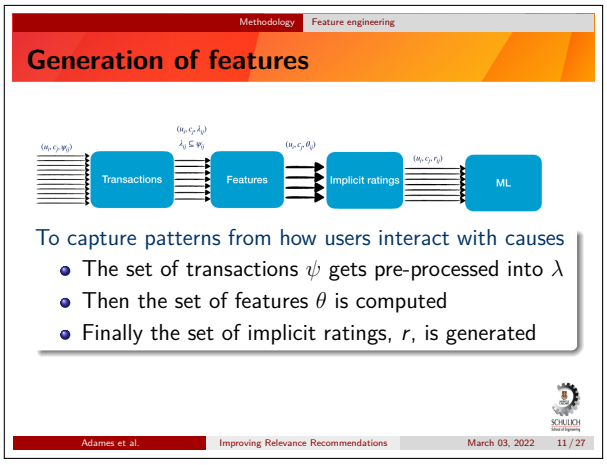
(b) ***How to make relevant recommendations?***

Figure 3: Capturing relevance of recommendations and research goal

3 Methodology



(a) ***Algorithm***



(b) ***Features***

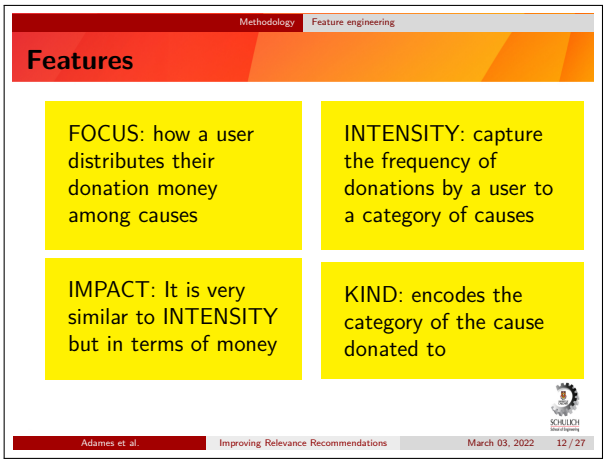
Figure 4: Computing recommendations and using domain knowledge for features

3.1 CF algorithm

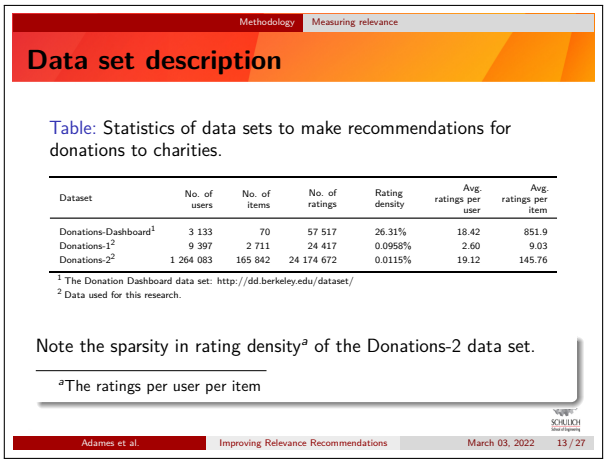
The most popular algorithm to create recommendation systems is the collaborative filter, CF. Figure 4a shows the actual algorithm used for this paper to compute recommendations given a user and transactions, although it did not make it to the paper due to limitations of space. In practical terms the classifier and regressor are generated before recommendation lists can be computed. These machine learning components can be trained in two independent parallel pipelines using dual-autoencoders. The cause embeddings produced by the classifier are used to compute cause similarity while the regressor is employed to predict ratings for unseen causes to the target users.

3.2 Feature engineering

A series of transformations are necessary to go from donation transaction records to implicit ratings. A subset of transactions are selected to represent useful intermediate aggregate values that get further transformed into features. These are the elements used to compute implicit ratings, figure 4b. The triplets made out of user, cause, and implicit rating become the training data for the machine learning components of the system.



(a) ***Definitions***



(b) ***Data set statistics***

Figure 5: Feature definitions and data set description.

Figure 5a shows high level descriptions of four features used to determine affinity for user to charitable donations. The paper contains detailed descriptions and mathematical formulas used to compute them with computer code. The implicit ratings from a user to a cause are computed as linear combinations of these features.

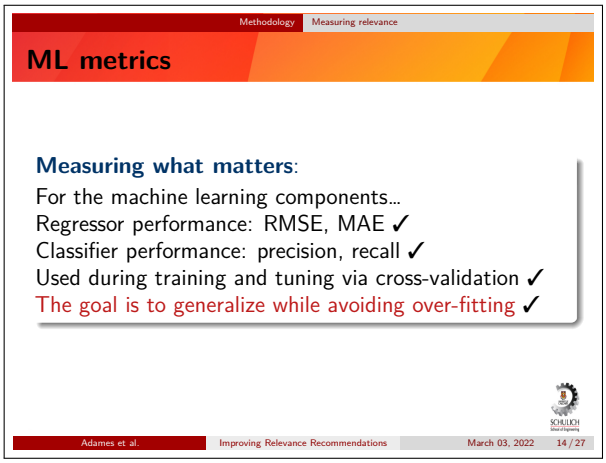
3.3 Measuring relevance

Figure 3a highlights the low rating density of the data set used for this research: *Donations-2*, compared with the smaller exploratory data set *Donations-1* and the one used for the Donations Dashboard system. This is a challenge to produce relevant recommendations because even the users with high donation activity tend to concentrate on a few causes from the large diversity of them available in the platform.

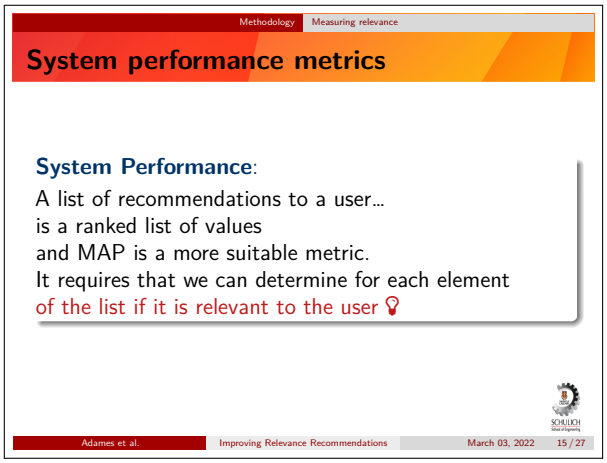
It is a normal practice to measure how well a machine learning model performs while training it compared to a subset of the training data or against an out of sample test data set. Figure 6a illustrates the metrics most commonly used for these tasks.

The performance of a recommendation system cannot be measured only by the performance of the individual classifier and regressor used to calculate recommendation lists within the CF algorithm. The main goal while training machine learning models is to make it as general as possible while avoiding overfitting to the training data.

Figure 6b introduces a more suitable metric called the Mean Average Precision, or MAP. It is used in information retrieval to estimate how well a ranked list of documents matches the topic(s) of a



(a) *Machine learning components*

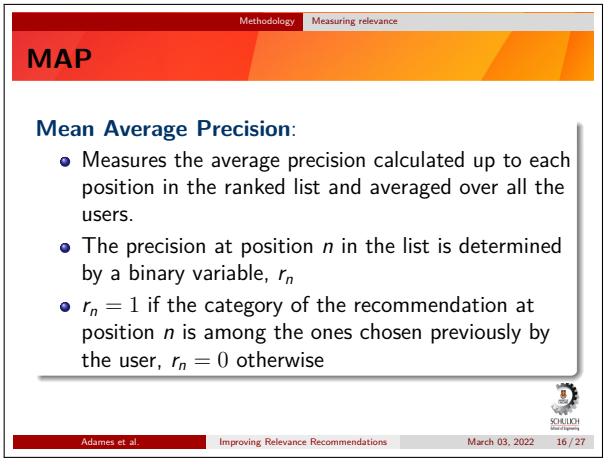


(b) *Recommendation system*

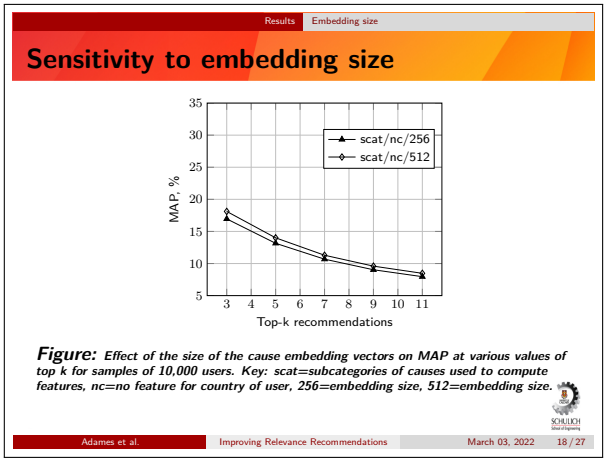
Figure 6: Metrics for assessing the performance of ML components and of the recommendation system.

target document from a library of documents. MAP requires that we can evaluate the relevance of any recommendation for a target user as a binary variable.

MAP gives a more descriptive value representing what percentage of the ranked list is relevant to the target user based on his profile. In our case the user profile is inferred from his donation activity, specifically the causes the user has donated to. A recommendation at position n of he ranked list is assumed to be relevant if is of the same category or subcategory of he donations made by the user, or not relevant if otherwise. More details about the formulas to compute MAP in the original paper. Suffice to say at this point that in order to compute MAP for a system with a large number of users, one has to compute recommendation lists of a certain length for a representative sample of the users, or, if not too computationally intensive one can compute it over the whole set of users.



(a) *Mean Average Precision for ranked lists*



(b) *Effect of embedding size*

Figure 7: MAP and analysis of effect of embedding size.

From this point on the presentation will focus on showing the performance of the system as MAP at various lengths of lists and on how to improve these values as a function of the options available for computing features.

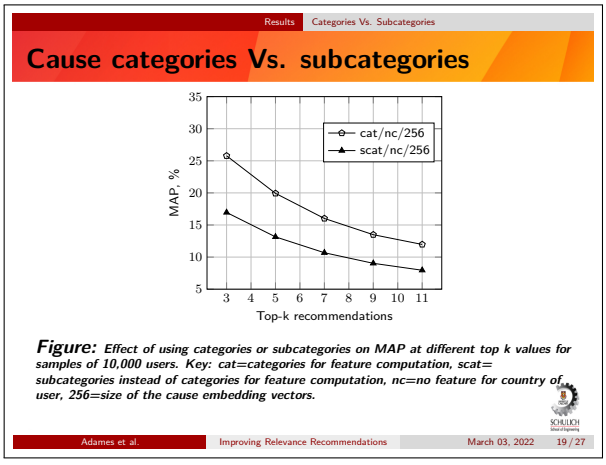
4 Results

The following results were generated by training the machine learning components of the system with different subsets of Donations-2. Then the system was used to produce recommendation lists of various lengths k for the same subset of 10,000 random users from among the ones in the training set. MAP was computed with those 10,000 recommendation lists of length k and then plots were prepared.

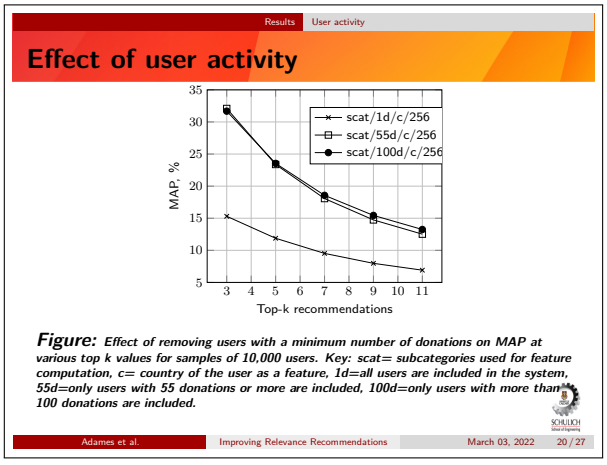
4.1 Embedding size

Figure 7b shows the relative effect of two embedding vector sizes on MAP. The features were computed usimg subcategories of causes and so were they used to compute MAP. The benefit of doubling the length of embeddings from 256 to 512 is slightly better at low top-k values but still only marginal compared to the higher computational time required. The total training time was in the 4 hour mark with size 256 already.

4.2 Categories Vs. Subcategories



(a) ***Effect of the cause taxonomy used***



(b) ***Effect of user activity***

Figure 8: Effects of cause taxonomy and user activity on MAP.

With the embedding vector length fixed at 256, Figure 8a shows the effect of using cause categories or subcategories to compute the features necessary for the implicit ratings. The apparent advantage of categories is offset by the lower user experience when receiving recommendations that are *relevant* due to the fact that they match a broad category in their profile. Choosing cause subcategories enhances the value of a *relevant* recommendation because they tend to be closer related due to the nature of a finer classification.

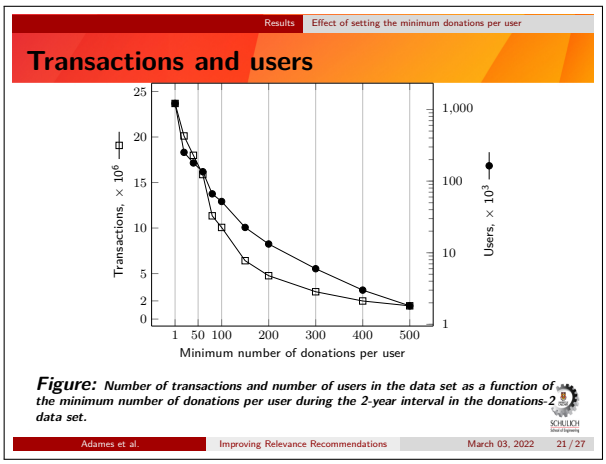
Therefore the decision was made at this point to look for a strategy to improve MAP of recommendations produced by a system trained on implicit ratings created from subcategories.

4.3 User activity

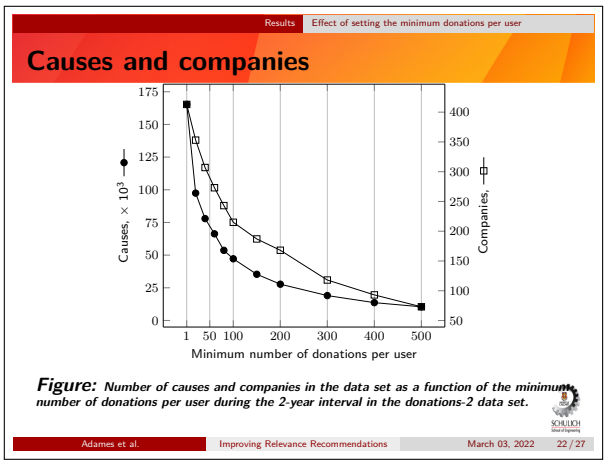
The search for a minimum number of donations per user to optimize MAP was addressed via simulation. Figure 8b shows how after limiting the users to 55 transactions or more the improvement in MAP at any top-k reaches a point of diminishing returns because the system is restricted to a smaller number of users.

The following set of simulations have the purpose of investigating the effect of various minimum number of transactions per user on the different characteristics of the data set. The goal was to investigate if a value around 55 was acceptable in terms of loss of diversity of the original data. This loss of *diversity* was associated empirically with diminishing number of users, causes, transactions, companies, and countries compared to the original data set.

4.4 Effect of setting the minimum donations per user



(a) *Transactions and users*



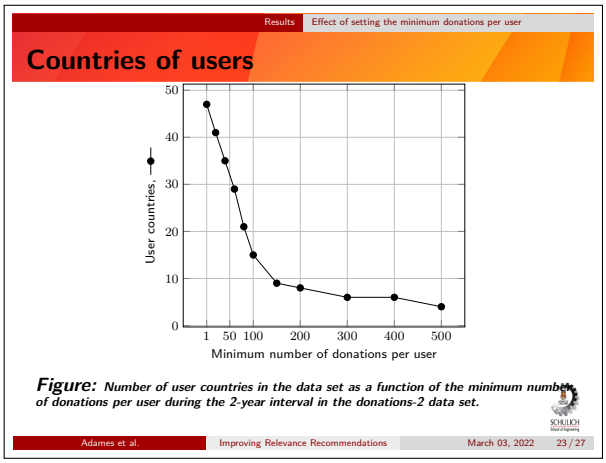
(b) *Causes and companies*

Figure 9: Effect of the minimum number of transactions per user on various characteristics of the original data.

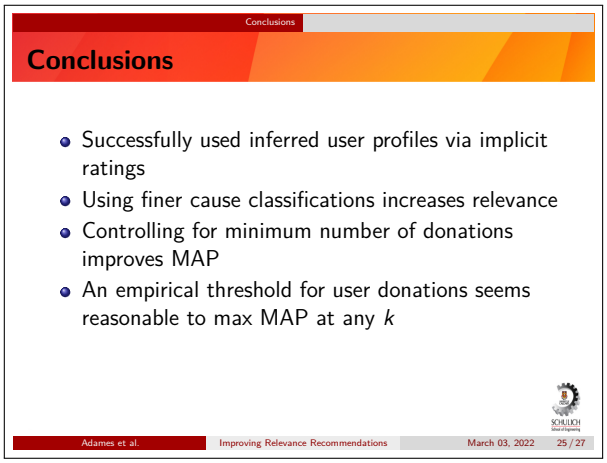
Figures 9a and 9b illustrate the loss of four characteristics of the original data set as a function of the minimum number or transactions per user. These two plots in conjunction with the one in figure 10a confirm that at around 50 transactions per user the rate of loss of diversity is at a maximum and lowering this value may potentially give up applicability of the recommendation system while not improving relevance of recommendations measured as the system MAP.

5 Conclusions

Figure 10b present the conclusions of this research. It was found that a meaningful recommendation system for suggesting charitable causes could be produced via custom feature engineering using the subcategories donated to by the users as the discriminating factor in computing those features and the implicit ratings derived from them.



(a) ***Countries of users***



(b) ***Conclusions***

Figure 10: Effect of the minimum number of transactions per user on the number of countries and conclusions.

Cause subcategories showed empirically to produce more related recommendations due to the fact that they capture more granular types of causes. These finer relations are then learned by the autoencoder used to train the classifier for relevance from which the embeddings vectors are extracted.

Finally relevance can be improved using subcategories if one limits the data used to build the CF algorithm to the transactions of user with 50 or more entries in the data set.

Alternative, less personalized recommendations can be offered to users without this requisite based on general trends like most popular causes, or a simpler content filter based on the donations made by the user.

6 Acknowledgements

Thanks to Benevity and MITACS for funding this research and to the University of Calgary the Master of Enginnering program in Software Engineering that gave us the principles to tackle this subject.

Figure 11: Acknowledgments

Additional Support slides

A data flow graph for the recommendation system that did not make it to the paper for lack of space. It shows the Software Engineer and the user as the two main agents external to the system. Paths 1 through 3 are used to prepare the system via machine learning. Path 4 addresses calculating a recommendation list. Path 5 has to do with computing MAP for system performance assessment.

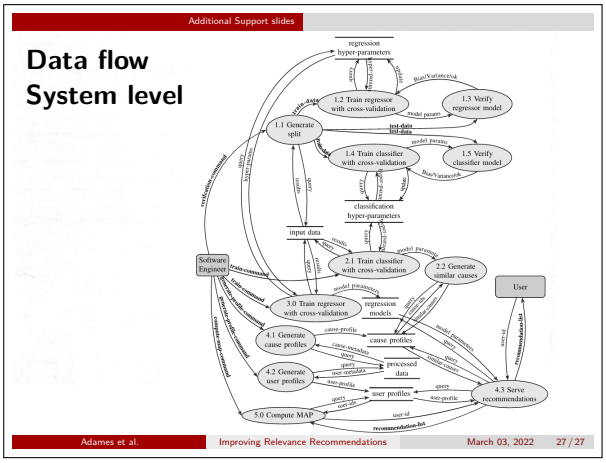


Figure 12: Data flow at system level