Workshop proceedings

ComplexRec 2018

Second Workshop on Recommendation in Complex Scenarios

October 7, 2018
RecSys 2018, Vancouver, Canada

Edited by
Casper Petersen, Toine Bogers, Marijn Koolen, Bamshad Mobasher, Alan Said

Contents

Preface

This volume contains the papers presented at the RecSys 2018 workshop Recommendation in Complex Scenarios (ComplexRec 2018) held on October 7, 2018 at the Parq Vancouver, in Vancouver, Canada.

State-of-the-art recommendation algorithms are typically applied in relatively straightforward and static scenarios: given information about a user's past item preferences in isolation, can we predict whether they will like a new item or rank all unseen items based on predicted interest? In reality, recommendation is often a more complex problem: the evaluation of a list of recommended items never takes place in a vacuum, and it is often a single step in the user's more complex background task or need. These background needs can often place a variety of constraints on which recommendations are interesting to the user and when they are appropriate. However, relatively little research has been done on how to elicit rich information about these complex background needs or how to incorporate it into the recommendation process. Furthermore, while state-of-the-art algorithms typically work with user preferences aggregated at the item level, real users may prefer some of an item's features more than others or attach more weight in general to certain features. Finally, providing accurate and appropriate recommendations in such complex scenarios comes with a whole new set of evaluation and validation challenges.

The current generation of recommender systems and algorithms are good at addressing straightforward recommendation scenarios, but the more complex scenarios as described above have been underserved. The ComplexRec 2018 workshop aims to address this by providing an interactive venue for discussing approaches to recommendation in complex scenarios that have no simple one-size-fits-all solution.

The workshop program contains a set of position and research papers covering many complex aspects of recommendation in various scenarios. There were 14 submissions. Each submission was reviewed by at least 3 program committee members. The committee decided to accept 7 papers (acceptance rate 50%).

We thank the program committee members for their timely and constructive reviews. We gratefully acknowledge the support of EasyChair for organizing paper submission and reviewing and producing the proceedings.

August 28, 2018
Copenhagen

Casper Petersen
Toine Bogers
Marijn Koolen
Bamshad Mobasher
Alan Said

Program Committee

Haggai Roitman	IBM Research Haifa
Jaap Kamps	University of Amsterdam
Soude Fazeli	Open University of Netherlands
Nafiseh Shabib	Norwegian University of Science and Technology
Robin Burke	DePaul University
Paolo Cremonesi	Politecnico di Milano
Marco De Gemmis	Dipartimento di Informatica - University of Bari
Fedelucio Narducci	University of Bari Aldo Moro
Cristina Gena	Department of Computer Science, University of Torino
Tommaso Di Noia	Politecnico di Bari
Fabio Gasparetti	Artificial Intelligence Laboratory - ROMA TRE University
Peter Dolog	Aalborg University
Cataldo Musto	Dipartimento di Informatica - University of Bari
Federica Cena	Department of Computer Science, University of Torino
Oren Sar Shalom	Bar Ilan University
Ludovico Boratto	Eurecat
Markus Schedl	Johannes Kepler University
Panagiotis Adamopoulos	Emory University
Frank Hopfgartner	The University of Sheffield
Juan F. Huete	University of Granada
Ivàn Cantador	Universidad Autónoma de Madrid
Ernesto William De Luca	Georg-Eckert-Institute - Leibniz-Institute for international Textbook Research

2nd Workshop on Recommendation in Complex Scenarios (ComplexRec 2018)

Toine Bogers
Department of Communication & Psychology
Aalborg University Copenhagen
Denmark
toine@hum.aau.dk

Marijn Koolen
Huygens ING, Royal Netherlands Academy of Arts and Sciences
Netherlands
marijn.koolen@huygens.knaw.nl

Bamshad Mobasher
School of Computing
DePaul University
United States
mobasher@cs.depaul.edu

Alan Said
University of Skövde
Sweden
alansaid@acm.org

Casper Petersen
Sampension
Denmark
cap@sampension.dk

ABSTRACT

Over the past decade, recommendation algorithms for ratings prediction and item ranking have steadily matured. However, these state-of-the-art algorithms are typically applied in relatively straightforward scenarios. In reality, recommendation is often a more complex problem: it is usually just a single step in the user's more complex background need. These background needs can often place a variety of constraints on which recommendations are interesting to the user and when they are appropriate. However, relatively little research has been done on these complex recommendation scenarios. The ComplexRec 2018 workshop addresses this by providing an interactive venue for discussing approaches to recommendation in complex scenarios that have no simple one-size-fits-all solution.

KEYWORDS

Complex recommendation

ACM Reference format:
Toine Bogers, Marijn Koolen, Bamshad Mobasher, Alan Said, and Casper Petersen. 2018. 2nd Workshop on Recommendation in Complex Scenarios (ComplexRec 2018). In *Proceedings of Twelfth ACM Conference on Recommender Systems, Vancouver, BC, Canada, October 2–7, 2018 (RecSys '18),* 3 pages.
DOI: 10.1145/3240323.3240332

1 INTRODUCTION

Over the past decade, recommendation algorithms for ratings prediction and item ranking have steadily matured, spurred on in part by the success of data mining competitions such as the Netflix Prize, the 2011 Yahoo! Music KDD Cup, and the RecSys Challenges. Matrix factorization and other latent factor models emerged from these competitions as the state-of-the-art algorithms to apply in both existing and new domains. However, these state-of-the-art algorithms are typically applied in relatively straightforward and static scenarios: given information about a user's past item preferences in isolation, can we predict whether they will like a new item or rank all unseen items based on predicted interests?

In reality, recommendation is often a more complex problem: the evaluation of a list of recommended items never takes place in a vacuum, and it is often only a single step in the user's more complex background task or need. These background needs can often place a variety of constraints on which recommendations are interesting to the user and when they are appropriate. However, relatively little research has been done on how to elicit rich information about these complex background needs or how to incorporate it into the recommendation process. Furthermore, while state-of-the-art algorithms typically work with user preferences aggregated at the item level, real users may prefer some of an item's features more than others or attach more weight in general to certain features. Finally, providing accurate and appropriate recommendations in such complex scenarios comes with a whole new set of evaluation and validation challenges.

The current generation of recommender systems and algorithms are good at addressing straightforward recommendation scenarios, yet more complex scenarios as described above have been underserved. The **ComplexRec 2018** workshop addresses this by providing an interactive venue for discussing approaches to recommendation in complex scenarios that have no simple one-size-fits-all solution. It is the second edition of this workshop, after a successful first edition in 2017 [5]. In addition to this first edition, other workshops have also been organized on related topics in recent years. Examples include the CARS (Context-aware Recommender Systems) workshop series (2009-2012) organized in conjunction with RecSys [1–4], the CARR (Context-aware Retrieval and Recommendation) workshop series (2011-2015) organized in conjunction with IUI, WSDM, and ECIR [6–9, 12], as well as the SCST (Supporting Complex Search Tasks) workshop series (2015, 2017) organized in conjunction with ECIR and CHIIR [10, 11].

RecSys '18, Vancouver, BC, Canada

DOI: 10.1145/3240323.3240332

2 FORMAT & TOPICS

ComplexRec 2018 will be organized as an interactive, half-day workshop. The workshop will start with two paper sessions, for which short papers and position papers of 2-4 pages in length were solicited. Accepted submissions will be invited for short 10-minute presentations with equal time for discussion. Evaluation criteria for acceptance include novelty, diversity, significance for theory/practice, quality of presentation, and the potential for sparking interesting discussion at the workshop. All submitted papers were reviewed by the program committee.

The second half of the workshop is planned to feature an industry panel, dealing with the issues of recommendation in complex real-world scenarios. Finally, the workshop will also feature a keynote presentation, to be shared with the related KARS 2018 workshop on knowledge-aware and conversational recommender systems workshop, which takes over after the ComplexRec workshop ends.

2.1 Topics of interest

Relevant topics for the ComplexRec workshop included:

- **Task-based recommendation** (Approaches that take the user's background tasks and needs into account when generating recommendations)
- **Feature-driven recommendation** (Techniques for eliciting, capturing and integrating rich information about user preferences for specific product features)
- **Constraint-based recommendation** (Approaches that successfully combine state-of-the-art recommendation algorithms with complex knowledge-based or constraint-based optimization)
- **Query-driven recommendation** (Techniques for eliciting and incorporating rich information about the user's recommendation need (e.g., need for accessibility, engagement, socio-cultural values, familiarity, etc.) in addition to the standard user preference information)
- **Interactive recommendation** (Techniques for successfully capturing, weighting, and integrating continuous user feedback into recommender systems, both in situations of sparse and rich user interaction)
- **Context-aware recommendation** (Methods for the extraction and integration of complex contextual signals for recommendation)
- **Complex data sources** (Approaches to dealing with complex data sources and how to infer user preferences from these sources)
- **Evaluation & validation** (Approaches to the evaluation and validation of recommendation in complex scenarios)

3 WORKSHOP SUBMISSIONS

A total of 14 papers were submitted to the workshop, which were all reviewed by a program committee of international experts in the field. Of these papers, 7 were accepted for presentation at the workshop, resulting in an acceptance rate of 50%. The accepted papers cover a range of topics.

De Pessemier et al. attempt to recommend to users which football games to bet on, which team to bet on and how much money to bet based on personal preference of risk and profit potential using a prototype recommendation tool based on different classification models. Their results show e.g. that the betting strategy (which game to bet on) fluctuate substantially, and that SVM and Random Forest classifiers are the most useful classification models.

Ringger et al. propose a home recommendation engine: which homes to recommend for purchase to a user. By combining collaborative-filtering and content-based recommendations, their results show a positive impact of home recommendation on website metrics such as click-through rate.

Drushku et al. seek to help users complete their reports in SAP by grouping queries coming from different documents, that all together bring more information than a ranked list of independent queries. Their results demonstrate that considering short and long-term user profiles, as well as an order on the queries, are essential.

Qian et al. propose an extension of the PRMF model to generate repeat consumption recommendations incorporating the usersâĂŹ tendency to repeat. Using the Tafeng dataset, their evaluation, overall, shows improvements over conventional models, though for different tendencies to repeat the results fluctuate more.

Loepp and Ziegler recommend personalised running routes based on users' preferences, goals and background. Route recommendations are ranked according to an overall score based on 12 different criteria such as length and pedestrian friendliness. Their recommendation engine is implemented in an app and evaluated as a proof-of-concept where users reported that their approach is overall valid and appreciated by the app's users.

Anelli et al. present a framework that exploits the local popularity of items combined with temporal information to compute personalised top-N recommendations by considering a user's neighbours. Their approach is evaluated on three datasets and found to be the best-performing compared to multiple item-based baselines.

Ekstrand et al. consider how to help teachers locate resources for use in classroom instruction using recommendations and information retrieval technology. Through interviews with teachers, they learn that there are multiple stakeholders, objectives and resources that exist and have to be balanced by the teacher for such technology to be effective.

4 WEBSITE & PROCEEDINGS

The workshop material (list of accepted papers, invited talk, and the workshop schedule) can be found on the ComplexRec workshop website at http://complexrec2018.aau.dk/. The proceedings will be made available online and linked to from the workshop website. A summary of the workshop will appear in SIGIR Forum to increase cross-disciplinary awareness of recommender systems research.

REFERENCES

[1] Gediminas Adomavicius, Linas Baltrunas, Ernesto William de Luca, Tim Hussein, and Alexander Tuzhilin. 2012. 4th Workshop on Context-aware Recommender Systems (CARS 2012). In *Proceedings of RecSys '12*. ACM, New York, NY, USA, 349–350.

[2] Gediminas Adomavicius, Linas Baltrunas, Tim Hussein, Francesco Ricci, and Alexander Tuzhilin. 2011. 3rd Workshop on Context-aware Recommender Systems (CARS 2011). In *Proceedings of RecSys '11*. ACM, New York, NY, USA, 379–380.

[3] Gediminas Adomavicius and Francesco Ricci. 2009. RecSys'09 Workshop 3: Workshop on Context-aware Recommender Systems (CARS-2009). In *Proceedings of RecSys '09*. ACM, New York, NY, USA, 423–424.

[4] Gediminas Adomavicius, Alexander Tuzhilin, Shlomo Berkovsky, Ernesto W. De Luca, and Alan Said. 2010. Context-awareness in Recommender Systems:

Research Workshop and Movie Recommendation Challenge. In *Proceedings of RecSys '10*. ACM, New York, NY, USA, 385–386.

[5] Toine Bogers, Marijn Koolen, Bamshad Mobasher, Alan Said, and Alexander Tuzhilin. 2017. Workshop on Recommendation in Complex Scenarios (ComplexRec 2017). In *RecSys '17: Proceedings of the Eleventh ACM Conference on Recommender Systems*. 380–381.

[6] Matthias Böhmer, Ernesto W. De Luca, Alan Said, and Jaime Teevan. 2013. 3rd Workshop on Context-awareness in Retrieval and Recommendation. In *Proceedings of WSDM '13*. ACM, New York, NY, USA, 789–790.

[7] Ernesto William De Luca, Matthias Böhmer, Alan Said, and Ed Chi. 2012. 2nd Workshop on Context-awareness in Retrieval and Recommendation: (CaRR 2012). In *Proceedings of IUI '12*. ACM, New York, NY, USA, 409–412.

[8] Ernesto William De Luca, Alan Said, Matthias Böhmer, and Florian Michahelles. 2011. Workshop on Context-awareness in Retrieval and Recommendation. In *Proceedings of IUI '11*. ACM, New York, NY, USA, 471–472.

[9] Ernesto W. De Luca, Alan Said, Fabio Crestani, and David Elsweiler. 2015. 5th Workshop on Context-awareness in Retrieval and Recommendation. In *Proceedings of ECIR '15*. Springer, 830–833.

[10] Maria Gäde, Mark Michael Hall, Hugo C. Huurdeman, Jaap Kamps, Marijn Koolen, Mette Skov, Elaine Toms, and David Walsh. 2015. First Workshop on Supporting Complex Search Tasks. In *Proceedings of the First International Workshop on Supporting Complex Search Tasks, co-located with ECIR 2015*.

[11] Marijn Koolen, Jaap Kamps, Toine Bogers, Nicholas J. Belkin, Diane Kelly, and Emine Yilmaz. 2017. Current Research in Supporting Complex Search Tasks. In *Proceedings of the Second Workshop on Supporting Complex Search Tasks, co-located with CHIIR 2017*. 1–4.

[12] Alan Said, Ernesto W. De Luca, D. Quercia, and Matthias Böhmer. 2014. 4th Workshop on Context-awareness in Retrieval and Recommendation. In *Proceedings of ECIR '14*. Springer, 802–805.

Recommendations for sports games to bet on

Toon De Pessemier, Bram De Deyn, Kris Vanhecke, Luc Martens
imec - WAVES - Ghent University
{toon.depessemier,bram.dedeyn,kris.vanhecke,luc1.martens}@ugent.be

ABSTRACT

The outcome of sports games, such as football, is non-deterministic since it is subject to many human actions of players and referees, but also injuries, accidents, etc. Betting on the outcome is becoming increasingly popular which is reflected by the growing sports betting market. This research tries to maximize profit from sports betting on football outcomes. Predicting the outcome can be considered as a classification problem (Home team/Draw/Away team). To decide on which games to bet (betting strategy) and the size of the stake of the bet (money management), recommendations can be provided based on personal characteristics (risk taking/risk averse). Profitable ternary classifiers were found for each of the five major European football leagues. Using these classifiers, a personal assistant for bettors was engineered as a recommendation tool. It recommends the betting strategies and money management systems that were the most profitable in recent history and outputs the game outcome probabilities generated by the classifier.

CCS CONCEPTS

• **Information systems → Information systems applications**; *Data analytics*; Data mining;

KEYWORDS

Sports betting, Recommendation, Classification, Data mining

ACM Reference Format:
Toon De Pessemier, Bram De Deyn, Kris Vanhecke, Luc Martens. 2018. Recommendations for sports games to bet on. In *Proceedings of ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios.* ACM, New York, NY, USA, Article 4, 5 pages.

1 INTRODUCTION

Football, also called association football or soccer in some countries, is the most popular sport internationally. But, it is also a sport that can be very difficult to predict because of a whole series of factors that can influence the outcome: current performance and motivation of the 11 players per team on the pitch, and some additional substitutes, decisions made by the players, interactions between players, decisions of coaches and referees, injuries, etc. Therefore, an increasing share of the multi-billion dollar gambling industry is directed to betting on the outcome of football games. Both academical researchers and industrial organizations have a growing interest in the football odds to predict the outcomes thereby profiting from potential market inefficiencies. Most of them focus on football game forecasts and the main objective is often the accuracy of the prediction model, i.e. the fraction of correctly predicted outcomes of football games.

In commercial applications, bookmakers take their share before paying out the winning bets, i.e. the profit margin. In case of a balanced book (e.g. approximately the same amount of money is bet on both outcomes of a fifty-fifty bet) their profit is assured. In case of unbalances, bookmakers might have to pay out more than what was staked in total, or they earn more than was expected. To avoid unbalances, bookmakers allow their odds to dynamically change in proportion to the amount of money staked on the possible outcomes to obtain a more robust book. However, if the bookmakers' odds are significantly deviating from the true event probabilities, these faulty odds provide opportunities to make profit from the bets. Research has shown that the odds of individual bookmakers suffer from these deviations, implying that the gambling market is inefficient [5].

This paper goes further than finding a model with a good accuracy and also considers the profitability of the prediction model (relative to market odds) as an assessment tool . To achieve a profitable model, market odds have to be sufficiently less accurate relative to those generated by the prediction model so that the bookmakers' profit margin can be overcome [7]. The expected profit is calculated based on the discrepancy between the output of the prediction model and the market odds. Only a few research initiatives consider profitability, on top of that, this paper proposes a personal assistant that provides recommendations for betting (which game and which stake), instead of predicting every game.

2 RELATED WORK

Game results and scores have been modeled since the eighties. Maher [15] proposed a Poisson model, in which home and away scores are modeled independently. Lots of improvements on this model have been published since then, such as incorporating time, giving weights to different kinds of scores etc [8]. Besides, the influence of home advantage on the outcome of the game has been proven [7]. Many researchers have investigated features and models to figure out the dependencies between the historic statistics and game results [2, 16, 17]. It has been proven that accuracies above 54% can lead to guaranteed net profit [19], if bettors use an adequate betting method and money management system - assuming that the bookmakers use a moderate profit margin. Neural network, naive Bayes, random forest, and multinomial logistic regression classifiers succeed to achieve accuracies up to 55% [2, 20].

However, it is necessary to consider both accuracy and profit to get the full informative and practical sense of a model's performance [4, 6]. To calculate the possible profit a model could generate, its predicted outcome probabilities have to be compared to the published odds. Bookmakers' published odds have multiple times been shown to be good forecasts for game outcomes [9, 19, 20] and

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, October 7, 2018, Vancouver, Canada.

have been called "the golden odds" for exactly that reason. Studies suggest that bookmakers acquire extra information that exceeds the historical game data available for the public, improving their published odds [6].

Multiple ways exist one can go about betting and choosing the size of the stakes - so called betting strategies and money management systems [1, 13, 14], each with a potential profit and an associated risk. Many studies have been focused on prediction accuracy, thereby neglecting the betting decision users have to make. This paper goes further and proposes how recommendations can assist bettors in their choices: on which game to bet? (betting strategy), how much to bet? (money management), and on which team to bet? (outcome prediction).

3 DATA

Historical data about football games can be retrieved through a sports data provider. For this research the API of SportRadar.com was used. Historical data was fetched for five major national professional European leagues: the Spanish LaLiga, the English Premier League, the Italian Serie A, the German Bundesliga, and the Belgian Pro League. For each league, data were fetched for all games since the 2011/2012 season until the end of 2017 (middle of the 2017-2018 season). Besides statistics about the football games, the data provider also specifies probabilities about the outcomes of games without profit margins. In this research, 109 different features were considered. Most of them are available for both the home playing team (HT) and the away playing team (AT). In addition, some features are specific for the past confrontations of the two teams, also called head-to-head games (H2H).

- *Recent Game Results.* To predict the outcome of a game, a set of obvious features represents the outcome of the most recent game(s) of the teams: win, draw or loss. For each prediction of a game between a specific home team and a specific away team, the most recent games played by the home team (HT Recent Games) as well as the most recent games of the away team (AT Recent Games) are considered. In addition, the most recent confrontations between home and away team are a feature (H2H Recent Games).
- *Goal (Difference).* The difference in goals (scored goals minus against goals) during the most recent games is used to estimate the effectiveness of the team. A strictly positive number means that the considered team won, whereas a strictly negative number indicates a loss. Zero stands for a draw. Large differences in the number of goals reflect large performance differences between the teams. Besides, for each team also the absolute number of scored goals is a feature.
- *Ranking.* The number of points that the team won in the national league during the current season is a measure for its performance (win=3,draw=1,loss=0 points). To compensate for a different number of games played by different teams, the number of points is divided by the number of games played by the team in that league.
- *Fatigue.* Consecutive games might exhaust a team, and cause a poor performance in the next game. The number of games played by the team in the last couple of weeks is used as an indicator for the fatigue of the team. Also the distance that the away team has to travel is used as a feature, since long trips may fatigue the team.
- *Historical game statistics.* Many game statistics of the previous games can be an indicator of a well or poorly performing team. The following were considered: ball possession, free kicks, shots on target, shots off target, shots saved, offsides, yellow cards, yellow-red cards, red cards, corners, successful passes, successful crosses, successful duels, and created chances.

  Many of these statistics (such as recent game results, goal difference, or historical game statistics) can be aggregated over a longer period of time, or aggregated over multiple games to obtain a more reliable value. The results of the 5 most recent and 10 most recent games were considered (older games are considered as less relevant). E.g. a feature can aggregate the amount of goals made by the team during the last 10 games.

4 FEATURE SELECTION

Football games are characterized by a rich set of features, and for each team the past performance is available as the outcome of previous games. An important research question is: Which of these features (based on historical records) are correlated to the outcome of the game that has to be predicted? For feature selection, four algorithms of the WEKA workbench [10, 18] were used: OneR, InfoGain, GainRatio and Correlation. The OneR algorithm assesses the importance of each feature by evaluating the corresponding one feature classifier, and ranking all these classifiers. InfoGain evaluates the worth of a feature by measuring the information gain with respect to the class. GainRatio is similar but measures the information gain ratio. So, both algorithms are evaluating how much a feature reduces the entropy. The Correlation ranker calculates the Pearson correlation between the feature and the result class, i.e. a linear relationship between both is searched.

Table 1 shows the features with the highest information gain according to the InfoRatio algorithm. The results of the other algorithms are consistent. These results show that features derived from the goal differences in the recent past are the most important. In addition, the recent game results of both teams, and the results of the H2H games provide a significant information gain. Also the ranking of both teams in the national league can be used to predict the game result. The absolute number of goals scored by both teams has a lower information value as well as the results of the last H2H games. Noteworthy, none of the historical game statistics and none of the features reflecting the fatigue of the team was found to have a significant information gain.

5 GAME OUTCOME PREDICTION

The goal of the model is to predict which team wins the game (Home team/Draw/Away team). This is tackled as a classification problem with unbalanced classes because of the home advantage [15] (Approximated probabilities based on historical data: 45% Home team, 25% Draw, 30% Away team). While cross validation is considered to be the standard evaluation method, it does not reflect a realistic scenario for sport predictions where the date of the game is important. Cross validation would allow the classifier to find dependencies that cannot be replicated outside the training and evaluation phase. Therefore, a more realistic evaluation approach is adopted by splitting the labeled data thereby remaining the chronological order of the games. An 80% split is used, which means the most recent

Attribute	Information Gain
HT Goal Difference - 10 most recent games	0.0398
H2H Goal Difference - 10 most recent games	0.0379
H2H Goal Difference - 5 most recent games	0.0361
HT Ranking	0.0358
HT Recent Game Results - 10 most recent games	0.0356
AT Goal Difference - 10 most recent games	0.0352
HT Goals Made - 10 most recent games	0.0312
HT Goal Difference - 5 most recent games	0.0310
H2H Recent Game Results - 10 most recent games	0.0303
AT Ranking	0.0297
AT Goals Made - 10 most recent games	0.0293
H2H Recent Game Results - 5 most recent games	0.0284
AT Goal Difference - 5 most recent games	0.0277
AT Recent Game Results - 10 most recent games	0.0275

Table 1: Features with the highest information gain.

20% of the games is predicted with models that were trained on the oldest 80%.

To avoid overfitting, three *feature reduction* methods are tested to reduce the number of features to 25. The first method is based on the Pearson correlation. Features are ranked by their correlation with the game outcome, and only the 25 features with the highest correlation values are used for classification. The GainRatio method works similar and only keeps the 25 features that have the highest information gain. Principal Component Analysis is a more advanced method and transforms the 109 features into a reduced set of 25 new features which are a combination of the original features.

To measure the accuracy improvement of complex classifiers, two simple, *baseline predictors* were used. ZeroR uses none of the features and predicts the majority result class for every record. So, ZeroR always predicts the home team to be the winner of every game. OneE is a predictor based on one feature, the feature that produces the smallest error for the training set. For the other predictors, different classifiers available in WEKA and LibSVM [3] are used.

- *Support Vector Machines (SVM)* are non-probabilistic binary linear classifiers. The used SVMs of WEKA are trained using Sequential Minimal Optimization (SMO). In addition, the C-SVC (Support Vector Classifier) type of LibSVM is used. Different kernels are evaluated: linear kernels, polynomial kernels (standard and normalized version), sigmoid kernels, RBF (radial basis function) kernels, and PUK (Pearson function-based universal) kernels.
- *Naive Bayes Classifiers* are probabilistic classifiers with the assumption that features are independent (WEKA).
- *Multi Layer Perceptrons (MLP)* are feedforward neural networks utilizing backpropagation as supervised learning technique (WEKA).
- *Random Forest* is an ensemble technique using multiple learning algorithms to obtain less overfitting and better predictive performance (WEKA).
- *Bagging* is a bootstrap ensemble method. As a base learner, it uses REPTree, a decision tree based on information gain.
- *Simple Logistic Regression* estimates the probability of a binary outcome using a logistic function. For fitting the logistic models, LogitBoost (ensemble algorithm) with simple regression functions as base learners is used (WEKA).

Table 2 lists the accuracy of the different prediction models based on data of the five national football leagues. Each predictor was evaluated with the full set of 109 features (Full), and with reduced sets of 25 features. These reduced sets are generated using the GainRatio (GR), Correlation (Corr.) or Principal Component Analysis (PCA) technique. All results above 53.50% are in bold, since these models are useful in view of generating profit [19]. Simple classifiers, such as OneR, provide already an accurate baseline, as is common for classification problems [11]. The best result (54.37%) was obtained using RandomForest.

Model	Full	GR	Corr.	PCA
ZeroR	47.46%	-	-	-
OneR	52.29%	(`ht_goal_diff10`)		
SMO (PolyKernel)	52.54%	52.95%	53.001%	**53.81%**
SMO (Norm.PolyKernel)	48.62%	53.15%	53.10%	52.84%
SMO (RBFKernel)	52.14%	52.54%	52.54%	47.46%
SMO (Puk)	48.16%	**53.51%**	**53.81%**	49.44%
C-SVC (sigmoid)	**53.71%**	53.20%	53.45%	**53.71%**
C-SVC (polynomial)	**53.76%**	52.54%	52.44%	46.38%
C-SVC (radial)	52.95%	**53.81%**	**53.71%**	**53.96%**
C-SVC (linear)	53.15%	52.79%	52.84%	**53.76%**
NaiveBayes	27.00%	49.79%	50.40%	26.75%
MLP	51.93%	53.30%	53.20%	52.89%
RandomForest	**53.96%**	**53.71%**	**54.37%**	52.54%
Bagging	47.45%	47.45%	47.45%	47.45%
SimpleLogistic	52.89%	52.89%	52.89%	52.84%

Table 2: Accuracy of the predictors for data of all leagues.

The analysis was repeated for each league separately, since most teams do not play (often) against teams of other leagues. Support vector classifiers (C-SVC of LibSVM) showed to be the most consistent models over the leagues. Table 3 shows the most accurate model per league, together with the Kernel, the optimal value of the complexity parameter C, and the used technique to reduce the number of features. Large accuracy differences were witnessed over the different leagues. The highest accuracy was achieved for the Premier League, followed by the Serie A and LaLiga.

6 BETTING RECOMMENDATIONS

The accuracy results of Section 5 are calculated as if a bet was placed on every game. However, better results, in terms of accuracy and profit, can be achieved by holding off on some of the more uncertain bets. Therefore, different *betting strategies* can be considered. Bettors typically have their own preferences or decision rules to decide on a bet. Often these decisions are driven by the risk users are willing to take.

- *Published favorites.* This simple, baseline strategy is to always bet on the team that is the favorite, according to the published odds.
- *Predicted favorites.* This strategy always bets on the team that is the favorite, according to the predicted odds. If the model is more accurate than the published odds, this strategy can be profitable.
- *Predicted safe favorites.* A bet will only be placed if one of the teams is the clear favorite. In this experiment, betting is done if the probability that the favorite wins is at least 10% higher than

League	Accuracy	Kernel	C	Reduction technique
LaLiga	55.30%	linear	0.5	PCA
Premier League	60.09%	radial	1	GainRatio/Corr.
Serie A	57.83%	linear	0.125	None
Bundesliga	51.87%	sigmoid	4.0	None
Pro League	49.52%	radial	2.0	GainRatio

Table 3: The LibSVM parameters with the highest accuracy.

the other game outcomes. This strategy is more robust and often recommended to risk-averse users.

- *Playing the odds.* This is a commonly-used term in the betting jargon. If users suspect that the odds of a game published by the bookmaker are not correct, they bet on the game. This incorrectness can be estimated by comparing the published odds with the estimated probabilities of the model. Bets will be placed on outcomes that are underestimated by the bookmaker, but only if the probability of the prediction model is at least 10% higher than the probability of the bookmaker. Since this strategy also bets on underdog teams, it is recommended to users who are willing to take more risk with the perspective of a higher profit.
- *Home underdogs.* Bets are made if the away playing team is the favorite and the difference in probability between the home and away playing team is at least 10%. Because of the bookmakers' bias towards higher ranked teams (favorites), this strategy can be profitable [5]. Bookmakers often overestimate the odds of the favorite team, and underestimate the effect of the home crowd of the underdog. Since this strategy always bets on underdog teams, it is recommended to users who take big risks.

Besides recommendations for deciding on which games to bet (betting strategy), users can get recommendations for the size of the stake of the bet (money management). The output of the different *money management* (MM) strategies is a real number between 0 and 1, which can be multiplied by the maximum amount of money the user wants to spend per bet.

- *Unit bet (UB).* In this simple strategy, every bet gets the same stake, 1. This is a high risk, high reward MM strategy, since bets with a high risk get a high stake and thus a high potential profit.
- *Unit return (UR).* This strategies determines the stake size based on the odds to obtain equal unit sized returns. So, each winning bet yields the same amount of money, 1. UR is recommended to risk-averse users since risky bets receive a lower stake.
- *Kelly Ratio (KR).* This strategy is typically used for long term growth of stock investments or gambling [12]. The strategy is based on the difference between the model's estimated probabilities and the bookmaker's odds, and is therefore similar to playing the odds. If the model's probability is much higher than the bookmaker's odd, the bet is placed with a high stake. This strategy focuses on a consistent profit growth in the long term.

To evaluate the betting and MM strategies, a simulation is performed based on historical data. A fixed profit margin of 7.5% (This is an upper bound for realistic profit margins) is used to calculate the bookmaker's odds from the probabilities without profit margin. Again, the most recent 20% of the games are used for evaluation. Figure 1 shows the results of the different betting and MM strategies obtained with the best model for the Premier League (SVM with SMO and RBF Kernel). Playing the odds as betting strategy and unit bet as MM strategy showed to have the highest profit. The total profit after about 340 bet opportunities is 29.48 times the unit stake. However, this combination of strategies is characterized by strong fluctuations. A more risk-averse user can be recommended to use playing the odds in combination with UR or KR. Voting for the underdog was not profitable.

This analysis was repeated for the other leagues as well. Playing the odds and UB showed to have the highest profit potential; but for some seasons/leagues also big losses were made. This indicates that another strategy might be optimal for each league, but also that the results are strongly influenced by the game outcomes.

To demonstrate the prediction models, an interactive tool (called the betting assistant) generating rule-based recommendations for sports betting was developed. Users first specify their risk profile, which determines their matching betting and MM strategy. Optionally, they can specify their betting preferences such as the league. Subsequently, users get recommendations for football games to bet on, together with a recommendation for the size of their stake (value ranging from 0 to 1). Then, it is up to the user to accept the betting advice or not.

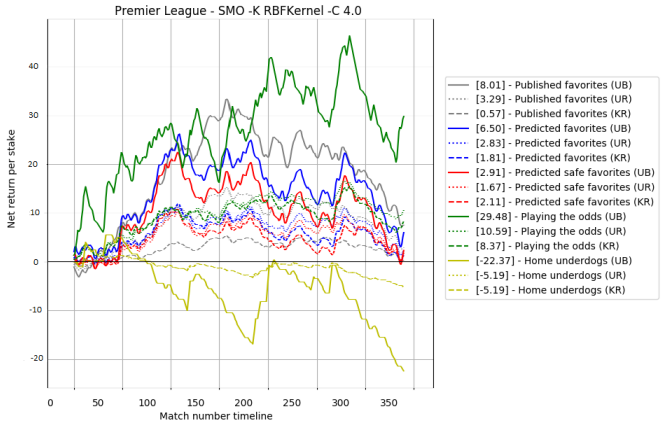

Figure 1: The evolution of different betting strategies.

7 CONCLUSIONS

Predicting the outcome of football games is a research topic with a growing interest. Various prediction models are assessed for this classification problem based on data of five European leagues. The game predictions are used in a prototype recommendation tool that suggests users on which game to bet (betting strategy), on which team (prediction outcome), and how much to bet (money management) depending on their personal preferences regarding risk and profit potential. These prediction models might be applied to other domains as well, such as predicting stock prices, or the outcome of elections. In future work, we will investigate the causality between game features (such as number of offsides, free kicks, etc.) and the game outcome in order to identify the drivers of the game's outcome. These drivers may expose the weaknesses of a team, which can be used by the team's coach to focus on specific tactical aspects during training sessions.

REFERENCES

[1] Georgi Boshnakov, Tarak Kharrat, and Ian G. McHale. 2017. A bivariate Weibull count model for forecasting association football scores. *International Journal of Forecasting* 33, 2 (2017), 458–466.

[2] Maurizio Carpita, Marco Sandri, Anna Simonetto, and Paola Zuccolotto. 2016. Discovering the drivers of football match outcomes with data mining. *Quality Technology and Quantitative Management* 12, 4 (2016), 561–577.

[3] Chih-Chung Chang and Chih-Jen Lin. 2018. LIBSVM - A Library for Support Vector Machines. https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[4] Kong Wing Chow and Karen Tan. 1995. The use of profits as opposed to conventional forecast evaluation criteria to determine the quality of economic forecasts. *Applied Economics Research Series* 1 (1995), 187–200.

[5] Anthony Costa Constantinou and Norman Elliott Fenton. 2013. Profiting from arbitrage and odds biases of the European football gambling market. *Journal of Gambling Business and Economics* 7, 2 (2013), 41–70.

[6] Anthony Costa Constantinou, Norman Elliott Fenton, and Martin Neil. 2012. Pi-football: A Bayesian network model for forecasting Association Football match outcomes. *Knowledge-Based Systems* 36 (2012), 322–339.

[7] Anthony Costa Constantinou, Norman Elliott Fenton, and Martin Neil. 2013. Profiting from an inefficient Association Football gambling market: Prediction, Risk and Uncertainty using Bayesian networks. *Knowledge-Based Systems* 50 (2013), 60–86.

[8] Mark J. Dixon and Stuart G. Coles. 1997. Modelling Association Football Scores and Inefficiencies in the Football Betting Market. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 46, 2 (1997), 265–280.

[9] David Forrest, John Goddard, and Robert Simmons. 2005. Odds-setters as forecasters: The case of English football. *International Journal of Forecasting* 21, 3 (jul 2005), 551–564.

[10] Eibe Frank, Mark A Hall, and Ian H Witten. 2016. The WEKA Workbench. *Morgan Kaufmann, Fourth Edition* (2016), 553–571. http://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

[11] Robert C Holte. 1993. Very simple classification rules perform well on most commonly used datasets. *Machine learning* 11, 1 (1993), 63–90.

[12] John L Kelly Jr. 2011. A new interpretation of information rate. In *The Kelly Capital Growth Investment Criterion: Theory and Practice*. World Scientific, 25–34.

[13] Siem Jan S.J. Koopman and Rutger Lit. 2015. A dynamic bivariate Poisson model for analysing and forecasting match results in the English Premier League. *Journal of the Royal Statistical Society. Series A: Statistics in Society* 178, 1 (2015), 167–186.

[14] Helge Langseth. 2013. Beating the bookie: A look at statistical models for prediction of football matches. In *Frontiers in Artificial Intelligence and Applications*, Vol. 257. 165–174.

[15] Michael J Maher. 1982. Modelling association football scores. *Statistica Neerlandica* 36, 3 (1982), 109–118.

[16] Byungho Min, Jinhyuck Kim, Chongyoun Choe, Hyeonsang Eom, and R. I. (Bob) McKay. 2008. A compound framework for sports results prediction: A football case study. *Knowledge-Based Systems* 21, 7 (oct 2008), 551–562.

[17] Joel Oberstone. 2009. Differentiating the Top English Premier League Football Clubs from the Rest of the Pack: Identifying the Keys to Success. *Journal of Quantitative Analysis in Sports* 5, 3 (2009).

[18] The University of Waikato. 2018. Weka 3 - Data Mining with Open Source Machine Learning Software in Java. https://www.cs.waikato.ac.nz/ml/weka/

[19] Martin Spann and Bernd Skiera. 2009. Sports forecasting: A comparison of the forecast accuracy of prediction markets, betting odds and tipsters. *Journal of Forecasting* 28, 1 (jan 2009), 55–72.

[20] Niek Tax and Yme Joustra. 2015. Predicting the Dutch football competition using public data: A machine learning approach. *Transactions on Knowledge and Data Engineering* 10, 10 (2015), 1–13.

Finding Your Home: Large-Scale Recommendation in a Vibrant Marketplace

Eric Ringger, Alex Chang, David Fagnan, Shruti Kamath, Ondrej Linda,
Wei Liu, Imri Sofer, Nicholas Stevens and Taleb Zeghmi

Zillow Group, Seattle, Washington, USA
Contact: ericri@zillow.com

ABSTRACT

Finding a home for purchase is a complex problem requiring reliable data, a non-trivial amount of time, and (often) good human assistance. Despite the dynamic nature of the vibrant housing market, the rarity of individual purchases, and the variety of a user's intents throughout the lifetime of that user's participation in the market, home recommendation can substantially aid a buyer in exploring the housing market and finding suitable, available homes. Unlike retail products, each home is unique. Market research reveals increased competition, so it is increasingly critical for a potential buyer to find relevant homes in a timely fashion and to follow through on making an offer. To help, we leverage the interaction history of a potential buyer to make automatic home recommendations at national market scale. We adapt state of the art methods for recommendation and combine them in a custom way to address the unique challenges of this market. We introduce engaged buyers to active listings and demonstrate significant re-engagement. The same methods are also fundamental to enhancing the browsing experience of prospective buyers.

1INTRODUCTION

Recommender systems and personal feeds should show the right content to the right person at the right time. Finding a home is a problem well suited to automatic personalized recommendations like those from a well-informed real estate agent. Home recommendation is a complex recommendation problem due to the nature of the housing market, the rarity of purchases, and the variety of a user's intents over the course of that user's participation in the market. Unlike retail products, on average there is increased competition for each unique home. Location is paramount and a valuable differentiator among otherwise similar homes. Furthermore, market research by Zillow, a large real estate data portal, reveals increasing competition over those homes in the U.S. market. The supply of new homes on the market is holding steady over time: over the timeframe of mid-2012 through early 2018, the deseasonalized, smoothed *new listing* count is relatively flat, just below 500K homes. Meanwhile, the *daily inventory* of homes is shrinking from approx. 1.8 million to approx. 1.2 million homes over the same time period, primarily because the average number of days on the market is dropping [1]. Figure 1 depicts the relationship of the normalized versions of these counts.

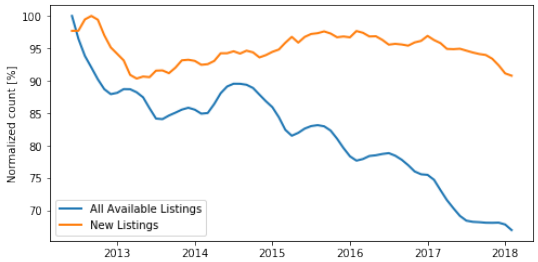

Figure 1. (top/orange) the new listings count normalized by the initial count (as of June 2012) and (bottom/blue) the normalized total number of listed homes. The normalized new listings count is holding nearly steady while the daily inventory is shrinking.

The consequence is more pressure on an often novice potential buyer to quickly identify a relevant home and make an offer. Home buying is also a rare event for which the buyer's own experience is limited. Based on an industry survey, 42 percent of buyers are first-time buyers [2]. For a buyer, gathering information about a home's suitability and quality is a difficult and time-consuming process and goes beyond online shopping. Multiple people -- including (usually) the agents for the seller and the buyer -- get involved to help acquire those signals and to help the transaction succeed. Online shopping is an important part of the process: 79 percent shop online for their home, but seeing the property in person is important: 78 percent of buyers take a private tour of the home, and 43 percent of buyers attend an open house. Because of the importance of the process and the challenge of finding a good fit, a typical U.S. buyer spends 4.3 months searching for their new home prior to making an offer [3].

The purchase of a home is the largest single purchase for most people and is often an exercise in compromise. According to the same survey, in order to meet their criteria, 29 percent of buyers spend more than originally intended on a home. For buyers under 37 years of age, that fraction jumps up to 37 percent and for urban buyers: 42

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, Vancouver, Canada.

percent! Similarly, 41% of buyers say that finding a home within their desired price range was their top challenge. Finding a suitable home is half the battle, as 46 percent of buyers submit two or more offers on a home. In competitive markets, the typical buyer submits more than four offers [3].

To aid in the home-buying process, we leverage the online interaction history of a potential buyer on the Zillow.com real estate site to make automatic home recommendations at national market scale. The challenge of making suitable home recommendations is amplified by the variety of intents of a user: about 45% of visitors intend to buy to own within the next 12 months; the remaining 55% are some mix of exploring (incl. tourism), understanding their neighborhood, understanding the market more broadly, or other unknown intents. In section 2, we describe our adaptation of state of the art methods for recommendation to address the unique challenges of making timely recommendations in the U.S. housing market. In section 3 we briefly describe both offline and online experimental results involving recommendations provided in email notifications.

2 HOME RECOMMENDATIONS

2.1 Data Sources

We start with data on 110 million U.S. homes. Approximately 1.2 million homes are on the market on any given day, and it is that pool of on-market homes that we consider to be eligible for recommendation. Property data is drawn from county governments, banks, site users, real estate listings, and other third parties. For-sale listings come from real estate Multiple Listing Services (MLS), brokers, and other listing sources. User-property interaction history comes from over 160 million monthly active users on the Zillow.com real estate website to the tune of over 1 TB of user events per day, including visiting a home detail page, tagging a home as a favorite, or saving a home for later view.

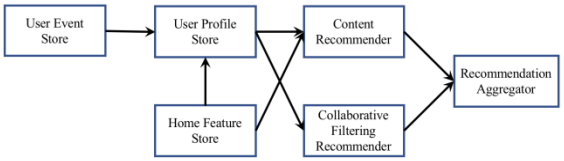

Figure 2. High-level flow from user events to recommendations.

2.2 Recommendation Engine Design

The recommendation engine consists of several components, as depicted in Figure 2. First, the user events are collected in the User Event Store (UES). A catalog of all listed homes from the present as well as the past are collected with their attributes in the Home Feature Store (HFS). On a daily basis, events from the UES are combined with home features from the HFS into user profiles stored in the User Profile Store (UPS). A user profile contains the aggregated information about the user's home preferences along with the list of that user's recently interacted homes. Next, profiles from the UPS are used as input to a Collaborative Filtering (CF) Recommender, and profiles together with home features from HFS are used as inputs to the Content-based (CB) Recommender. Finally, the recommendations from CF and CB are combined in the Recommendation Aggregator (RA) and distributed via multiple channels to prospective buyers. The following sections describe the details of the collaborative filtering and content-based recommender systems and their inclusion specifically in email campaigns.

2.3 Collaborative Filtering

Collaborative filtering (CF) methods enjoy widespread popularity due to their ability to leverage buyers' shared interests without explicitly modeling the attributes of the items they care about. While many CF applications train on explicit user feedback, such as customer reviews or star ratings, we leverage implicit feedback, the nature of which we discuss below. In particular, we employ the method of Implicit Matrix Factorization (IMF) for collaborative filtering [4]. The IMF method generates recommendations for a particular user by inferring those recommendations from the user's own preferences along with other users' overlapping preferences. These preferences are reflected in the implicit feedback provided by user events. One important and unique aspect of home recommendations is that our user-item interaction matrix is not only highly sparse, but it also consists -- essentially -- of several disjoint sub-matrices. This disjointness is due to users interacting with homes primarily in a single region where they are looking to buy a home. For example, a user who searched for homes in Seattle is very unlikely to interact with homes on the East coast in the same home search effort. Hence, due to this locale-specific interest, we apply IMF independently in each geographic zone. Figure 3 shows an illustrative comparison of traditional e-commerce and our home market user-item matrix.

Figure 3. Illustrative depiction of the User-Item matrix for traditional e-commerce vs. home shopping.

We experimented with combinations of different types of implicit feedback representing the degree of a user's

preference for a home, such as (1) the time that a user viewed (or "dwelled" on) a home detail page including its photos, (2) whether the user saved or shared a home, (3) whether the user expanded a particular section of the home detail page, or (4) whether the user scrolled to the end of the home detail page. Our experiments concluded that normalized dwell time on a home detail page is the signal of implicit feedback that optimizes the quality of home recommendations from IMF. In particular, IMF recommendations trained on normalized dwell time from days 1 through N give the best performance in terms of the precision and recall of homes viewed on day N+1. Future work should consider how to best leverage the disparate implicit feedback signals collectively, as in the recent work of Shalom et al. [5].

IMF computes short, dense vector representations for each user and each item (home listing) that capture the key elements of a user's preferences and the distinct nature of a home. The dot product of the two latent vectors predicts preference values. For any given user, the highest item scores compromise the top recommendations, and we remove homes already seen by that user in previous recommendations to create a fresh set of relevant recommendations. For model training we use the method of Alternating Least Squares (ALS), which updates the user and item vectors in alternating fashion while keeping the other fixed. Each step of ALS results in further minimization of the loss function, namely the aggregated square of the difference between the predicted preference score and the observed preference. The number of factors (i.e., latent vector dimensions) is optimized by balancing performance with computation time. Figure 4 shows the improvement in precision@10 in various geographic regions against the number of factors. For prediction, we assign users to regions based on their most recent activity and run ALS for each region in parallel.

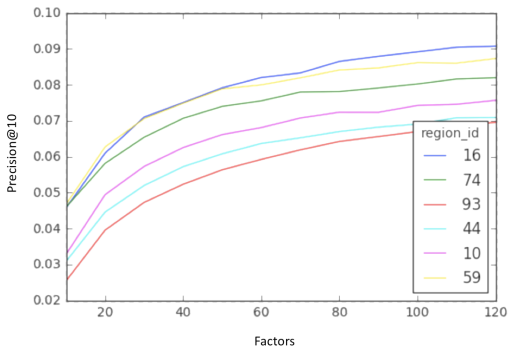

Figure 4. Precision@10 for various factor sizes (vector dimensions)

2.4 Content-based Recommendations

The high relevance of new homes on the market requires another approach. Without implicit feedback for a new listing, the collaborative filtering method has no basis for recommendation. Consequently, we employ a content-based recommendation system to address this cold start problem. The content-based method generates a set of candidate homes (including new listings) for a user based on the user's most likely preferred regions. Next, a classification model is used to score all candidate homes based on how well they match the user's profile.

Unlike the collaborative filtering user representation, this user profile representation requires an explicit enumeration of attributes of interest. The user profile is composed of histograms of sequence-weighted counts [6] of attribute values across several home attributes, such as location (Figure 5 top), price (Figure 5 center), size in square feet (Figure 5 bottom), number of bathrooms, number of bedrooms, etc. As such, the user profile provides a snapshot of a user's interest in time.

The goal of candidate generation for each user is to select a subset of all active homes which is both large enough to contain all relevant homes and small enough to minimize false positives and improve computational efficiency. To achieve this balance, we perform candidate generation based on a user's preference towards location and price. For each user, the set of all active homes is first filtered using a set of the(up to) 10 most relevant postal zip-codes based on the user's profile histogram. Next, the candidate set is further filtered by considering only homes within the user's likely price range, from the 5th to the 95th percentile of the user's price histogram.

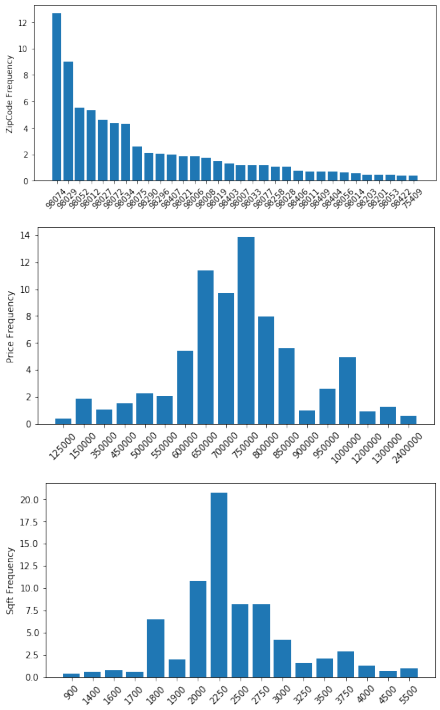

Figure 5. Dimensions from a sample user's profile: (top) Zip Code preference, (center) Price preference, (bottom) Square footage preference.

For a specific user and for each home in the associated candidate set, features is extracted. These features are designed to represent (a) the quality of the user-home match as well as (b) the general popularity and attributes of each home. For match quality, each attribute from the home item profile is used as a key into the corresponding user profile attribute histogram, and the strength of the user's preference for that value of the home's attribute is used as the feature value. These match-based features are then supplemented with non-user-dependent home attributes, such as general home landing page click-through rate or neighborhood popularity.

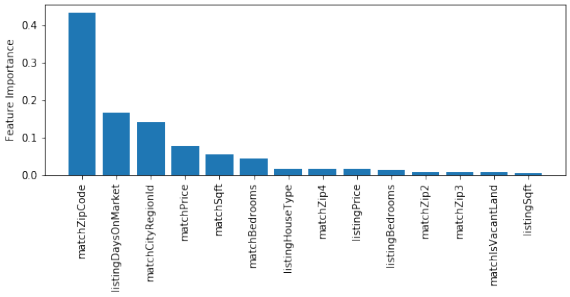

Figure 6. GBDT content model feature importance

To train a scoring model that predicts the quality of match between user profile and specific home, each input feature vector is annotated with a binary label. This label is set according to whether a user interacted with a candidate home (e.g., visited the home detail page, saved for later or tagged as a favorite) or not. Due to an abundance of negative samples, negative class subsampling is applied [7]. Many machine learning models are suitable for modeling this user-home preference. For the results presented in this paper, the content recommendation system uses a Gradient Boosted Decision Tree (GBDT), since its performance was superior to other model types measured. Figure 6 depicts the feature importance for the GBDT model.

2.5 Recommendation Aggregation

The collaborative filtering method has a bias toward listings that have been on the market for a longer time, whereas the content-based approach includes newer listings by design. To observe this visually, we compare the mean number of days on the market for the top 10 recommended homes for each user for both the CF and the CB methods in Figure 7. The optimal combination of these two approaches into a single set of recommendations is the subject of ongoing research. We seek a data-driven, model-based approach to combine both types of recommendations: whether blending scores, using a meta-model, or stacking both models will perform best for our application remains to be seen (c.f., [8]).Currently, we combine recommendations by interleaving both types of recommendations.

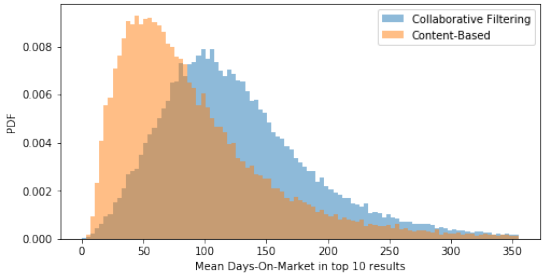

Figure 7. Mean days on the market for content-based and collaborative filtering models.

3 EXPERIMENTAL RESULTS

The home recommendation engine described above has been successfully deployed to production and used to deliver relevant recommendations to customers via several channels. In particular, personalized recommendations are incorporated into several email campaigns, used to power mobile app push notifications, and displayed in collections on the website. We focus here on personalized recommendations through the *email channel*. See Figure 8 for an example of such recommendations.

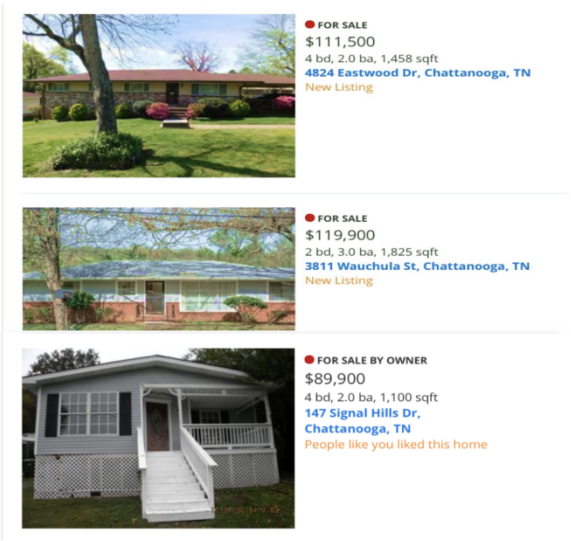

Figure 8. Example home recommendations in email

The baseline strategy for the campaign was a filter-based saved search strategy. We ran an A/B test comparing the filter-based baseline solution to the recommendation-based solution from our system and observed significant relative lifts across most of our key metrics, including click-through rate, as summarized in Table 1.

Metric	Relative Uplift
Home click-through rate	+14.30%
Home agent contact rate	+17.20%
Home save rate	+18.60%
Home share rate	+13.70%

Table 1. Impact of home recommendations on website metrics.

4 CONCLUSIONS

The home buying process stands to benefit significantly from automated recommendations that are personalized to individual preferences. By combining collaborative-filtering and content-based recommendations, we provide significant re-engagement with prospective buyers, introducing them to relevant active listings. As presented our work does not address how to engage users at different stages of their home-buying journeys. One dimension of our ongoing work focuses on modeling the stager of the user's journey and customizing the mode, channel, and frequency of recommendations.

REFERENCES

[1] Gudell, S. 2017. "Inventory Is Down, But Listings Aren't", June 2017 Market Report, Zillow Group, Publication date: Jul. 20, 2017, https://www.zillow.com/research/june-2017-market-report-15956/

[2] Zillow Group. 2017. "Zillow Consumer Housing Trends Report 2017". https://www.zillow.com/report/2017/buyers/

[3] Zillow Group. 2017. "Zillow Consumer Housing Trends Report 2017: Buyers Challenges".https://www.zillow.com/report/2017/buyers/challenges/

[4] Hu, Y. and Koren, Y. and Volinsky, C. 2008. "Collaborative Filtering for Implicit Feedback Datasets".IEEE International Conference on Data Mining (ICDM 2008). Pp. 263-272. https://dl.acm.org/citation.cfm?id=1511352

[5] Shalom O. and Roitman, H. and Amihood, A. and Karatzoglou A. 2018. "Collaborative Filtering Method for Handling Diverse and Repetitive User-Item Interactions". Proceedings of the 29th on Hypertext and Social Media (HT 18). Pp. 43-51. https://dl.acm.org/citation.cfm?id=3209550

[6] Agarwal, D. and Chen, B. 2016. Statistical Methods for Recommender Systems. Cambridge University Press.New York, NY, USA.https://dl.acm.org/citation.cfm?id=3019548

[7] Yu, H.-F. and Bilenko, M. and Lin, C.-J. 2016. "Selection of Negative Samples for One-class Matrix Factorization". Technical report, National Taiwan University. https://www.csie.ntu.edu.tw/~cjlin/papers/one-class-mf/biased-mf-sdm-with-supp.pdf

[8] Volkovs, M. and Yu, G. and Poutanen, T. 2017. "DropoutNet: Addressing Cold Start in Recommender Systems". Advances in Neural Information Processing Systems 30 (NIPS 2017). https://papers.nips.cc/paper/7081-dropoutnet-addressing-cold-start-in-recommender-systems

User and Context Aware Composite Item Recommendation

Krista Drushku
SAP
Paris, Levallois-Perret
krista.drushku@[sap.com,etu.univ-tours.fr]

Alexandre Chanson
University of Tours
Blois, France
alexandre.chanson@etu.univ-tours.fr

Ben Crulis
University of Tours
Blois, France
ben.crulis@etu.univ-tours.fr

Nicolas Labroche
University of Tours
Blois, France
nicolas.labroche@univ-tours.fr

Patrick Marcel
University of Tours
Blois, France
patrick.marcel@univ-tours.fr

ABSTRACT

We introduce a novel Composite Item Recommender algorithm named BFCM in a Business Intelligence application to provide users with customized recommendations to complete their reporting Tasks. To this extent, we propose a complete pipeline from the analysis of previous reports to the discovery of user intents to context-aware recommendations of Composite Items completing a report. Reported experiments show the importance of user profile in recommendation of composite items and the robustness of the proposed solution to the quality of the the user profile.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Clustering*; *Business intelligence*;

KEYWORDS

Recommendation, Composite items, Bundles, Clustering, Business Intelligence

ACM Reference Format:
Krista Drushku, Alexandre Chanson, Ben Crulis, Nicolas Labroche, and Patrick Marcel. 2018. User and Context Aware Composite Item Recommendation. In *Proceedings of ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios.* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

In today era of personal assistants [12] and with the growing need for intelligent data analysis tools that make data exploration less tedious, traditional search systems generally reach their limits for two main reasons.

The first reason relates directly to the format of the recommendation itself that is generally a ranked list of items: in this case the relevance of each item to the query is evaluated independently of the other items in the list and all items are treated equally. This kind of recommender may thus be inefficient in contexts where the list should be considered as a sequence and where the rank should reflect a relevance allowing to grasp a process, like in database exploration [1, 10], e-learning [9], or to recommend different types of items like in tourist itinerary planning [6] or finally to recommend complementary and diverse items at once like text, images and videos in web search engines [14]. In all the aforementioned cases, users expect more complex structures, that are called **composite items** or **bundles** [3, 4, 8]. These bundles group representative, cohesive, but still diverse and novel suggestions [16, 17], coming from different sources, possibly with several objectives in mind, and with distinct types of items, which makes the overall recommendation process more complex. Recent works in composite retrieval propose methods for building bundles of items around some central verticals as BOBO [4] or CPS [8] or as subset of clusters [2].

The second reason is that most of the traditional recommendation approaches do not introduce the context of the query or the user intent. Several works have been conducted to provide user intent model for recommendation, notably in the context of intelligent personal assistant [12], or for the recommendation of queries in the context of data exploration [11]. [2, 5, 15] are the first noticeable works to explicitly introduce a user intent term in a bundle recommendation process. In order to seamlessly compare items and users, the proposed method projects all items and users in a vector space of types [15] or topics [2], provided respectively by the item metadata or an LDA algorithm [7]. At the heart of the method is a constrained fuzzy c-means (FCM) algorithm that builds bundles around cluster centroids using a greedy function that aggregates several constraints like cohesiveness, personalization, diversity, etc. [3].

In this paper, we tackle the problem of completing Web Intelligence documents[1], each composed of several reports, with visualized queries, which asks to recommend items that are both conform to user interest and complementary to the current report. More precisely, the Web Intelligence platform of SAP aims at helping users constructing or completing their reports with queries already designed and shared by their colleagues. This way, they complete their reports faster, the existing reports become reusable and new information retrieved from the databases eventually become quickly visible. Our objective is then to group queries coming from different documents, that all together bring more information than a ranked list of independent queries.

To this aim, we propose an improved version of the work by [2] that introduces two new penalty terms related with i) the relevance to the user short-term interest and ii) the order of the queries in

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, October 7, 2018, Vancouver, Canada.

[1] https://help.sap.com/viewer/c95594c101a046159432081ca44d6b18/4.2.3/en-US/

the bundle. As we work with particular dynamic documents of the Web Intelligence platform, the user short term interest is its actual context, represented by the current report opened to edit. It completes the user long-term interest defined over their past actions. A user interest consists on a package of queries responding to the same user need. It is important to notice that because of the richness and the diversity of criteria used to build our bundles, more straightforward composite items recommendation algorithms like BOBO [4], that focuses on bundle cohesiveness, or CPS [8], that favors diversity among the items in the bundle, cannot reach the trade-off that we are looking for.

Similarly to [2], we define a vector space specifically tailored for this use case that allows to compute distances between items, between items and user and between items and report. Contrary to previous works, topics are discovered performing an efficient fuzzy k-medoid [13] over the past queries to learn global intents for all users. A proper metric is then learned on this vector space following the same methodology as presented in [11].

The paper is organized as follows: Section 2 introduces the definition of our user intents (or topics) space, how we define it and how to build a metric on this space. Section 3 formalizes our problem and describes the objective function of our modified fuzzy c-means algorithm. Finally Section 4 proposes some experiments on real data from the Web Intelligence platform at SAP, that show the importance of considering short and long term user profiles as well as an order on the queries in our context and Section 5 concludes and opens futures directions of research.

2 USER INTENT DISCOVERY

The Web Intelligence platform includes three elements that play an important role in the bundle construction: (i) the queries we can recommend, (ii) the user to whom we recommend and (iii) a report to complete, which represents a short-term interest. Each of these elements represents an *entity* of our platform which have to be projected in the same vector space to build the bundles, similarly to [2, 3, 5].

Definition 1. Let e be an *entity* and $I = \langle I_1, \ldots, I_N \rangle$ a set of N user intents or topics. An *entity profile* $I^e = \langle I^e_1, \ldots, I^e_N \rangle$ is a vector of weights $I^e_j \in [0, 1], \forall j \in [1, N]$, and such that $\sum_{j=1}^{N} I^e_j = 1$, defined over the set of intents. $I^e = \langle I^e_1, \ldots, I^e_N \rangle$ represents the relative importance of each intent $I_j, j \in [1, N]$ for a given entity e.

User intent space discovery as a clustering problem. Our objective is thus to define such user intent space that could represent the main information contained in the past queries. In [11], the authors identify user intents in the context of data exploration using a hard clustering algorithm on query representation. In our case, it can be observed in Definition 1 that the relation between the queries and the user intent space is more gradual. Indeed, I^e_j can be seen as the membership of an entity e to a user intent I_j. In this context, we use an efficient fuzzy k-medoid algorithm (FCMD) [13] to discover the user intents based on past queries. This algorithm needs a metric between queries to operate. As presented in Table 1 and following the methodology defined in [11] we define a set of features and their associated distance measures for each query: 3 features are built on the queries metadata (*same universe*, *same user*, *same folder*) and 2 others use topics discovered using LDA [7] over the query parts, defined in [11], or report and document titles, similarly to LDA topics in [5].

The overall distance $Dist(q_1, q_2)$ between queries q_1 and q_2 is defined as a linear combination of distance d_f for each specific feature f from the set of all features F as follows:

$$Dist(q_1, q_2) = \sum_{f \in F} \lambda_f \times d_f(q_1, q_2) \tag{1}$$

The learning of the appropriate metric corresponds to the learning of the weights λ_f. To this aim, we rely on the queries of labeled pairs of documents by SAP experts, who have judged for each pair if they represent the same intent or not. We train a Linear SVM classifier to learn the relative importance λ_f of each feature f.

Projection of queries, user and reports in user intent space. As per the definition of entity profile, a query, a report and a user profile can be represented by a weighted vector of importance of each interest I_j.

- Knowing the user intents as the clusters produced by the FCMD algorithm and the medoid of each cluster, it is possible to directly compute a membership vector for each new query based on the metric.
- A report r is basically a set of queries Q^r. It is thus possible to compute the coordinates of a report in the user intent space by averaging the coordinates of its queries as follows:

$$I^r = \frac{1}{|Q^r|} \sum_{q \in Q^r} I^q \tag{2}$$

- A user u can also be represented by the set of their previous queries Q^u. However, we take into account the frequency f_q of each query q in their past history as follows:

$$I^u = \frac{1}{\sum_{q \in Q^u} f_q} \sum_{q \in Q^u} I^q \times f_q \tag{3}$$

Table 1 details each feature used to compare two entities and the weights attributed by Linear SVM.

Feature f	Weight λ_f	Distance
same universe	0.24	MaxFrac.
same user	0.38	MaxFrac.
same folder	0.49	NormInt.
LDA query Parts	-0.56	Cosine
LDA titles	0.25	Cosine

Table 1: Query features description. Distance relates to metrics defined in [11]. Weights with the highest absolute score correspond to the most decisive features. 'same user' and 'same folder' favor the grouping of queries into the same user interest while 'LDA query parts' tend to discriminate among user intents.

3 COMPOSITE ITEM CONSTRUCTION

We can formulate the problem of building bundles as an optimization problem, that firstly aims at finding representative summaries of items and secondly selects the group of items respecting several constraints, to assure the relevance to the user profile, complementary and cohesion of this package. *Representativeness* assures that

each bundle is built around a representative item of the dataset in order to cover the whole input data and it is ensured by applying a FCM over the items. Each cluster k is represented by a centroid c_k, which is projected in the same N-vector space, uniformly to the profile of all other entities. More precisely, given the set of items $|Q|$, the FCM algorithm returns K centroids and a partition matrix $M = \mu_{i,j} \in [0,1], i \in [1,|Q|], j \in [1,K]$ with $\sum_{k=1}^{K} \mu_{ik} = 1$ where each $\mu_{i,j}$ represents the degree to which a query i belongs to the cluster j.

We complete the objective function of [2] with new constraints to better fulfill user expectations. The only hard constraint we should respect for the creation of bundles is the number of queries they should contain: 5 in our use case of Web Intelligence reports, concluded by the experts as the adequate number of queries to recommend and easy to integrate in the interface of the existing BI platform.

The implementation of the penalty terms is different from the previous studies [2, 3, 5], as we work with different data. We define a distance function $dist()$, which measures the distance between entity profiles in the projected user intents space and a diversity function $div()$ that estimates the gain new items added in the bundle bring by presenting new visualization types.

Definition 2. Let $e1, e2$ be two different entities and I^{e_1} and I^{e_2} their corresponding profiles projected in the vector space, we define the function $dist$ of these entities as the squared Euclidean distance between their profiles:

$$dist(e_1, e_2) = \sum_{j \in [1,N]} (I_j^{e_1} - I_j^{e_2})^2 \tag{4}$$

Definition 3. Let r be the report to complete, B be the candidate bundle composed of Q^B queries and $viz(Q^B)$ the group of their visualizations. We define H_n as the normalized entropy function calculated over the visualization types of bundle and current report queries as follows:

$$div(Q^B \cup_B Q^r) = 1 - H_n(viz(Q^B \cup_B Q^r)) \tag{5}$$

Diversity was already explored by Amer-Yahia et al. in previous studies [2]. We differ in the implementation of diversity, which is computed using the type of query visualization (i.e. Pie chart, Graph, etc), assuring an orthogonal space different from other constraints.

Objective Function. Given a user u, an actual report r and the set of queries of all users Q, we aim to find (i) a set of K fuzzy clusters $C = \{C_1, ..., C_K\}$ of queries in Q, (ii) a membership function μ indicating the membership of each query to each cluster, and (iii) a set of K bundles $B = \{B_1 ... B_K\}$ with $B_k \in C_k$, $\#B = |B_k|$, which minimizes the following function:

$$\arg\min \ \frac{\alpha}{|Q|} \sum_{i=1}^{|Q|} \sum_{k=1}^{K} \mu_{ik}^m dist(q_i, c_k) \ +$$

$$\frac{1}{K} \sum_{k=1}^{K} \left(\frac{\beta}{\#B} \sum_{q \in B_k} dist(q, c_k) \ \textcircled{1} + \right.$$

$$\frac{\gamma}{\#B} \sum_{q \in B_k} dist(q, u) \ \textcircled{2} \ + \delta div(Q^{B_k} \cup_B Q^r) \ \textcircled{3} +$$

$$\left. \frac{\rho}{\#B} \sum_{q \in B_k} dist(q, r) \ \textcircled{4} \ + \omega \sum_{i=1}^{|Q^{B_k}|-1} dist(q_i, q_{i+1}) \ \textcircled{5} \right)$$

where ① ensures the bundle $uniformity$ minimizing the distance of queries of the bundle to the center of cluster, ② guarantees that the suggested queries are *personalized* and correspond to the user long-term interests while ④ guarantees that they are *relevant* to the short-term interests, corresponding to the report they will be added in, ③ ensures the $diversity$ of query visualizations and ⑤ enforces a logical *ranking* of items, ensuring the proximity between two consecutive queries in the bundle.

This definition of a minimization problem, using a distance measure can be changed to use a Similarity-based formulation by replacing the distance with a similarity measure, as Cosine Similarity for example, and *argmin* by *argmax*.

This problem of constructing composite items reduces to the algorithm described in [2], following the standard FCM membership update and the modified centroids update rule and simply extending the greedy selection heuristics used in bundle composition to introduce our new penalty terms.

4 EXPERIMENTS

This section presents a set of experiments that illustrate the importance of considering the short and long terms interests in recommending for a final user and the significant weight of modeling a good user profile. We test their impact in recommending qualitative bundles. Due to space limitations, we limit the experimentation to a simple protocol with only a few settings for the objective function hyper parameters.

Data preparation. We use a selected set of 194 combinations of recent reports viewed by 46 users, containing more than 6 queries. They are separated in two parts: $future$ composed of the 5 last queries and *seed* containing the remaining queries of the beginning of the report. We run our algorithm over the *seed* and we try to recommend items close to the $future$, that follow the same logical ordering as well.

Experimental protocol. We evaluate our bundles in terms of precision and recall, comparing to the expected $future$. As it is unlikely that a query appears in several reports, we consider a similarity threshold, as defined in [1], above which two recommended queries are considered identical and the recommendation successful. We compare our algorithm to an adapted version of BOBO, based on our distance between entities, but that is agnostic of any user model or ordering of the items. We have used the same combination of hyper parameters for all our experiments: $\alpha = 0.1$, $\beta = 0.3$ and $\gamma = 0.3$, $\rho = 0.3$. Diversity δ and ranking ω are set to 0 unless otherwise stated.

Evaluating user constraint. We simulated an ideal user profile and degraded it with random noise, modifying the scores of membership to the learned intents. The ideal user profile is generated using the $future$ queries that should be discovered and the report profile is generated based the queries of the *seed*. We expect this setting to provide the highest precision score. To compare with a real context,

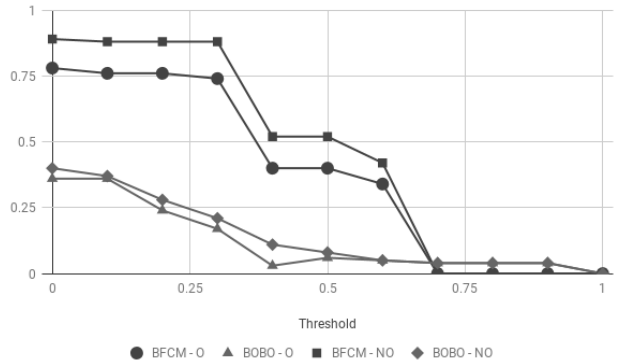

Figure 1: BOBO vs Constrained FCM

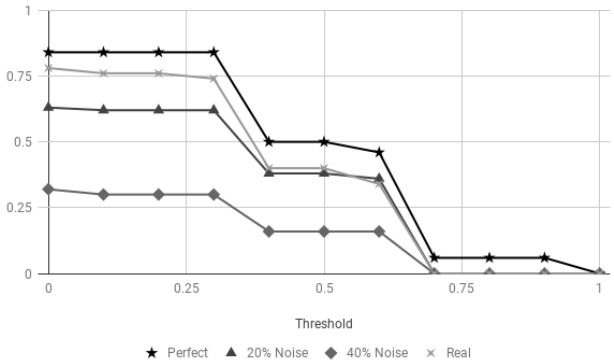

Figure 2: Precision with ordered measure

we learn the user profile as presented in Equation 3, based on the profiles of the queries she has consulted in the past.

In Figure 2 we compare the precision at different thresholds of prediction using respectively a perfect user profile, degraded with 20% noise, 40% noise and a real user profile extracted from the user's previous query usage.

Evaluating order constraint. As only our algorithm takes into account the potential order of items, we make two comparisons of the result to the $future$: (O) an ordered measure, where the n^{th} query of the recommendation is only compared to the n^{th} query of the $future$ to compute precision and (NO) an unordered measure, where we test each combination of pairs (predicted, expected) and keep the highest score.

5 RESULTS AND CONCLUSIONS

Importance of user profile. Results presented in Figure 1 show that BOBO, that is agnostic of user constraint, performs worst in this experiment compared to the real user profile as computed from user past history.

Robustness to user profile quality. Figure 2 shows that the best the quality of the user profile fed to our algorithm, the more relevant items are recommended. The perfect user profile allows very good recommendations for low similarity threshold, while the noise degrades the performances as expected. According to this test, it is possible to observe that our real user profile corresponds to approximately 20% of noise in the user profile. This is due to the lack of information in the query log used for this experiment.

Ordering items inside a bundle. As it can be seen in the Figure 1 for our algorithm BFCM, the order of items we recommend is close to the order of the expected queries as shown by the proximity of the plots for BFCM-O (ordered) and BFCM-NO (non-ordered).

Future work. We conducted several tests with different sets of hyper parameters, notably for ranking and diversity. However using the aforementioned precision measure we were unable to conclude on the contribution of the ranking as different ranking did not impact precision. This calls for a more subjective measure of quality of the bundle, that would be able to transcribe to which extent the bundle was beneficial for the user.

REFERENCES

[1] J. Aligon, E. Gallinucci, M. Golfarelli, P. Marcel, and S. Rizzi. 2015. A collaborative filtering approach for recommending OLAP sessions. *DSS* 69 (2015), 20–30.
[2] M. Alsaysneh, S. Amer-Yahia, E. Gaussier, V. Leroy, J. Pilourdault, R. M. Borromeo, M. Toyama, and J. Renders. 2017. Personalized and Diverse Task Composition in Crowdsourcing. In *Proceedings of the IEEE Transactions on Knowledge and Data Engineering, TKDE 2017.*
[3] S. Amer-Yahia, F. Bonchi, C. Castillo, E. Feuerstein, I. Méndez-Díaz, and P. Zabala. 2013. Complexity and algorithms for composite retrieval. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, Companion Volume.* 79–80.
[4] S. Amer-Yahia, F. Bonchi, C. Castillo, E. Feuerstein, I. Méndez-Díaz, and P. Zabala. 2014. Composite Retrieval of Diverse and Complementary Bundles. *IEEE Trans. Knowl. Data Eng.* 26, 11 (2014), 2662–2675.
[5] S. Amer-Yahia, É. Gaussier, V. Leroy, J. Pilourdault, R. M. Borromeo, and M. Toyama. 2016. Task Composition in Crowdsourcing. In *2016 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016, Montreal, QC, Canada, October 17-19, 2016.* 194–203.
[6] I. Benouaret and D. Lenne. 2016. A Composite Recommendation System for Planning Tourist Visits. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2016, Omaha, NE, USA, October 13-16, 2016.* 626–631.
[7] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3 (March 2003), 993–1022.
[8] H. Bota, K. Zhou, J. M. Jose, and M. Lalmas. 2014. Composite retrieval of heterogeneous web search. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014.* 119–130.
[9] S. Changuel, N. Labroche, and B. Bouchon-Meunier. 2015. Resources Sequencing Using Automatic Prerequisite-Outcome Annotation. *ACM TIST* 6, 1 (2015), 6:1–6:30.
[10] M. Djedaini, N. Labroche, P. Marcel, and V. Peralta. 2017. Detecting User Focus in OLAP Analyses. In *Advances in Databases and Information Systems - 21st European Conference, ADBIS 2017, Nicosia, Cyprus, September 24-27, 2017, Proc.* 105–119.
[11] K. Drushku, J. Aligon, N. Labroche, P. Marcel, V. Peralta, and B. Dumant. 2017. User Interests Clustering in Business Intelligence Interactions. In *Advanced Information Systems Engineering - 29th International Conference, CAiSE 2017, Essen, Germany, June 12-16, 2017, Proceedings.* 144–158.
[12] R. Guha, V. Gupta, V. Raghunathan, and R. Srikant. 2015. User Modeling for a Personal Assistant. In *WSDM.* Shanghai, China., 275–284.
[13] R. Krishnapuram, A. Joshi, O. Nasraoui, and L. Yi. 2001. Low-complexity fuzzy relational clustering algorithms for Web mining. *IEEE Trans. on Fuzzy Systems* 9 (2001), 595–607.
[14] M. Lalmas. 2017. "Engage moi": From retrieval effectiveness, user satisfaction to user engagement. In *17ème Journées Francophones Extraction et Gestion des Connaissances, EGC 2017, 24-27 Janvier 2017, Grenoble, France.* 1–2.
[15] Vincent Leroy, Sihem Amer-Yahia, Éric Gaussier, and Seyed Hamid Mirisaee. 2015. Building Representative Composite Items. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015.* 1421–1430.
[16] Maximilien Servajean, Reza Akbarinia, Esther Pacitti, and Sihem Amer-Yahia. [n. d.]. Profile Diversity for Query Processing using User Recommendations. *Inf. Syst.* 48 ([n. d.]), 44–63.
[17] Cong Yu, Laks V. S. Lakshmanan, and Sihem Amer-Yahia. 2009. It takes variety to make a world: diversification in recommender systems. In *EDBT 2009, 12th International Conference on Extending Database Technology, Saint Petersburg, Russia, March 24-26, 2009, Proceedings.* 368–378.

Recommendations for repeat consumption products considering users' tendency to repeat

Zhang Qian, Koki Nagatani, Masahiro Sato, Takashi Sonoda, Tomoko Ohkuma
Fuji Xerox Co., Ltd
Yokohama, Kanagawa, Japan
{qian.zhang, nagatani.koki, sato.masahiro, takashi.sonoda, ohkuma.tomoko}@fujixerox.co.jp

ABSTRACT

In the modern recommender system, most research has been focused on how to recommend unconsumed products that the customers have not previously purchased. However, in some domains, such as grocery shopping and music screaming services, customers consume not only unconsumed but also previously consumed items. In the re-consumption behavior, users' preferences for re-consumption are different. Some customers tend to consume what they have always consumed instead of trying new items. On the other hand, other customers tend to like trying unconsumed items. In this paper, we extend a conventional approach to generate repeat consumption recommendations incorporating the users' tendency to repeat. Furthermore, we use real-life retailer data for evaluation and our experimental results show that the proposed method outperforms existing methods.

KEYWORDS

repeat purchase, personalized recommendation

1 INTRODUCTION

Most of the modern recommender systems have concentrated on recommending unconsumed items to users. The reason is that the early recommender systems have been evolved in the domains of movies [2,1], books [5], news [2]. Generally, users in these domains do not re-consume what they have consumed before. The recommendation of these fields helps to encourage users to reuse services by recommending unconsumed items for them.

However, in other domains, such as grocery shopping and music streaming services, people consume not only unconsumed but also previously consumed items. As time passes people forget things [8]; therefore, it is possible that consumers cannot recall items which they have consumed and liked in the past. On the other hand, while doing their shopping, people stand a good chance of inadvertently forgetting some products which they need and always purchase. Thus, in these domains, it is helpful to recommend repeated consumption as well as unconsumed items, that can help to generate more personalized recommendations.

Recently, new studies related to the recommendation of repeat consumption items have appeared [4,9,7,3]. According to these studies, the recommender systems can reasonably recommend not only previously purchased but also unconsumed items for customers. Sommer et al. [4] proposed a novel method by integrating repeated interaction data directly into a matrix factorization (MF). This model represents users' preferences based on users' purchase and repeat purchase logs. Both types of logs contribute equally to users' preferences.

Nevertheless, according to the repeat consumption behavior, consumers' preferences for repeat purchases, namely the users' tendency to repeat, are different. For instance, Mary does not want to buy an item she has never bought before, so she always buys the same milk, bread or vegetables. David, on the other hand, is easily tired from the same food, so he likes to try various kinds of products. It is more reasonable to recommend more repeated purchase products to people like Mary, who frequently consume previously purchased items. Contrarily, it makes sense to recommend relatively few repeated purchase products to people like David, who tend to seldom consume again what they have previously purchased. Therefore, the users' repeat purchase logs should not be regarded as equal to the users' purchase logs across different users.

Accordingly, in this paper, we extend the existing method by weighting repeat consumption. We aim at improving the performance of the conventional recommendation method for repeated consumption items by taking into account the users' tendency to repeat. Furthermore, we use real-world retailer data for the empirical evaluation.

The key contributions of the paper are:

- We extend the existing repeat consumption recommendation method by integrating the users' tendency to repeat to generate recommendations for repeated consumption products.
- We conduct experiments using a real-world dataset to verify the performance of our extension model.

2 RELATED WORK

Recently, the trend to propose recommendation methods encouraging users' repeat behavior has been growing. Among these studies, there are two main kinds of repeat behavior: repeat visits [6,1,10] to the same shop (*e.g.*, ecommerce sites, music/movie streaming services, and retail stores) and repeat consumption of the same items [4,9,7,3]. The former has been addressed previously, while the latter has been researched

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, Vancouver, Canada.

recently. In this paper, we focus on the latter in order to recommend repeated consumption items.

One type of recommendation concerning repeat behavior is analyzing customer behavior and predicting revisits. There is numerous research on this topic. For example, Liu et al. [6] described how to generate various types of metrics from customer activity data and used them to generate repeat buyer predictions. Anderson et al. [1] found that the recency and quality of consumption are the strongest predictors of revisits; thus, they proposed a hybrid model that combines the two indicators to generate predictions. Du et al. [10] proposed a novel convex formulation to predict the time of the users' next return to services.

The other type of recommendations concerning repeat behavior is the repeated consumption item recommendation. Lerche et al. [9] extended the Bayesian Personalized Ranking introducing the idea that the recurring consumption on an item might be an indicator of stronger preferences than a single purchase. Dimyati et al. [7] recorded if a certain product was purchased up to three times and used an item-based Collaborative Filtering (CF) algorithm to address the repeated consumption recommendation problem. Similarly, a repurchase-based CF technique was proposed to make recommendations for unpurchased and repurchased items in [3]. The method extended the traditional user-based CF algorithm by incorporating item-repurchase probabilities. Sommer et al. [4] introduced latent factors of repeated consumption items into MF.

In our study, we attempt to contribute to the latter type of repeated consumption studies. The conventional methods of repeated consumption recommendations do not take into account users' tendency to repeat. Therefore, we attempt to extend the existing method in order to improve its recommendation performance by considering the customers' tendency to repeat.

3 MODEL

3.1 Conventional Model

Some of the successful realizations of latent factor models are based on MF, which characterizes both the user and the item by a vector of factors. The item receives a rating or 0/1 indicating whether a user purchased. The SVD++ approach [13] is one of the MF's popular extensions. The model assigns user u's rating r_{ui} to item i as follow:

$$\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^{\mathrm{T}}(p_u + |\mathrm{N}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}(u)} y_j), \quad (1)$$

where q_i and p_u are k-dimensional vectors of latent factors with respect to the items and users, μ is the overall average rating, b_u and b_i are the biases of user u and item i, respectively. $\mathrm{N}(u)$ denotes the set of items that the user has interacted with. A user who showed a preference for items in $\mathrm{N}(u)$ is characterized by the vector $\sum_{j \in \mathrm{N}(u)} y_j$.

Recently, based on the SVD++ approach, a method to directly integrate repeated interaction data into a MF was proposed. Sommer et al. [4] called it PRMF and the model follows Equation 2.

$$\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^{\mathrm{T}} \left(p_u + |\mathrm{N}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}(u)} y_j + |\mathrm{T}(u)|^{-\frac{1}{2}} \sum_{t \in \mathrm{T}(u)} x_t\right). \quad (2)$$

This model further extended the user-specific factor vector of SVD++ by adding a repeated interaction parameter x_t. Here, the set of repeated interaction items $\mathrm{T}(u)$ consists of items that the user has interacted with more than once. The items in $\mathrm{T}(u)$ are also included in $\mathrm{N}(u)$.

3.2 Our Extension Model

As described in the introduction of our paper, we introduce the users' tendency to repeat into the conventional model. Our new approach follows the following equation:

$$\hat{r}_{ui} = \mu + b_u + b_i + {q_i}^{\mathrm{T}} \left(p_u + |\mathrm{N}(u)|^{-\frac{1}{2}} \sum_{j \in \mathrm{N}(u)} y_j + {R_u}^{\beta} |\mathrm{T}(u)|^{-\frac{1}{2}} \sum_{t \in \mathrm{T}(u)} x_t\right), \quad (3)$$

where R_u defines the users' tendency to repeat. It is calculated as the ratio of user u's number of repeat consumption products and the number of total consumption products over a period of time. To explicitly take into account the users' tendency to repeat, we use R_u to weight the repeat interaction parameter x_t. β is a hyper-parameter which is used to adjust the users' tendency to repeat. The prediction rating $\hat{r}_{ui}$ signifies user u's preference for item i. Using R_u to weight the repeated interaction parameter x_t, this model can adjust the balance of the parameters y_j and x_t. For users with a large R_u, the model recommends more repeated consumption items; contrarily, for users with a small R_u, the model recommends few repeated consumption items. We obtain the parameters μ, b_u, b_i, q_i, p_u, y_j, and x_t by training and compute $\mathrm{N}(u)$, $\mathrm{T}(u)$, and R_u by aggregating.

4 EXPERIMENTS

In this section, we first introduce our experimental setting and evaluation metrics. Then we conduct the experiments to evaluate our model's performance in comparison with the conventional models.

4.1 Dataset and Setting

In the experiment, we used the Tafeng[1] dataset which consists of over 4 months of transactions from a Chinese grocery store. Due to its sparsity, we filtered out users with less than 10 days of purchase history and items with less than 20 days of purchase history. After cleaning the data, the dataset contained 1,803 users and 160,134 interactions for 6,128 items.

Every 2 weeks of the last 6 weeks were used for training, validation, and testing, respectively. We constructed 8 weeks before each 2-weeks period to aggregate the users' purchased items $\mathrm{N(u)}$, repurchased items $\mathrm{T(u)}$, and to calculate users' tendency to repeat $\mathrm{R_u}$. We utilized the validation set to obtain the hyper-parameters and the test set to evaluate the performance of

[1] http://recsyswiki.com/wiki/Grocery_shopping_datasets

our model. We set the number of latent factors to a 1,000. We randomly chose 100 items as negative samples. Throughout our experiment, we found that our model performs well when the hyper-parameter β is set to 0.1.

4.2 Learning Approach

We applied the stochastic gradient descent approach to obtain the parameters. An item was assigned a rating of 1 if the user had purchased it, 0 otherwise, and the 2 weeks training data was treated as our correct label. We trained our model so that $\sigma(\hat{r}_{ui})$ approximated the correct label. The log loss function was used as the cost function.

4.3 Evaluation Metrics

The top-*N* product recommendation would be generated for a given active customer. We set *N*=10 for its real-world practicality. To verify the performance of our method, we compared it with the SVD++ approach described in Eq.1 and the PRMF model introduced in Eq.2. We used three evaluation metrics: Precision@*N*, Recall@*N*, and nDCG@*N*.

4.4 Experimental Results

First, we compared our extension model with the conventional methods. The results are shown in Table 1.

Table 1: Test set performance on the Tafeng dataset. The best performances are printed in bold.

Evaluation metrics	SVD++	PRMF	Our model
Precision@10	0.0729	0.0740	**0.0742**
Recall@10	0.0808	0.0820	**0.0831**
nDCG@10	0.0979	0.1019	**0.1028**

Comparing the three models, our model resulted in the best evaluation metrics. The results show that the recently proposed PRMF indeed had a better performance than the SVD++. By extending the PRMF, our method which incorporated the users' tendency to repeat consumption achieved further improvement.

Moreover, to better present the characteristic of our model, we analyzed how our model applies to different types of users. As we focus on recommending repeated consumption products in this paper, we classified users into 5 groups based on their tendency to repeat (R_u) ignoring the users whose R_u were 0. Based on the precision@10 metric, we compared the performance of our extension model, the SVD++, and the PRMF for each group. Users in Group 1 had the lowest tendency to repeat, while users in Group 5 had the highest. The results are shown in Table 2.

The second column of Table 2 shows the range of $\mathrm{R_u}$ within each group. For users with low $\mathrm{R_u}$ (Group 1,2), our model and the SVD++ approach performed better than the PRMF model, while for users with high R_u (Group 3,4,5), our model and the PRMF model performed better than the SVD++ approach. The users in

Table 2: Comparison of the precision@10 metric across groups of users with different tendency to repeat ($\boldsymbol{R_u}$).

Groups	R_u range	SVD++	PRMF	Our model
1	(0, 0.04]	0.0532	0.0481	0.0519
2	(0.04, 0.08]	0.0569	0.0554	0.0571
3	(0.08, 0.12]	0.0669	0.0684	0.0690
4	(0.12, 0.16]	0.0930	0.1019	0.0981
5	(0.16, 1]	0.1236	0.1250	0.1260

Group 1,2 buy more previously unconsumed items than the users in Group 3,4,5. Thus, the prediction performance might improve if we recommend more unconsumed items to Group 1,2 and more repeated consumption items to Group 3,4,5. Our model attempted to achieve the above by adjusting the balance of parameter y_j (the users' preference from the purchase logs) and parameter x_t (the users' preferences from the repeated purchase logs). The SVD++ approach creates recommendations solely based on parameters from the purchase logs; therefore, it performs poorly for users with higher R_u (Group 3,4,5). On the other hand, the PRMF model assigns the same weights for the parameter y_j and parameter x_t across all users, ignoring the users' tendency to repeat. The model might recommend too many repeated consumption items that leads to a poor prediction performance for users with lower R_u (Group 1,2). Our method, which takes advantage of the users' tendency to repeat by weighting repeat interactions, recommended more suitable items to both customers who frequently consume and who seldom consume preciously consumed items.

5 CONCLUSION

In this paper, we extended a conventional approach to generate repeat consumption recommendations by accounting for the users' tendency to repeat. The experimental results showed that our extension model outperforms comparative recommendation methods. It can recommend more suitable items to both users with high tendency to repeat and users with low tendency to repeat.

In our next project, we plan to investigate the performance of our proposed model using other datasets or evaluate our model in an online environment. On the other hand, since users' tendency to repeat are also different depend on categories, for example, some consumers prefer to repeatedly purchase food but tend to try different shampoos or laundry detergents, thus we are interested to take into account this phenomenon to our model as well.

REFERENCES

[1] A. Anderson, R. Kumar, A. Tomkins and S. Vassilvitskii. 2014. The Dynamics of Repeat Consumption. In *Proceeding of the 23rd International Conference on World Wide Web* (WWW'14). ACM, New York, NY, USA, 419–430.

[2] B. Sarwar, G Karypis, J Konstan and J Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *Proceeding of the 10th international conference on World Wide Web* (WWW'01). ACM, New York, NY, USA, 285–295.

[3] C. S. Yang and C. H. Chen. 2017. Considering Consumers' Repeater Consumption Behaviors in Collaborative Filtering Recommendation. In *Proceeding of the 21th Pacific Asia Conference on Information System* (PACIS'17).

[4] F. Sommer, F. Lecron and F. Fouss. 2017. Recommendation system: The case of repeated interaction in Matrix Factorization. In *proceedings of the International Conference on Web Intelligence* (WI'17). ACM, New York, NY, USA, 843–847.

[5] G. Linden, B. Smith and J York. 2003. Amazon.com Recommendations Item-to-Item Collaborative Filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.

[6] G. Liu, T. T. Nguyen, G Zhao, W. Zha, J. B. Yang, J. Cao, M. Wu, P. L. Zhao and W. Chen. 2016. Repeat Buyer Prediction for E-Commerce. In *Proceeding of the 22th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD'16). ACM, New York, NY, USA, 155–164.

[7] H. Dimyati and R. Agasi. 2018. Collaborative Filtering in an Offline Setting Case Study: Indonesia Retail Business. AusDM 2017. In *Data Mining*, Y. L. Boo, D. Stirling, L. H. Chi, L. Liu, K. L. Ong and G. Williams (Eds.). Communications in Computer and Information Science, Vol. 845. Springer, Singapore, 223–232.

[8] L. Averell and A. Heathcote. 2011. the Form of the Forgetting Curve and the Fate of Memories. *Journal of Mathematical Psychology* 55 (2011), 25–35.

[9] L. Lerche and D. Jannach. 2014. Using Graded Implicit Feedback for Bayesian Personalized Ranking. In *Proceeding of the 8th ACM Conference on Recommender System* (RecSys'14). ACM, New York, NY, 353–356.

[10] N. Du, Y. C. Wang, N. He and L. Song. 2015. Time-Sensitive Recommendation from Recurrent User Activities. In *proceeding of 28th International Conference on Neural Information Processing System* (NIPS'15). MIT Press Cambridge, MA, USA, 3492–3500.

[11] P. Melville, R. J. Mooney and R Nagarajan. 2002. Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceeding of the 8th National Conference on Artificial Inte*

[12] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl. 1994. GroupLens: an open Architecture for Collaborative Filtering of netnews. In *Proceeding of the 1994 ACM conference on computer supported cooperative work* (CSCW'94). ACM, New York, NY, USA, 175–186.

[13] Y. Koren, R. Bell and C. Volinsky. 2009. Matrix Factorization Techniques for Recommender System. *IEEE Computer* 42, 8 (2009), 30–37.

Recommending Running Routes: Framework and Demonstrator

Benedikt Loepp
University of Duisburg-Essen
Duisburg, Germany
benedikt.loepp@uni-due.de

Jürgen Ziegler
University of Duisburg-Essen
Duisburg, Germany
juergen.ziegler@uni-due.de

ABSTRACT

Recommending personalized running routes is a challenging task. For considering the runner's specific background as well as needs, preferences and goals, a recommender cannot only rely on a set of existing routes ran by others. Instead, each route must be generated individually, taking into account many different aspects that determine whether a suggestion will satisfy the runner in the end, e.g. height meters or areas passed. We describe a framework that summarizes these aspects and present a prototypical smartphone app that we implemented to actually demonstrate how personalized running routes can be recommended based on the different requirements a runner might have. A first preliminary study where users had to try this app and ran some of the recommended routes underlines the general effectiveness of our approach.

KEYWORDS

Recommender Systems; Running; Route Generation; Sports

1 INTRODUCTION AND RELATED WORK

In recent years, there has been a significant increase in research on interactive technologies that support users in performing sports activities [7]. While running is one of the most popular sports, contemporary applications such as *Runtastic*, *Strava* or *Endomondo*[1] support users primarily in the process of running a route or in keeping track of activities. More advanced tools such as *TrainingPeaks* or *SportTracks*[2] focus on structuring training and creating workout plans. Features that allow generating new running routes or receiving suggestions are, however, usually not available. Typically, the only possibility is searching for routes already recorded, either by the same user or by someone else in the platform's community, that can then be run again. Yet, in many cases, runners are not only looking for routes that start and end at some location, but also satisfy e.g. length constraints or pass through specific areas, while avoiding taking the same way or leading through the same area twice. Finding routes that fulfill such requirements can be cumbersome or even impossible without adequate support. *RouteLoops*[3], for instance, allows users to generate new routes automatically, but only takes start-/end point and length into account. To find closed cycles, random walks on the map graph are performed. Other tools require users to manually search for intermediate steps, or select via-vertices automatically and connect them by means of shortest path algorithms. Still, further adaptation towards current needs and personal preferences is not supported, neither is the consideration of the user's running history or any other route property that might be of relevance for running, such as amount of elevation or which areas are passed, e.g. forests or meadows.

Generating routes is a common task referring to the traditional route planning problem. Yet, most existing research has been attributed to finding shortest paths, although other aspects have also been found relevant for people who want to follow a route: For instance, in [9], an approach for recommending emotionally pleasant walking routes within a city is presented, which however requires availability of crowdsourced data regarding attractiveness of streets. In [11], itineraries between points of interest are created. Subsequently, users can customize the suggested routes, which appeared beneficial for learning about user preferences, and thus, further personalization of recommendations. While there have been similar attempts for automobile navigation [e.g. 8, 10], research on generating routes for cycling [14] and running [5] is limited to optimization with respect to length. Beyond that, there indeed exists research on recommender systems in this area, for instance, for helping runners to achieve new personal records or pace races [2, 12, 13]. Nevertheless, although it can be difficult for users to find running routes on their own without external assistance, especially in unknown environments or when trying to find routes with certain characteristics (e.g. specific length when practicing for a race or street lighting for evening runs), there is a lack of research on supporting runners with routes that are specifically tailored for them and take all such aspects into account.

In this paper, we propose a framework that may help to generate personalized running routes. We present a prototypical smartphone app which we developed to demonstrate the effectiveness of our approach, and describe a corresponding proof-of-concept study where participants had to use this app and ran recommended routes.

2 A FRAMEWORK FOR RECOMMENDING PERSONALIZED RUNNING ROUTES

As already outlined in the previous section, standard recommender algorithms are not sufficient for creating personalized running route recommendations: Suggesting routes ran by others might be difficult due to data sparsity at the current user's location. Moreover, each runner has a different background, ranging from beginners who want to change their lifestyle and start improving their fitness to experienced runners who train for the next marathon. Accordingly, runners have different needs, preferences and goals. Besides, some might follow a training plan, such that specific requirements have to be considered with respect to the route for the next workout. Consequently, ranking a list of existing alternatives as in typical recommender situations is not an option: Instead, recommendable

[1] https://www.runtastic.com, https://www.strava.com, https://www.endomondo.com
[2] https://www.trainingpeaks.com, https://www.sporttracks.mobi
[3] http://www.routeloops.com

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, October 7, 2018, Vancouver, Canada.

items, i.e. routes, first need to be generated especially for the current user. This, in turn, requires that map data is available and preprocessed. Moreover, as in multi-criteria recommendation [1], the user's individual preferences for specific item properties, but also context data, are much more relevant than in many other cases: While location is obviously the most important information for recommending an appropriate route, attributes like elevation or street lighting together with weather conditions and time of the day also need to be considered to satisfy the runner in the end.

Recommending personalized running routes can thus be seen as a complex and challenging task. Since route generation in general is well-explored, we propose to split the recommendation process into two steps: 1) finding candidate routes, and 2) creating recommendations. In the following, we detail on these steps, the challenges involved, and explain how we address them in our framework.

2.1 Generating Candidate Routes

First, we generate candidate routes by means of the graph model derived from the underlying map data. We follow one of the approaches proposed in [5], namely the *partial shortest paths algorithm*, which has been shown fast enough for practical application: The idea is to determine a number of via-vertices in a way that the intermediate paths between them are the shortest paths of equal length, summing up to the desired route length. Fig. 1 illustrates this procedure for two via-vertices, i.e. the route forms a "triangle".

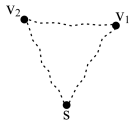

Figure 1: As in [5], we create closed routes by determining via-vertices v_1 and v_2 in a way that the length of the shortest paths between start-/end point s and v_1, v_1 and v_2, as well as v_2 and s, equals one third of the specified route length.

According to [5], this method guarantees to produce routes with a given length and only a maximum deviation, and can easily be extended to more than two via-vertices. Thus, we apply the same algorithm also with more via-vertices in order to get a more diverse set of candidates. Note that, since route generation is decoupled from creating actual recommendations, this method is interchangeable with any other algorithm that allows to find a number of cycles within a given graph, i.e. closed candidate routes in a map.

Besides constraining routes to a certain length, for later being able to recommend running routes several more aspects need to be taken into account. We have identified the following:

- Some graph edges might *not be suitable for running*, e.g. highways or closed parking lots. Consequently, these edges have to be identified and removed from the graph before applying the route generation algorithm. In case this leads to the starting vertex being in a small subgraph disconnected from the rest, a new starting point must be chosen.
- Shortest paths between vertices might *share edges*, i.e. route segments would be run twice. Also, segments of the shortest path towards a vertex x might be very *close to segments* of the path leaving x. Thus, users would, for instance, have to run on one side of a street, and return on the other. For these reasons, we introduce penalty values, which are assigned to edges already visited before. In addition, we calculate the area within the cycle that represents the route in the graph, and maximize this area to avoid long and narrow route shapes, but to create more rounded ones.
- Starting from the current location might lead to a set of candidate routes that later *do not allow to fulfill all requirements*. For instance, in case the distance to the nearest forest is more than $1/(n+1)$ of the desired route length, no route with n via-vertices will ever reach it from the original starting point. Fig. 2 illustrates two solutions: a) using a virtual starting point s_v as input for the route generation algorithm, and add the way towards and back from s_v, b) increasing the distance between s and v_1, and changing the other distances accordingly, so that the routes include more distant vertices.

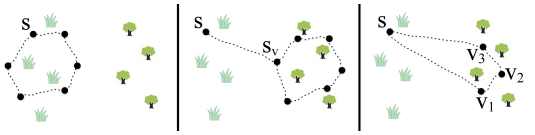

Figure 2: A route does not pass a desired area (left). As a solution, a virtual start point s_v can be introduced (center), or distances between vertices can be enlarged/reduced (right).

2.2 Creating Route Recommendations

The next step after having generated appropriate candidate routes is to rank them according to all properties that might be relevant for a runner with respect to his or her next workout. For this, we calculate scores for a number of criteria that we have identified to be important. Indeed, the following list is non-exhaustive and there might be more requirements some runners want to take into account. However, we in a first step aim at considering those that are, from our perspective, the most interesting ones, and in particular, can actually be implemented using available datasources.

(1) *Length:* Especially for experienced runners constraining the route to a specific length is very important. We use the deviation of candidate routes (derived as explained in Sec. 2.1) from the desired length to determine a score.
(2) *Uniqueness:* Maximum uniqueness of a route is reached when each edge is different from each other, i.e. runners do not have to run a certain path twice. Under the assumption that there is no meaningful reason not to maximize this value, we always try to reach a high score in this respect.
(3) *Shape:* This score is defined by the area within a route's cycle (as explained above), and should be as high as possible.
(4) *Lighting:* Runners who prefer routes with street lights might want this criterion to be considered after sunset, which is defined by the proportion of a route that is lit. This score can automatically be ignored at day time.
(5) *Elevation:* The elevation score is defined as the amount of incline and decline on a route. Having a lot of height meters largely influences a route's difficulty, which can either be seen challenging (i.e. as a special kind of training) or as an undesired property of the route.
(6) *Pedestrian friendliness:* Some ways or paths are more suitable for running than others, e.g. large streets or bikeways. The corresponding score describes the proportion of a route that is designated for pedestrians, e.g. small paths or tracks. As

special pedestrian zones raise this score as well, this may also help runners who prefer well-traveled routes.

(7) *Turns:* The number of turns a runner has to take refers to the complexity of a route. While some might see a high number to be an interesting feature, others might not desire this because it makes navigation more difficult or is inappropriate for a specific form of training (e.g. intervals). This score is calculated by means of the angles of adjacent edges.

(8) *Nature:* Running in cities can be exhausting and dangerous. Also, some runners might find it less attractive. Thus, the amount of nature is an important factor as more scenic routes may positively influence the running experience. Moreover, runners might enjoy, for instance, quietness and less air pollution. Accordingly, we introduce four different scores:
- *Trees:* Represents the proportion of a route leading through forests or segments being surrounded by trees.
- *Grass:* Represents the proportion a route goes through grass, meadows or farmland.
- *Sand:* Represents the proportion of a route crossing beaches or segments being surrounded by sand.
- *Water:* Represents whether lakes or oceans are visible, taking the distance to water into account.

(9) *History:* This score is related to the possibility of providing an opinion on routes ran in the past. Beyond the consideration of preferences expressed with respect to the aforementioned criteria, this allows us to automatically refine the suggestions to better reflect the current runner's taste. For this personalization step, similarities between items, i.e. routes, are calculated as in multi-criteria recommender systems based on all relevant item properties [1]. Then, the more similar a route, the larger the influence of the corresponding user rating (if available) on this score. For instance, if a user rated a route with lots of trees and few height meters very positive, a similar candidate route will receive a higher score.

Finally, we calculate an *overall score* for each candidate route. For this, we take the mean of the differences between the *individual scores* (as introduced above) and *desired values* for all criteria. These values can be either predefined, e.g. high for *Shape* and low for *Elevation*, set by the user initially (e.g. *Length*), or later during an interactive preference elicitation phase (e.g. *Nature*). Independent of the actual implementation of individual scores (see Sec. 3 for details on how we calculate them in our prototypical smartphone app), the overall score thus allows to rank the candidate routes.

3 THE *RUNNERFUL* APP

Runnerful is a prototypical Android app that implements our framework. We use the *OpenStreetMap API* to collect map data. Using edge annotations contained in this data (e.g. to ignore highways), we then create a graph to apply the route generation algorithm as described in Sec. 2.1. To find shortest paths between via-vertices (we use 2–4, after initial pretests), we use a modified *A* search algorithm* that penalizes nodes already visited (we vary distances and set different starting points, as described in Sec. 2.1). Scores are calculated for all criteria described in Sec. 2.2: For some criteria, such as *Length* or *Shape*, we calculate scores based on the graph data itself. For others, such as *Lighting* or *Pedestrian friendliness*, we rely on edge annotations provided by *OpenStreetMap*. For *Elevation*, we additionally send requests to the *Google Maps Elevation API*. Regarding the amount of *Nature*, we take surrounding areas and their *OpenStreetMap* annotations into account: Using a ray-casting algorithm, we determine whether segments cross forests, farmland or beaches. The proportion of edges for which this applies then defines the respective score. Moreover, we calculate the distance of every route point to areas that represent water.

As user input, the app initially only requires the desired route length. Then, taking the current GPS position, recommendations are generated. Fig. 3 shows a screenshot with two suggested routes: The user has requested routes of 4 km length. When looking at the actual values, both routes have high accuracy in this regard, which is reflected accordingly in the net diagram (dimension depicted by yellow ruler). As also shown in the net diagram, both routes go through some forest, which can easily be seen in the map (route 1 through the Zoo in the north, route 2 through the community garden in the south). Furthermore, the routes strongly differ in shape (depicted by the oval in the net diagram): While route 1 fills a large area and avoids visiting streets twice, this is very different for route 2, which has a more narrow shape with route segments close to each other or even ran multiple times. Using the arrows left and right to the net diagram, the user can scroll through the results. The buttons below allow to request a new set of recommendations, run the recommended route (which leads to a new screen for route navigation, showing workout duration and progress of the run), and critique the current recommendation.

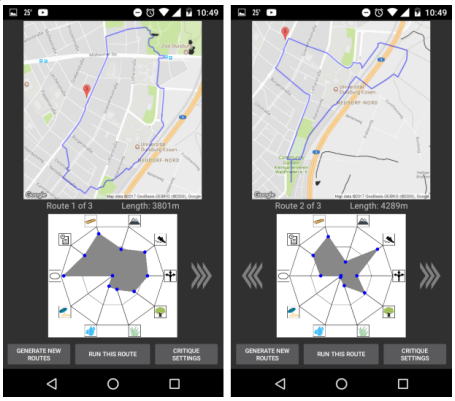

Figure 3: Two routes recommended by *Runnerful*: The user is presented with a map view as well as a net diagram showing the scores for the different criteria.

Fig. 4 shows a part of the screen for critiquing: The user can drag criteria he or she wants to be considered less or more into the respective areas. This decreases or increases the *desired values* used to calculate the *overall scores* of the candidate routes.

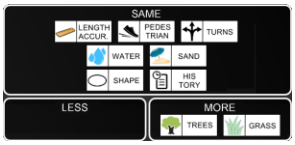

Figure 4: The user is critiquing the recommended route.

After finishing a run, the user can express his or her opinion by rating the route. This rating then influences the *History* score as described in Sec. 2.2 to give more personalized recommendations.

4 EVALUATION

We conducted a first user study as a proof-of-concept for the application of our framework. We recruited 11 participants (6 female) with an average age of 28.18 ($SD = 11.42$), 64 % students and 36 % employees. They had to use their own Android smartphone to test *Runnerful*. Apart from a short introductory video, no further help was provided, nor were participants controlled in any way. The study took place over two weeks, with the only task to run at least two recommended routes. Before the experiment, participants had to fill in a questionnaire we used to elicit demographics, fitness using IPAQ [4], running route preferences and previous experience with running apps. Afterwards, we assessed usability by means of SUS [3], and used items from [6] to assess recommendation quality and related aspects. Items were assessed on a positive 5-point Likert scale. We also recorded finished routes and corresponding ratings.

Participants reported that they performed vigorous physical activities for $M = 46.73$ min ($SD = 28.68$) on $M = 3.63$ days ($SD = 2.06$) in the week prior to the experiment. Most of them reported that nature is an important route property (9). Length (2) and elevation (2) were mentioned less frequently, which could however be due to our sample, without any competitive runners. Only 3 stated to have never used a running app before. Nevertheless, none of the participants ever tried a route recorded by another community member. Most of them reported to spontaneously decide for a route (82 %), but 6 stated to sometimes use a map or ask friends for advice.

We recorded 17 workouts from 9 participants (recording failed in two cases). Routes received average ratings ($M = 3.05$, $SD = 0.80$), and slightly higher ones when critiques were applied ($M = 3.17$, $SD = 0.69$). However, participants did not use critiquing very often, possibly because it was not displayed prominent enough. They stated that effort for receiving recommendations was low ($M = 2.00$, $SD = 0.98$), while perceived recommendation quality was above average ($M = 3.32$, $SD = 1.09$). When asked whether routes had the expected amount of desired properties, results were broadly average (e.g. for trees and forest: $M = 3.11$, $SD = 1.29$). Yet, this could be due to parametrization, favoring criteria such as length and elevation, together with our sample. Nevertheless, participants stated that they almost always found a suitable route ($M = 3.55$, $SD = 1.37$), which was most often novel ($M = 3.45$, $SD = 1.30$). Usability was rated as "good" (SUS-score of 80). Overall, it thus seems that our approach is in principle valid and appreciated by users. Fine-tuning, e.g. of the calculation of individual and overall scores, is, however, clearly needed. Still, most qualitative comments were concerned with aspects of our preliminary implementation (e.g. issues with the navigation function) rather than of our approach in general.

5 CONCLUSIONS AND OUTLOOK

In this paper, we discussed the challenges arising when recommending running routes, such as availability of data that is rich enough to adequately personalize these routes, and presented a framework describing aspects that have to be taken into account for this personalization to be effective. *Runnerful*, our proposed app, exploits easily accessible map datasources to generate routes of user-specified length. Then, by further processing the map data, we rank these candidate routes according to individual requirements. Critiquing allows the runner to interactively refine the results.

While the implementation of our framework together with the study shows the potential of the underlying approach, there is still room left for improvement. Also, more comprehensive evaluation with a larger number of users performing more workouts is required. For instance, the influence of the history criterion could not yet be adequately investigated. On the other hand, exploiting the user's running history more extensively might help to reduce interaction effort even further by letting the system learn which criteria are most important for him or her. Moreover, there exist contextual factors that could additionally be considered, such as current weather for recommending runs through the forest in midday heat or avoiding steep climbs in case of icy roads. Also, current fitness state as well as training fatigue could automatically be integrated when calculating the scores. Beyond that, practical issues such as scalability will be subject of future work: Our prototype lacks efficiency when generating longer routes which is necessary for experienced runners, but also in case it is adapted to e.g. cycling. Generating 5 km routes took up to 1 min, but in a dense city environment and with a rather average VM running the algorithm in the background. Thus, this is not a principle limitation, but requires additional preprocessing of map data, a more advanced selection mechanism of candidate routes, and highly depends on server infrastructure and used datasources. In general, the app needs technical improvements: For example, to make it more useful outside the study context, users should receive more support for following a route, e.g. by spoken navigation instructions. Nevertheless, and despite small sample size and the fact that some study results are still rather average, we think that the prototypical implementation of our framework successfully demonstrates our approach, and can thus be seen as a promising starting point for further research.

REFERENCES

[1] G. Adomavicius and Y. Kwon. 2015. *Recommender Systems Handbook*. Springer US, Chapter Multi-Criteria Recommender Systems, 847–880.

[2] J. Berndsen, A. Lawlor, and B. Smyth. 2017. Running with recommendation. In *HealthRecSys '17*.

[3] John Brooke. 1996. SUS – A quick and dirty usability scale. In *Usability Evaluation in Industry*. Taylor & Francis, 189–194.

[4] C. L. Craig, A. L. Marshall, M. Sjorstrom, A. E. Bauman, M. L. Booth, B. E. Ainsworth, M. Pratt, U. Ekelund, A. Yngve, J. F. Sallis, et al. 2003. International physical activity questionnaire: 12-country reliability and validity. *Med. Sci. Sport. Exer.* 35, 8 (2003), 1381–1395.

[5] A. Gemsa, T. Pajor, D. Wagner, and T. Zündorf. 2013. Efficient computation of jogging routes. In *SEA '13*. Springer, 272–283.

[6] B. P. Knijnenburg, M. C. Willemsen, and A. Kobsa. 2011. A pragmatic procedure to support the user-centric evaluation of recommender systems. In *RecSys '11*. ACM, 321–324.

[7] F. Mueller, J. Marshall, R. A. Khot, S. Nylander, and J. Tholander. 2014. Jogging with technology: Interaction design supporting sport activities. In *CHI '14*. ACM, 1131–1134.

[8] D. Münter, A. Kötteritzsch, T. Islinger, T. Köhler, C. Wolff, and J. Ziegler. 2012. Improving navigation support by taking care of drivers' situational needs. In *AutoUI '12*. ACM, 131–138.

[9] D. Quercia, R. Schifanella, and L. M. Aiello. 2014. The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city. In *HT '14*. ACM, 116–125.

[10] N. Runge, P. Samsonov, D. Degraen, and J. Schöning. 2016. No more autobahn! Scenic route generation using Google's street view. In *IUI '16*. ACM, 147–151.

[11] R. Schaller and D. Elsweiler. 2014. Itinerary recommenders: How do users customize their routes and what can we learn from them?. In *IIiX '14*. ACM, 185–194.

[12] B. Smyth and P. Cunningham. 2017. A novel recommender system for helping marathoners to achieve a new personal-best. In *RecSys' 17*. ACM, 116–120.

[13] B. Smyth and P. Cunningham. 2017. Running with cases: A CBR approach to running your best marathon. In *ICCBR '17*. Springer, 360–374.

[14] P. Stroobant, P. Audenaert, D. Colle, and M. Pickavet. 2018. Generating constrained length personalized bicycle tours. *4OR* (2018), 1–29.

Time-aware Personalized Popularity in top-N Recommendation

Vito Walter Anelli, Joseph Trotta,
Tommaso Di Noia, Eugenio Di Sciascio
Polytechnic University of Bari
Bari, Italy
firstname.lastname@poliba.it

Azzurra Ragone
Independent Researcher
Milan, Italy
azzurra.ragone@gmail.com

ABSTRACT

Items popularity is a strong signal in recommendation algorithms. It affects collaborative filtering algorithms and it has been proven to be a very good baseline in terms of results accuracy. Indeed, even though we miss an actual personalization, global popularity of items in a catalogue can be used effectively to recommend items to users. In this paper we introduce the idea of a *time-aware personalized popularity* by considering both items popularity among neighbors and how it changes over time. Although the proposed approach results computationally light, our experiments show that its accuracy results highly competitive compared to state of the art model-based collaborative approaches.

1 INTRODUCTION

Collaborative-Filtering (CF) [18] algorithms more than others have gained a key-role among various approaches to recommendation, in helping people to face the information overload problem. Some of them use additional information to build a more precise user profile in order to serve a much more personalized list of items [4, 8]. However, it is well known [12] that all the algorithms based on a CF approach (either in their pure version or in a hybrid one) are affected by the so called "popularity bias" where popular items tend to be recommended more frequently than the ones in the long tail. The effect of popularity on recommender systems has been debated for a long time. Initially considered as a shortcoming of collaborative filtering algorithms, unlikely useful to produce good recommendations [11], in some works it has been intentionally penalized [17]. However, in some recent works, popularity has been considered as a natural aspect of recommendation, and measuring the user tendency to diversification, it can be exploited in order to balance the recommender optimization goals [13]. Very interestingly, a recommendation algorithm purely based on most popular items, although it does not exploit any actual personalization, has been proven to be a strong baseline [6]. Moreover, a popularity-based recommendation algorithm does not require a heavy computational effort as it just considers the occurrence of an item within the profiles of all the users in a system. In the approach we present here, we change the "global" perspective of a "most popular" algorithm and we introduce a more fine-grained personalized version of popularity by assuming that it is conditioned by the items that a user u already experienced *in the past*. To this extent, we look at a specific class of neighbors, that we name *Precursors*, defined as the users who already rated the same items of u *in the past*. This leads us to the introduction of a time-aware analysis while computing a recommendation list for u.

As time is considered a contextual feature, most of the works that make use of it are classified as constituting a specialized family of Context-Aware RS (CARS) [2]: Time-Aware RS (TARS) [1, 14, 24]. In TARS, the freshness of different ratings is often considered a discriminative factor between different candidate items. This can be implemented using a time window [15] that filters out all the ratings that stand before (and/or after) a certain time relative to the user or the item. Recently, an interesting work that makes use of time windows has been proposed in [5] where the authors focused on the last common interaction between the target user and her neighbors to populate the candidate items list. When dealing with a time-aware algorithms, the splitting strategy plays a key-role. In [5], a single timestamp is used as a splitting condition for all the users, so that they retain 80% of ratings for training and the remainder for testing. A pioneer work was proposed more than a decade ago in [7] which used an exponential decay function $e^{-\lambda t}$ to penalize old ratings. After that, an exponential decay function [14] was used to integrate time in a latent factors model. In the last years, several Item-kNN [7, 16] with a temporal decay function have been deployed. Another interesting work was proposed in [23] where three different kinds of time decay were proposed: exploiting concave, convex and linear functions.

In this paper we present `TimePop`, an algorithm that combines the notion of personalized popularity conditioned to the behavior of users' neighbors while taking into account the temporal dimension. Differently from some of the approaches previously described, in `TimePop` we avoided both the time window approach that could severely restrict the selection of candidates and the fixed number of candidate items that could heavily affects the algorithm results. We evaluated our approach on three different datasets and compared with state of the art collaborative approaches, thus showing that `TimePop` outperforms significantly the competing algorithms in terms of nDCG.

The reminder of the paper is structured as follows: in the next section, we detail the ideas behind `TimePop` and we expose the lightweight process needed to compute recommendations. Section 3 is devoted to the description of the experimental setting. Conclusion and future work close the paper.

2 TIME-AWARE LOCAL POPULARITY

The leading intuition behind `TimePop` is that the popularity of an item has not to be considered as a global property but it can be personalized if we consider the neighbors of a user. We started from this observation to formulate a form of personalized popularity, and then we added the temporal dimension to strengthen this idea.

Given an item, the first step towards the introduction of this time-aware local popularity is the identification of subsets of users sharing the same degree of popularity and refer to them as neighbors when

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, Vancouver, Canada.

DOI:

computing a recommendation list. Still, in each of these subgroups we have that the same item is enjoyed by users in different time frames. In our model, among these people, for a single user u we consider those who already enjoyed the same items of u but before she did it. We name these users *Precursors*. This leads us to the second ingredient behind `TimePop`: personalized popularity is a function of time. The more the ratings about an item are recent, the more its popularity is relevant for the specific user. In order to exploit the temporal aspect of these rating the contributions of *Precursors* can be weighted depending on their freshness.

We now introduce some basic notation that will be used in the following. We use $u \in U$ and $i \in I$ to denote users and items respectively. Since we are not just interested in the items a user rated but also at when the rating happened, we have that for a user u the corresponding user profile is $P_u = \{(i_1, t_{ui_1}), \ldots, (i_n, t_{ui_n})\}$ with $P_u \subseteq I \times \Re$, being t_{ui} a timestamp representing when u rated i.

Definition 2.1 (Candidate Precursor). Given $(i, t_{ui}) \in P_u$ and $(i, t_{u'i}) \in P_{u'}$, we say that u' is a **Candidate Precursor** of u if $t_{u'i} < t_{ui}$. We use the set $\hat{\mathcal{P}}^u$ to denote the set of Candidate Precursors of u.

A user u' is a *Candidate Precursor* of u if u' rated at least one common item i before u. Although this definition catches the intuition behind the idea of *Precursors*, it is a bit weak as it considers also users u' who have only a few or even just one item in common with u and rated them before she did. Hence, we need to introduce a threshold taking somehow into account the number of common items in order to enforce the notion of *Precursors*. This threshold can be personalized or computed automatically (see Equation (1)).

Definition 2.2 (Precursor). Given two users u' and u such that u' is a Candidate Precursor of u and a value $\tau_u \in \Re$ we say that u' is a **Precursor** of u if the following condition holds.

$$|\{i \mid (i, t_{ui}) \in P_u \wedge (i, t_{u'i}) \in P_{u'} \wedge t_{u'i} < t_{ui})\}| \geq \tau_u$$

We use $\mathcal{P}^u$ to denote the set of **Precursors** of u.

A general procedure to evaluate τ_u can be that of considering the mean of the common items previously rated by $\hat{\mathcal{P}}^u$.

$$\tau_u = \frac{\sum_{u' \in \hat{\mathcal{P}}^u} |\{i \mid (i, t_{ui}) \in P_u \wedge (i, t_{u'i}) \in P_{u'} \wedge t_{u'i} < t_{ui})\}|}{|\hat{\mathcal{P}}^u|} \quad (1)$$

To give an intuition on the computation of Precursors and of τ_u let us describe the simple example shown in Figure 1. Here, for the sake

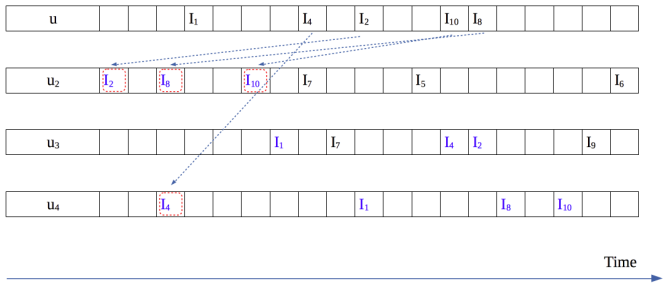

Figure 1: Example of Precursors computation.

of simplicity, we suppose that there are only four users and ten items and u is the user we want to provide recommendations to. Items that users share with u are highlighted in blue and items with a dashed red square are the ones that have been rated before u. We see that $\hat{\mathcal{P}}^u = \{u_2, u_4\}$. Indeed, although u_3 rated some of the items also rated by u they have been rated after. By Equation (1) we have $\tau_u = \frac{4}{2} = 2$. Then, only u_2 results to be in $\mathcal{P}^u$ because she has $3 > 2$ shared items rated before those of u. As for u_3, it is more likely that u is a Precursor of u_4 and not vice versa.

2.1 The temporal dimension

As the definition of Precursor goes through a temporal analysis of user behaviors, on the one side, we may look at the timestamp of the last rating provided by a Precursor in order to identify how active she is in the system. Intuitively, the contribution to popularity for users who have not contributed recently with a rating is lower than "active" users. On the other side, given an item in the profile of a Precursor we are interested in the freshness of its rating. As a matter of fact, old ratings should affect the popularity of an item less than newer ratings. Summing up, we may classify the two temporal dimensions as **old/recent user** and **old/recent item**. In order to quantify these two dimensions for Precursors we introduce the following timestamps:

- $\mathbf{t_0}$ this is the reference timestamp. It represents the "now" in our system;
- $\mathbf{t_{u'i}}$ is the time when u' rated i;
- $\mathbf{t_{u'l}}$ represents the timestamp associated to the last item l rated by the user u'.

In order to embed time in a recommendation approach, a temporal decay $e^{-\beta \cdot \Delta T}$ is usually adopted where ΔT is a variable taking into account the age of a rating. Different temporal variables are typically used [7, 14], and they mainly focus on **old/recent items**. ΔT may refer to the timestamp of the items with reference to the last rating of u' [7] $\Delta T = \mathbf{t_{u'l} - t_{u'i}}$ or to the reference timestamp [14] $\Delta T = \mathbf{t_0 - t_{u'i}}$. As we stated before, our approach captures the temporal behavior of both **old/recent users** and **old/recent items** at the same time. We may analyze the desired ideal behavior of ΔT depending on the three timestamps previously defined as represented in Table 1. Let us focus

	recent user ($\mathbf{t_0 \approx t_{u'l}}$)	**old user** ($\mathbf{t_0 \gg t_{u'l}}$)
recent item ($\mathbf{t_{u'l} \approx t_{u'i}}$)	≈ 0	$\mathbf{t_0 - t_{u'l}}$
old item ($\mathbf{t_{u'l} \gg t_{u'i}}$)	$\mathbf{t_{u'l} - t_{u'i}}$	$\mathbf{t_0 - t_{u'l}}$

Table 1: Ideal values of ΔT w.r.t. the Precursor characteristics

on each case. In the upper-left case we want ΔT to be as small as possible because both u' and the rating for i are "recent" and then highly representative for a popularity dimension. In the upper-right case, the rating is recent but the user is old. The last item has been rated very close to i but a large value of ΔT should remain because the age of u' penalizes the contribution. The lower-left case denotes a user that is active on the system but rated i a long time ago. In this case the contribution of this item is almost equal to the age of its rating. The lower-right case is related to a scenario in which both the rating and u' are old. In this scenario, the differences between the reference timestamp minus the last interaction and the reference timestamp minus the rating of i are comparable: $(\mathbf{t_0 - t_{u'l}}) \approx (\mathbf{t_0 - t_{u'i}})$. In this case, we wish the contribution of ΔT should consider the elapsed time from the last interaction (or the rating) until the reference timestamp. All the above observations, lead us to the following formulation:

$$\Delta T = |\mathbf{t_0 - 2t_{u'l} + t_{u'i}}| \quad (2)$$

It is quite straightforward deriving the ideal behavior for each case in Table 1 using Equation (2). In order to avoid different decay coefficients, all ΔTs are transformed in days (from milliseconds) as a common practice.

2.2 The Recommendation Algorithm

We modeled our algorithm TimePop to solve a *top-N* recommendation problem. Given a user u, TimePop computes the recommendation list by executing the following steps:

(1) Compute $\mathcal{P}^u$;
(2) For each item i such that there exists $u' \in \mathcal{P}^u$ with $(i, t_{u'i}) \in P_{u'}$ compute a score for i by summing the number of times it appears in $P_{u'}$ multiplied by the corresponding decay function;
(3) Sort the list in decreasing order with respect to the score of each i.

For sake of completeness, in case there were no precursors for a certain user, a recommendation list based on global popularity is returned to u. Moreover, if TimePop is able to compute only m scores, with $m < N$, the remaining items are returned based on their value of global popularity[1].

3 EXPERIMENTAL EVALUATION

In order to evaluate TimePop we decided to test our approach considering different domains and different tasks. Two of them related to the movie domain —the well-known Movielens1M[2] dataset (focused on free movie recommendations) and Amazon[3] Movies (in which single items can be purchased)— and a dataset referring to toys and games —Amazon Toys and Games. Moreover these datasets come with timestamps, which are needed for our purposes. "All Unrated Items" [22] protocol has been chosen to compare different algorithms where, for each user, all the items that have not yet been rated by the user all over the platform are considered.

Dataset splitting. In order to evaluate time-aware recommender systems in an offline experimental setting, a typical k-folds or hold-out splitting would be ineffective and unrealistic. We wanted the training set to be as close as possible to an on-line real scenario in which the recommender system is deployed. To reach this goal we used the splitting from [9], also used in [5]. Best practices in recommender systems evaluation suggest that it represents a more realistic temporal dynamic in an actual time-aware recommendation scenario. In details, we used a fixed timestamp $\mathbf{t_0}$ that has been chosen by finding the one that maximizes the number of users that maintain at least 15 ratings in the training set and 5 ratings in the test set. The choice of setting a minimum number of ratings less than 15 would have heavily affected the results, shadowing the evaluation of performance in a non cold-start scenario. Hence we decided to not address the cold-start users in this work. We exploited such a timestamp thus obtaining a training set that represents the past of our system with reference to $\mathbf{t_0}$, and a test set that collects the events that are going to happen, i.e., all those ratings happening after $\mathbf{t_0}$. Training set and test set for the three datasets are publicly available[4] along with the splitting code[5] for research purposes. The resulting datasets characteristics are depicted in Table 2.

In order to evaluate the algorithms we measured *normalized Discount Cumulative Gain*@N (*nDCG@N*). The metric was computed per user and then the overall mean was returned using the RankSys[6] framework. The threshold used to consider a test item as relevant has been set to the value of 4 w.r.t. a 1-5 scale for all the three datasets.

dataset	# users	# items	# ratings	spars.(%)
movielens	859	3,375	185,035	93.61756
AmazonMovies	3619	68,514	288,339	99.88371
AmazonToys	1108	24,158	38,317	99.85685

Table 2: Datasets statistics after splitting.

Baselines. We evaluated our approach w.r.t CF and time-aware techniques. **MostPopular** was included as TimePop is a time-aware variant of "Most Popular". From model-based collaborative filtering approaches we selected some of the best performing matrix factorization algorithms **WRMF** trained with 10 latent factors, a regularization parameter set to 0.015, α set to 1 and 15 iterations, and **FM**[19], computed with an ad-hoc implementation of a 2 degree factorization machine with 10 latent factors, considering users and items as features, trained using Bayesian Personalized Ranking Criterion[20]. We considered the ad-hoc implementation needed because we found no Java implementations of Factorization Machines optimized using BPR criterion explicitly written for recommender systems. Moreover, we compared our approach against the most popular memory-based kNN algorithms, **Item-kNN** and **User-kNN** [21], together with their time-aware variants (**Item-kNN-TD**, **User-kNN-TD**)[7]. We included **TimeSVD++** [14] in our comparison even though this latter has been explicitly designed for the rating prediction task while TimePop computes a *top-N* recommendation list. We included TimeSVD++ as it is one of the most important advances in time-aware RS. Finally **BFwCF** [5] is an algorithm that takes into account interaction sequences between users and it uses the last common interaction to populate the candidate item list. In this evaluation we included the BFwCF variant that takes advantage of similarity weights per user and two time windows, left-sided and right-sided (Backward-Forward). BFwCF was trained using parameters from [5]: 100 neighbors, indexBackWards and indexForwards set to 5, normalization and combination realized respectively via DummyNormalizer and SumCombiner. Recommendations were computed with the implementation publicly provided by authors. For all user-based and item-based scheme algorithms 80 neighbors were considered. Recommendations for **MostPopular**, **WRMF**, **User-kNN** were computed using the MyMediaLite[7] implementation. **Item-kNN**, **User-kNN-TD** and **Item-kNN-TD** were computed with an ad-hoc implementation based on MyMedialite and [7]. In particular, in order to guarantee a fair evaluation, for all the time-based variants the β coefficient was set to $\frac{1}{200}$ [14]. TimeSVD++ was trained using parameters used in [14]. All ad-hoc implementations are publicly available[8] for research purposes.

3.1 Results Discussion

Results of experimental evaluation are shown in Figure 2 which illustrate nDCG (2a, 2b, 2c) curves for increasing number of top ranked items returned to the user. Significance tests have been performed for accuracy metrics using Student's t-test and p-values and they result consistently lower than 0.05. By looking at Figure 2a we see that TimePop outperforms comparing algorithms in terms of accuracy on AmazonMovies dataset. We also see that algorithms exploiting a Temporal decay function perform well w.r.t. their time-unaware variants (User-kNN and Item-kNN) while matrix factorization algorithms (WRMF ,TimeSVD++ and FM) perform quite bad. We assume

[1] We wish to highlight that in the experimental evaluation presented in this work, the former conditions never occur. Hence, the results only refer to recommendations provided by TimePop.
[2] https://grouplens.org/datasets/movielens/1m/
[3] http://jmcauley.ucsd.edu/data/amazon/
[4] https://github.com/sisinflab/DatasetsSplits
[5] https://github.com/sisinflab/DatasetsSplits/tree/master/SplittingAlgorithms/FixedTimestamp
[6] http://ranksys.org/
[7] http://www.mymedialite.net/
[8] https://github.com/sisinflab/recommenders

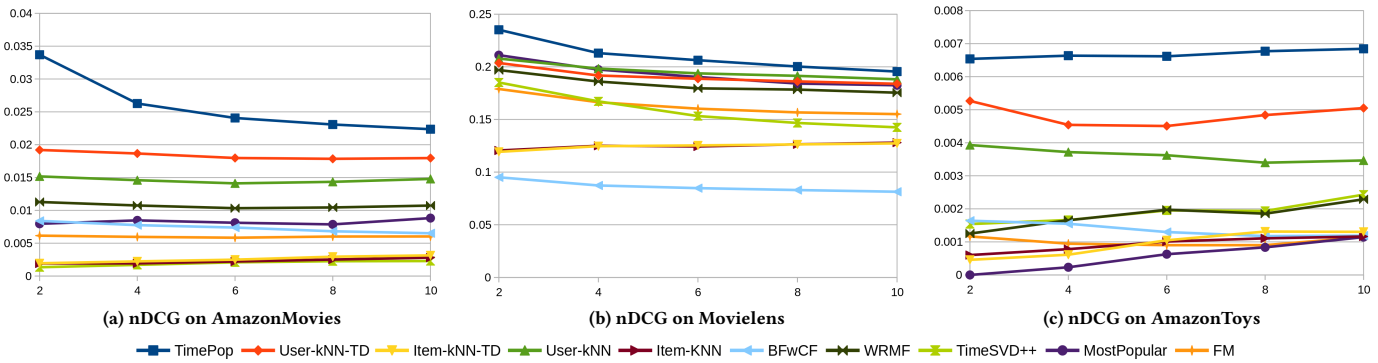

Figure 2: nDCG @N varying N in 2..10

this is mainly due to the high sparsity of the User-Item matrix (99.88%, Table 2). Results for Amazon Toys and Games dataset are analogous to those computed for Amazon Movies. The evaluation on the Movielens dataset shows some differences with reference to the previous two datasets. `TimePop` is still the most accurate approach, however WRMF, Time SVD++ and FM provide results which are more accurate than those computed for the two Amazon datasets. If we look at the sparsity values of the User-Item matrix (see Table 2) we observe Movielens dataset is less sparse than the other datasets. For this dataset it is worth to notice that taking into account time is not a key element (User-kNN-TD and Item-kNN-TD perform worse than the time-unaware variants), and MostPopular shows very high performance, even better than Matrix Factorization techniques. This is probably due to the strong popularity bias of `MovieLens` dataset. As for the lower influence of time in the accuracy of results we took a look at the distribution of timestamps in the various datasets with reference to the users. It is well known that timestamps in Movielens are related to the time ratings are inserted on the platform and they do not reflect the exact time of fruition for the item [3, 5, 10]. Moreover, in Movielens there are several users that rated a lot of movies the same day, with a user who reached the maximum of 1,080 movies rated the same day and another one with an average number of ratings per single day of 884. In Amazon Movies (maximum of 216 and a maximum average of 37) and Amazon Toys (maximum of 42 and a maximum average of 22.3) the trends are much different and this could heavily affect the results of time-aware algorithms. Nonetheless, it is important to notice that, despite that, `TimePop` always outperforms competing algorithms also in a dataset with low sparsity and high popularity bias such as Movielens.

4 CONCLUSION AND FUTURE WORK

In this paper we presented `TimePop`, a framework that exploits local popularity of items combined with temporal information to compute top-N recommendations. The approach relies on the computation of a set of time-aware neighbors named Precursors that are considered the referring population for a user we want to serve recommendations. We compared `TimePop` against state-of-art algorithms showing its effectiveness in terms of accuracy despite its lower computational cost in computing personalized recommendations.
We are currently working on the adoption of the ideas behind `TimePop` to the identification and computation of time-aware communities in recommender systems.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Multidimensional recommender systems: a data warehousing approach. *Electronic commerce*, pages 180–192, 2001.
[2] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.
[3] V. W. Anelli, V. Bellini, T. Di Noia, W. La Bruna, P. Tomeo, and E. Di Sciascio. An analysis on time-and session-aware diversification in recommender systems. In *Proceedings of the 25th UMAP*, pages 270–274. ACM, 2017.
[4] V. W. Anelli, T. Di Noia, E. Di Sciascio, and P. Lops. Feature factorization for top-n recommendation: from item rating to features relevance. In *Proceedings of the 1st RecSysKTL Workshop* 2017.
[5] A. Bellogín and P. Sánchez. Revisiting neighbourhood-based recommenders for temporal scenarios. In *Proceedings of the 1st Workshop on Temporal Reasoning in Recommender Systems*, pages 40–44, 2017.
[6] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the Fourth ACM RecSys '10*, pages 39–46. ACM, 2010.
[7] Y. Ding and X. Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM CIKM*, pages 485–492. ACM, 2005.
[8] I. Fernández-Tobías, M. Braunhofer, M. Elahi, F. Ricci, and I. Cantador. Alleviating the new user problem in collaborative filtering by exploiting personality information. *User Modeling and User-Adapted Interaction*, 26(2-3):221–255, 2016.
[9] A. Gunawardana and G. Shani. Evaluating recommender systems. In *Recommender Systems Handbook*, pages 265–308. 2015.
[10] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM TiiS*, 5(4):19, 2016.
[11] T. Jambor and J. Wang. Optimizing multiple objectives in collaborative filtering. In *Proceedings of the 4th ACM RecSys*, pages 55–62, 2010.
[12] D. Jannach, L. Lerche, F. Gedikli, and G. Bonnin. What recommenders recommend - an analysis of accuracy, popularity, and sales diversity effects. In *Proceedings of the 21th UMAP Proceedings*, pages 25–37, 2013.
[13] M. Jugovac, D. Jannach, and L. Lerche. Efficient optimization of multiple recommendation quality factors according to individual user tendencies. *Expert Syst. Appl.*, 81:321–331, 2017.
[14] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.
[15] N. Lathia, S. Hailes, and L. Capra. Temporal collaborative filtering with adaptive neighbourhoods. In *Proceedings of the 32nd ACM SIGIR*, pages 796–797. ACM, 2009.
[16] N. N. Liu, M. Zhao, E. Xiang, and Q. Yang. Online evolutionary collaborative filtering. In *Proceedings of 4th ACM RecSys*, pages 95–102. ACM, 2010.
[17] J. Oh, S. Park, H. Yu, M. Song, and S. Park. Novel recommendation based on personal popularity tendency. In *11th IEEE ICDM*, pages 507–516, 2011.
[18] S. Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.
[19] S. Rendle. Factorization machines. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pages 995–1000, 2010.
[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: bayesian personalized ranking from implicit feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 452–461, 2009.
[21] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.
[22] H. Steck. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 213–220. ACM, 2013.
[23] C. Xia, X. Jiang, S. Liu, Z. Luo, and Z. Yu. Dynamic item-based recommendation algorithm with time decay. In *Natural Computation (ICNC), 2010 Sixth International Conference on*, volume 1, pages 242–247. IEEE, 2010.
[24] A. Zimdars, D. M. Chickering, and C. Meek. Using temporal data for making recommendations. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 580–588. Morgan Kaufmann Publishers Inc., 2001.

Retrieving and Recommending for the Classroom

Stakeholders, Objectives, Resources, and Users

Michael D. Ekstrand
People & Information Research Team
Boise State University
Boise, ID
michaelekstrand@boisestate.edu

Ion Madrazo Azpiazu
People & Information Research Team
Boise State University
Boise, ID
ionmadrazo@u.boisestate.edu

Katherine Landau Wright
Literacy Lab
Boise State University
Boise, ID
katherinewright@boisestate.edu

Maria Soledad Pera
People & Information Research Team
Boise State University
Boise, ID
solepera@boisestate.edu

ABSTRACT

In this paper, we consider the promise and challenges of deploying recommendation and information retrieval technology to help teachers locate resources for use in classroom instruction. The classroom setting is a complex environment presenting a number of challenges for recommendation, due to its inherent multi-stakeholder nature, the multiple objectives that quality educational resources and experiences must simultaneously satisfy, and potential disconnect between the direct user of the system and the end users of the resources it provides. In this paper, we outline these challenges, highlight opportunities for new research, and describe our work in progress in this area including insights from interviews with working teachers.

KEYWORDS

multiple stakeholders, multi-objective recommendation

ACM Reference Format:
Michael D. Ekstrand, Ion Madrazo Azpiazu, Katherine Landau Wright, and Maria Soledad Pera. 2018. Retrieving and Recommending for the Classroom: Stakeholders, Objectives, Resources, and Users. In *Proceedings of ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios.* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

Students in the U.S. primary and secondary educational systems frequently engage with educational content through textbooks and commercially-available reading collections. Supplementing or replacing these readings with *authentic* (that is, created for purposes other than pedagogy, such as news or information), *current* texts that are *accessible* to students at their reading skill and domain knowledge and resonate with students' various interests has the potential to help students better engage with the material.

While suitable resources likely exist, it is difficult to find current news articles that are appropriate (both in content and readability) for upper-elementary, middle, and high school classrooms. As a result, teachers often either reuse outdated materials or opt not to engage students in this type of authentic reading. One Boise-area teacher we recently interviewed said "I want to make sure they are reading and writing in my class more, but I just, sometimes, either can't find or don't have the time to find the resources I need to get them to do that at the level where I know they can do that."

This is particularly true for teachers working with struggling readers and language learners, as the additional scaffolding such students require in order to understand the content of typical news sources seems (and often is) time- and cost-prohibitive. We see significant potential for information retrieval and recommendation technology to aid in this process, enabling teachers to quickly locate a diverse collection of texts from the Web to help their students connect their learning to life and the world around them.

Elsewhere [11] we have discussed some of the challenges, particularly around data availability, in building and evaluating applications in this setting. In this paper, we focus on the intrinsic complexity of the recommendation problem itself: locating relevant, current texts in a classroom setting. We identify four primary dimensions that complicate this problem — multiple stakeholders, multiple objectives, multiple desired resources, and a disconnect between the system user and the end user of the retrieved content — that together make it a significantly more complex recommendation scenario than is typically considered in the research literature.

Effectively meeting teacher and classroom information needs in this setting will require significant new advances across multiple disciplines and specialties. Our argument here draws from our study of the problem space, interviews with teachers about their current and desired information practices[1], and our experience developing a prototype tool for locating news articles for classroom use.

2 MULTIPLE STAKEHOLDERS

Many — if not most — recommendation problems involve multiple stakeholders [3]. Systems have direct users, but content creators, system operators, and society at large can be helped or harmed by the recommender system's operation. The extent to which these

ComplexRec 2018 Second Workshop on Recommendation in Complex Scenarios, October 7, 2018, Vancouver, Canada.

[1] Interviews were conducted under a study design approved by the Boise State University Institutional Review Board, protocol #113-SB17-238.

different stakeholders' perspectives should be considered or represented in the design or evaluation of recommender systems is just now starting to see exploration [3, 5].

Burke and Abdollahpouri [2] identify a number of stakeholders involved in certain educational recommendation contexts: in recommending educational opportunities to students in the Chicago City of Learning program, both individual students' needs and interests as well as the interests of the various institutions providing recommended opportunities are relevant to assessing the system's effectiveness at matching students with opportunities.

The classroom setting we endeavor to enhance creates even more complex problems in terms of the set of stakeholders:

- Individual students have an interest in their education, and also have particular interests, ambitions, and capabilities.
- The teacher has an interest in fostering student learning engaging students with content.
- The school and its supporting institutions (e.g. the state, in public education) have particular learning outcomes and established standards regarding student learning and classroom instruction.
- The community has an interest in well-educated youth who are able to apply content knowledge to their world and meaningfully interpret current events.

Accounting for the impact of new educational capabilities on these stakeholders in both the design and evaluation of these technologies is a challenge. We are taking a teacher-centered approach, trusting teachers to know their educational contexts as well as anyone, and beginning our work by seeking to understand how they locate and curate resources for their classrooms.

3 MULTIPLE OBJECTIVES

Most recommendation systems are optimized with a single objective in mind, i.e., optimize sales or click-rate. Multi-objective recommendation techniques [13] move beyond this to jointly optimize multiple criteria such as offline accuracy and diversity. Classroom material recommendations need to consider trade-offs between several objectives that sometimes compete between each other in order to find adequate resources; further, some of these objectives are imposed by external constraints.

The teachers we spoke with highlighted the difficulty of using existing systems to locate new texts. One teacher said "I try to look online, on Google and stuff like that but there's… a vast array of stuff and you have to really search for it…". Existing technologies, while effective at optimizing for general query relevance, do not take into account the specific objectives of teachers in a classroom setting. Tools that do so have the potential to make it far easier for teachers to locate useful material.

One immediate objective in the classroom setting is *readability*. In order to learn from a text with assistance, a child should be able to decode 90% of it; to learn independently, that requirement rises to 98% [1]. Multiple teachers mentioned this specific difficulty; one commented on the difficulty of finding "things they [her students] can read and understand." An effective retrieval or recommendation system for educational reading material should help the teacher ensure the documents are readable by each student in the class.

Retrieved materials should be *curricularly relevant*: they should connect to the curricular needs of the students and classroom so that core topics taught in class are reinforced by the readings. To date, there is not one set of curriculum standards, therefore the needs are going to vary by state and district. Furthermore, there are additional standards to meet the needs of diverse students — for instance, those instructing English Language Learners need to address both content and language development standards [15].

In addition to relevance to core curriculum topics such as math or science, the teacher may wish to target resources that promote *side skills* such as critical thinking, reasoning, or understanding and respect towards other cultures. For instance, students are better able to understand those who are different from them when they have an opportunity to read about and vicariously experience other perspectives [6, 14].

Student interest is important to motivate students and facilitate learning. Prioritizing resources likely to match student interests will make it easier for the teacher use the system to enhance their teaching. In order to prioritize interesting resources, the system should be able to consider the time of the year, location, and recency (a document might become of interest right after an specific event) of candidate resources, as well as individual backgrounds and personal interests of the students. Several teachers mentioned this challenge as well; one biology teacher specifically wished she could find readings to make the content more relevant, as very few of her students had interest in pursuing STEM fields.

Finally, the content of recommended resources should be *appropriate* for an educational environment, avoiding content that can risk the psychological integrity of the students. Defining such safe or suitable content, however, poses a challenge on itself, as it is influenced by multiple factors including age, culture, religion, geopolitical context, or even past experiences of the student. As the experts on their particular teaching context and group of students, teachers know these factors as well as anyone. A system that works with and empowers the teacher, instead of replacing or automating their work, can enable learning experiences that leverage that expertise to avoid local faux pas.

The complex multi-objective needs of recommendations in the classroom environment highlight a need of researchers from multiple disciplines to cooperate in order to adequately address the problem.

4 MULTIPLE RESOURCES

Much existing recommendation literature has focused on recommending individual items or lists of items from which the user will select one to purchase or experience. Some work has looked at *set* or *package* recommendations, where multiple items are to be consumed together, or where the set is selected as a whole with the goal of improving the user's overall experience with the decision-making process and its outcome.

In selecting resources for classroom instruction, the teacher will typically be looking for a collection of readings that will map to different students' interests, experiences, and abilities. One teacher we interviewed described her efforts to find "mild", "medium", and "spicy" (referring to the reading levels) texts on a similar topic to

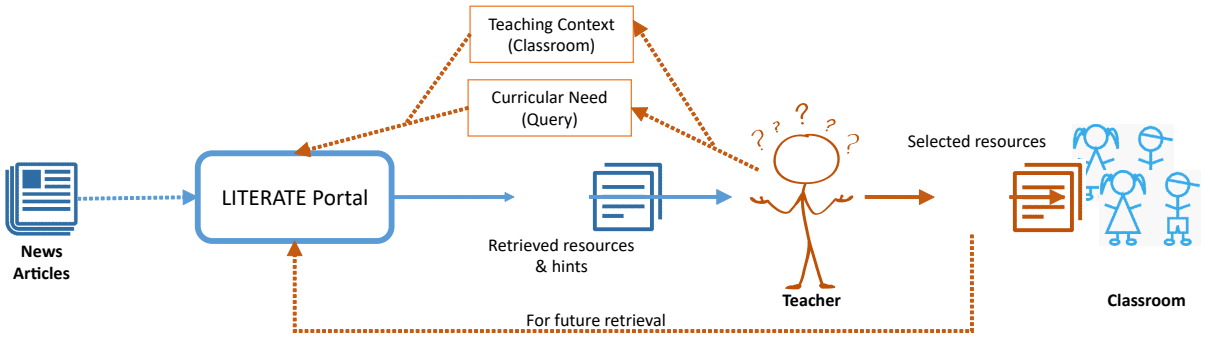

Figure 1: LITERATE Architecture

reach the diverse needs of her students. One-size-fits-all recommendation and retrieval is unlikely to produce a compelling learning experience.

We have the opportunity, though, to decompose the problem somewhat: rather than attempting to do single-shot recommendation of an entire collection of readings, we can consider algorithms and interfaces that support incremental curation of the final selection: suggesting articles that will meet student needs that are not already covered by the articles selected so far.

This setting will also provide opportunity to study additional modes of recommendation in the curation process, such as identifying items in the collection that have become redundant, or items that could replace existing items and improve the collection's overall usefulness for the classroom.

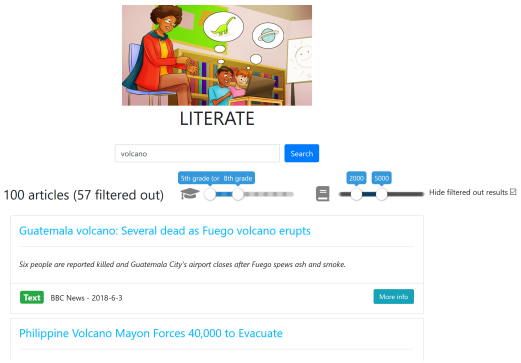

Figure 2: The current LITERATE Interface

5 EXPERT IN THE LOOP

Finally, supporting classroom instruction involves a system user (the teacher) who is distinct from the end users of the content (the students). This is quite useful for addressing some difficulties in supporting classroom instruction, such as final assessments of resource suitability and accounting for local context in selecting resources. However, it takes the problem outside of the realm of most existing research on human-recommender interaction.

The vast majority of research has focused on supporting direct users who are consuming content for themselves. Outside of human-centered recommender systems research, existing models and theories of information-seeking behavior [4, 12] similarly tend to focus on users seeking to meet their own information needs. There is little existing research to guide adaptations to algorithms, explanations, and other aspects of the system to such settings.

In addition to retrieval and recommendation algorithms for locating and ranking candidate resources, meeting educators' information needs will require substantial user interface work to enable the teacher to express their needs and provide them with aids and explanations to evaluate the retrieved resources. This may be eased somewhat by the fact that the system user has substantial domain expertise, but both the information need and resource relevance criteria have a great deal of information that needs to be elicited and displayed.

6 THE LITERATE PROJECT

These issues arise in the context of our work to develop LITERATE (Locating Informational Texts to Engage Readers And Teach Equitably) [11], a tool for helping teachers locate informational texts from the web to enhance their work with students.

Reading about and understanding the experiences of others can promote empathy and civility amongst students [8]. In its current iteration, which will serve as foundation for future research projects in this area, LITERATE aims to promote equity and empathy in education by helping teachers more efficiently find *news articles* to engage their students in dialog about current events. Incorporating such material into the classroom will help teachers engage learners in the democratic process; providing computational support to help the teacher tailor material to the needs and interests of the various students in the classroom will enhance their ability to provide these benefits to *all* students.

Using news to discuss current events in education gives students an opportunity to consider diverse perspectives and learn to engage as active and responsible citizens [7]. This sort of civic education can increase political engagement for underrepresented minority and marginalized populations [10]. We eventually want to help teachers locate resources from across the web, but the high pedagogical usefulness of news makes it a promising domain for the first version of LITERATE, shown in Figure 2.

As illustrated in Figure 1, LITERATE will support teachers in tailoring content to match individual student needs. We use NewsAPI to locate news articles for a teacher-specified topic, and filter and annotate the results based on *reading levels*, in addition to other contextual features, such as resource *length*, *type*, targeted *grade* ranges, and top-3 representative *keywords*. The key technical contribution of LITERATE's current iteration is the incorporation of readability into the search process.

To inform our development and research work, we have been interviewing teachers in the Boise, Idaho area about their current practices and desired capabilities for locating supplemental texts and incorporating them into their teaching.

LITERATE is an ongoing project, and its further development will require us to address each of the dimensions of complexity we have described, in addition to modeling nontrivial information in a complex information space. We will need to further develop news representations with rich metadata we can leverage to match K-12 curriculum, design and test interfaces to capture complex information needs via the expert in the loop, and adapting the content of "relevance" to capture classroom suitability, students' abilities and backgrounds and teachers' curricular needs.

Using LITERATE as a platform, we will be able to evaluate and refine solutions as we receive direct feedback from teachers, advancing the state of the art in supporting complex information retrieval and recommendation tasks. As an immediate next step, we expect to incorporate much richer notions of text cohesion and content suitability into our ranking strategy while slowly transitioning from retrieval to recommendations. We also aim to enable LITERATE to tune its results to the curricular and stakeholder requirements of an specific classroom and to suggest sets of news articles that match curricular needs as the academic year progresses while accounting for readability levels and other needs of the students in the corresponding classroom.

We see the Web as the greatest open textbook available to educators, and LITERATE will — we hope — give them the power to find the right page in their quest for suitable class resources.

7 CONCLUSION

Supporting teachers in the work of preparing for classroom instruction is a complex, multi-dimensional information need. Substantial new work in both the user experience and underlying algorithmic foundations of information retrieval and recommender systems will be needed in order to deliver applications that are efficient and responsive to pedagogical needs.

At the same time, there is great promise in the ability for new technologies to support the work of teachers in providing compelling, engaging, and current material to their students. The teachers we interviewed repeatedly highlighted the difficulties in locating, curating, and using new texts with existing technologies in the limited time they have available, and we don't think it needs to be so difficult.

Empowering teachers to improve the diversity, relevance, and representativeness of the texts in their classrooms will also have valuable social effects. The texts themselves are likely to promote civic and political engagements [8]. There is also a significant gap in the availability of enriching texts for students of different socioeconomic status [9]; aiding teachers in making use of freely available texts from the Web has the potential to help close this gap by providing richer sets of readings to students who previously did not have them available.

We expect our future work on this project to result in significant advancements in recommender systems and information retrieval technology, particularly in eliciting and meeting complex, multi-dimensional information needs, and have a positive impact on the work of teachers and their students' learning experiences.

ACKNOWLEDGMENTS

This work has been funded by a Civility Grant from the Boise State University College of Education, and partially funded by the National Science Foundation under Award 15-65937. Patrick Cullings and Michael Green developed the prototype LITERATE software, and David McNeill contributed to software development and analysis of teacher interviews.

REFERENCES

[1] Richard L Allington. 2013. What Really Matters When Working With Struggling Readers. *The Reading teacher* 66, 7 (April 2013), 520–530. https://doi.org/10.1002/TRTR.1154

[2] R Burke and H Abdollahpouri. 2016. Educational Recommendation with Multiple Stakeholders. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW)*. 62–63. https://doi.org/10.1109/WIW.2016.028

[3] Robin D Burke, Himan Abdollahpouri, Bamshad Mobasher, and Trinadh Gupta. 2016. Towards Multi-Stakeholder Utility Evaluation of Recommender Systems. In *UMAP Extended Proceedings*.

[4] Donald O Case and Lisa M Given. 2016. Models of Information Behavior. In *Looking for Information: A Survey of Research on Information Seeking, Needs, and Behavior*. Emerald Group Publishing, Chapter 7, 141–176.

[5] Michael D Ekstrand and Martijn C Willemsen. 2016. Behaviorism is Not Enough: Better Recommendations Through Listening to Users. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 221–224. https://doi.org/10.1145/2959100.2959179

[6] Tracey S Hodges, Erin McTigue, Katherine Landau Wright, Amanda D Franks, and Sharon D Matthews. 2018. Transacting With Characters: Teaching Children Perspective Taking With Authentic Literature. *Journal of Research in Childhood Education* 32, 3 (July 2018), 343–362. https://doi.org/10.1080/02568543.2018.1464529

[7] Jennice McCafferty-Wright and Ryan Knowles. 2016. Unlocking the Civic Potential of Current Events with an Open Classroom Climate. *Social Studies Research & Practice (Board of Trustees of the University of Alabama)* 11, 3 (2016).

[8] Erin McTigue, April Douglass, Katherine L Wright, Tracey S Hodges, and Amanda D Franks. 2015. Beyond the story map. *The Reading Teacher* 69, 1 (2015), 91–101.

[9] Susan B Neuman. 2016. Opportunities to Learn Give Children a Fighting Chance. *Literacy Research: Theory, Method, and Practice* 65, 1 (Nov. 2016), 113–123. https://doi.org/10.1177/2381336916661543

[10] Anja Neundorf, Richard G Niemi, and Kaat Smets. 2016. The compensation effect of civic education on political engagement: How civics classes make up for missing parental socialization. *Political Behavior* 38, 4 (2016), 921–949.

[11] Maria Soledad Pera, Katherine Wright, and Michael D Ekstrand. 2018. Recommending Texts to Children with an Expert in the Loop. In *Proceedings of the 2nd International Workshop on Children & Recommender Systems (KidRec)*. https://doi.org/10.18122/cs_facpubs/140/boisestate

[12] Peter Pirolli. 2007. *Information Foraging Theory: Adaptive Interaction with Information.* Oxford University Press, Cary, NC, USA.

[13] Marco Tulio Ribeiro, Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. 2012. Pareto-efficient Hybridization for Multi-objective Recommender Systems. In *Proceedings of the Sixth ACM Conference on Recommender Systems (RecSys '12)*. ACM, New York, NY, USA, 19–26. https://doi.org/10.1145/2365952.2365962

[14] Barbara J Shade, Cynthia A Kelly, and Mary Oberg. 1997. *Creating culturally responsive classrooms.* American Psychological Association.

[15] WIDA Consortium. 2014. *The WIDA Standards Framework and its Theoretical Foundations.* White Paper. University of Wisconsin. https://www.wida.us/standards/eld.aspx#2012