Link Prediction and the Evolution of Communities on Twitter

Master’s Thesis

Oscar Castañeda
Link Prediction and the Evolution of Communities on Twitter

MASTER’S THESIS

submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE
Track Information Architecture

by

Oscar Castañeda
born in Guatemala, Guatemala

Web Information Systems Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
http://eemcs.tudelft.nl
Link Prediction and the Evolution of Communities on Twitter

Abstract

This research is about the influence of link prediction on the evolution of communities on Twitter. We collected tweets from three technology micro-bloggers who led us through their followings and tweets to tens of thousands of unique users over several weeks. We analyzed conventional and alternative information streams for these micro-bloggers based on URLs embedded in their tweets and in tweets of followees and followees-of-followees. We model users based on the most recent URLs embedded on their tweets and the latest users they follow, from which we infer links and extract semantic entities that are indicative of their interests. Furthermore, we propose a pipeline of methods for user modeling and personalization of communities of interest on Twitter. We test the performance of different organizational principles in community design, including the principles of hierarchy, user interests and the baseline follower mechanism on Twitter, which is based on user intuitions.

The goal of this thesis is to create a better notion of community by automatically calculating adaptive and personalized structures of followees that produce highly interesting content. Designing communities in this way is useful because it enables people to know in which community they are organized during a given period of time and because it enables community-based recommendations. Furthermore, designing communities based on organizational principles enables their automatic construction. Currently, communities are manually constructed by users through a tedious process of following and unfollowing which is based on disconnected user intuitions. We investigate whether it is possible to infer links between Twitter users who are not explicitly connected on Twitter and explore whether such automatically inferred social networks would allow for improving content recommendations on Twitter.

Thesis Committee:

Prof.dr.ir. G.J.P.M. Houben, Faculty EEMCS (WIS)
Dr. L. Hollink, Faculty EEMCS (WIS)
Dr. F. Abel, Faculty EEMCS (WIS)
Dr. E. Visser, Faculty EEMCS (SERG)
Para mis abuelos, Miriam Taracena y Oscar Castañeda, Ruth De León y Francisco Villagrán Kramer.

To my grandparents, Miriam Taracena and Oscar Castañeda and Ruth De León and Francisco Villagrán Kramer.
Acknowledgements

• Supported by the International Information Systems Security Certification Consortium - ISC² - Graduate Research Scholarship.

• Supported by the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America, scholarship No. E07M401381GT.

• Con el apoyo del Programa Alβan, Programa de Becas de Alto Nivel de la Unión Europea para América Latina, beca No. E07M401381GT.

• Supported by the Google Summer of Code program 2010.

• Supported by the EURO organizing committee for attending the 24th European Conference on Operational Research (EURO) at the University of Lisbon.

• Supported by the IEEE for attending the 10th Conference on Peer-to-Peer (P2P’10) at Delft University of Technology.
Preface

Since I started my studies in Delft I have been intrigued by the possibilities of studying and understanding human behavior in online social networks. By the time I started the MSc. Computer Science, in September of 2008, I was busy finalizing my thesis on the organization of open source communities for the MSc. Management of Technology. It was quite challenging to follow courses while finishing my first research project, but at the same time it was a remarkably fun and enriching experience. My interest in the management, design and engineering of organizations using information and software technology continues to this day. In this research project I have taken a step further by studying how behavior and its observable organization changes over time. I see exciting opportunities in the study of evolution in online social networks by employing methods and techniques from artificial intelligence. In this thesis I have started exploring such directions.

I would like to thank my supervisors, Prof.dr.ir. Geert-Jan Houben, Dr. Laura Hollink, Dr. Fabian Abel and Dr. Eelco Visser. Thank you Geert-Jan for your mentorship, patience and insightful feedback during omicron group meetings and on preliminary versions of my report and in my proposals for further research. Thank you Fabian for sharing with me your insights into recommender systems, for your thoughtful questions and comments during the omicron group meetings and for your comments on preliminary versions of my report. Thank you Laura for agreeing to be part of my examination committee. Thank you Eelco for introducing me to model-driven software development, for agreeing to be my external supervisor and for your comments on preliminary versions of my report.

During my studies and research I benefited from the partial scholarship ‘Information Security Scholarship’ from the International Information Systems Security Certification Consortium (ISC²). I kindly thank the selection committee at ISC² for selecting me for this scholarship which helped me to pursue further education at Delft University of Technology. I would also like to thank the Al/Jan programme of the European Union for the financial support I received for my previous education at Delft University of Technology, which in many interesting ways intersected with the education I am now completing.

During the first summer period of my master’s in computer science I benefited from project sponsorship from Google’s Open Source Programs Office as part of their Summer of Code program. I would like to thank Carol Smith and Ellen Koe for believing in my alternate project proposals and for supporting my work. The Google Summer of Code program provided me with a fun way to learn about programming in Python for open source projects.

As part of my studies and research I attended the 10th Conference on Peer-to-Peer (P2P’10) at Delft University of Technology, held in Delft in August of 2010. I also attended the 24th European Conference on Operational Research (EURO) at Universidade de Lisboa, held in Lisbon in June of 2010. I would like to thank the
organizing committees from the IEEE and EURO for their support which enabled me to attend their conferences and thus get exposed to the latest developments in operations research and research in peer-to-peer technologies.

I wish to express appreciation to my classmates and friends. Thanks to my colleagues Richard, Olaf, Arjan and Jeremy for their help in my trial presentation and for their helpful comments on my work in progress during omicron group meetings. Thanks also to Siem with whom I shared fun mornings and afternoons working on an exciting project to introduce recommender technology into the WebDSL language. Thanks to Omer for giving me a space in his server to compute communities of interest.

Thanks also to Isaia for brightening my days with her sweetness. And finally, I wish to thank my family for their unconditional support and love. I thank my uncles for their endless support. I am grateful to my grandparents for always encouraging me to pursue further education and for stimulating me to pursue challenging goals. I thank my parents for inspiring me to pursue my dreams.
# Contents

Acknowledgements v  
Preface vii  
Contents ix  
List of Figures xiii  
List of Tables xiv  

## 1 Introduction  
1.1 Motivation and previous work  
1.1.1 Limitations of Conventional Information Streams  
1.2 Research gap  
1.2.1 Research goal  
1.2.2 Research questions  
1.3 Introduction to communities of interest on Twitter  
1.3.1 Modeling technology micro-bloggers  
1.4 Relevance of this research  
1.4.1 Scientific relevance  
1.4.2 Practical relevance  
1.5 Structure of the report  

## 2 Related Work  
2.1 Background on Social Web research focused on Twitter  
2.2 Social Web research focused on Twitter  
2.2.1 Semantic Enrichment for User Profile Construction  
2.2.2 Analysis of User Modeling  
2.2.3 Analysis of cross-system User Modeling  
2.2.4 Applications of Learning Semantic Relationships between Entities  
2.3 Link Prediction  
2.3.1 Link Prediction Problem  
2.3.2 Hierarchical Link Prediction  
2.3.3 Organizing principles in social networks  
2.3.4 Hierarchical structure model  
2.3.5 The method of missing link prediction
2.4 Collaborative Filtering

2.4.1 Link prediction approach to collaborative filtering

2.4.2 Limitations

2.4.3 Advantages

3 Designing Communities of Interest

3.1 Observations from related work

3.1.1 Semantic enrichment of tweets to generate user profiles

3.1.2 Consideration of temporal profile patterns

3.1.3 Adaptive faceted search based on URLs

3.1.4 Link prediction and recommender systems

3.2 Synthesized design approach

3.2.1 Key design decisions

3.2.2 Communities of interest

3.2.3 Pipeline of methods

3.3 Information Streams

3.3.1 Conventional Information Streams

3.3.2 Alternative Information Streams

LP-based predicted followee network

Filtered LP-based predicted followee network

CF-based predicted followee network

Filtered CF-based predicted followee network

4 Engineering Communities of Interest

4.1 Challenges and Solutions

4.1.1 Search Strategy

4.1.2 Tractability

4.1.3 API limitations

4.1.4 Bag-of-words for semantic entities

4.1.5 Timeouts

4.1.6 Centralized processing of recommendations and evaluations

4.1.7 Monitoring and logging

4.1.8 URL extraction and expansion

4.1.9 URL filtering

4.2 Engineering Information Streams

4.2.1 Conventional Information Streams

4.2.2 Alternative Information Streams

LP-based network

CF-based network

5 Evaluating Communities of Interest

5.1 Research Methodology

5.1.1 Measuring Relevance

5.1.2 Measuring Relevance in Information Streams

Evaluation Approach

Evaluation Strategy for Communities of Interest

5.2 Results

5.2.1 \( T^{(j)}_{\mu} > LP - based_{\mu}^{(j)} \): Hypotheses 1 and 2

Findings

5.2.2 \( T^{(j)}_{\mu} < CF - based_{\mu}^{(j)} \): Hypotheses 3 and 4

Findings
# List of Figures

1.1 Competing Tech Blogger Accounts on Twitter ................................ 10
1.2 Average number of tweets per day for three different users .......... 10

3.1 Conventional Followee Network ($T$) .............................................. 33
3.2 Predicted Followee Network ($LP$ − based) ................................. 35
3.3 Recommended Followee Network ($CF$ − based) .............................. 36

4.1 Snow-ball collection excerpt from CFN.py .................................... 39
4.2 Rate API excerpt from CFN.py ...................................................... 41
4.3 Bag-of-semantic-entities excerpt from ICF.py ............................... 42
4.4 Timeouts excerpt from CFN.py ...................................................... 42
4.5 centralized processing of recommendations and evaluations excerpt from $T$.py ... 43
4.6 Email from CFN.py ................................................................. 44
4.7 Monitoring and logging excerpt from CFN.py ................................. 44
4.8 URL extraction and expansion excerpt from CFN.py ..................... 45
4.9 URL filtering excerpt from CFN.py ............................................... 47
4.10 ($T$ − based network) - Dataset Overview .................................. 48
4.11 ($LP$ − based network) - Dataset Overview .................................. 50
4.12 ($CF$ − based network) - Dataset Overview .................................. 51

A.1 Link prediction vs. common neighbors predictor [38] .................... 73
A.2 Link prediction vs. graph distance predictor [38] ......................... 74
A.3 Link prediction vs. random predictor [38] ..................................... 75
A.4 Link prediction vs. common neighbors predictor [17] .................... 76
A.5 Link prediction vs. graph distance predictor [17] ......................... 76
A.6 Link prediction vs. random predictor [17] ..................................... 77

D.1 $T^{(f)}_n > LP$ − based$_{LP}^{(f)}$ (Source User: ‘Mashable’) .................. 101
D.2 $T^{(f)}_n < LP$ − based$_{LP}^{(f)}$ (Source User: ‘BoingBoing’) .................. 101
D.3 $T^{(f)}_n > LP$ − based$_{LP}^{(f)}$ (Source User: ‘TechCrunch’) .............. 102
D.4 $T^{(f)}_n < CF$ − based$_{CF}^{(f)}$ (Source User: ‘Mashable’) ............... 103
D.5 $T^{(f)}_n < CF$ − based$_{CF}^{(f)}$ (Source User: ‘BoingBoing’) .............. 103
D.6 $T^{(f)}_n$ $CF$ − based$_{CF}^{(f)}$ (Source User: ‘TechCrunch’) .................. 104
D.7 $LP$ − based$_{LP}^{(f)}$ $< CF$ − based$_{CF}^{(f)}$ (Source User: ‘Mashable’) ....... 104
D.8 $LP$ − based$_{LP}^{(f)}$ $< CF$ − based$_{CF}^{(f)}$ (Source User: ‘BoingBoing’) ...... 105
D.9 $LP$ − based$_{LP}^{(f)}$ $< CF$ − based$_{CF}^{(f)}$ (Source User: ‘TechCrunch’) ...... 105

### List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Summary of Findings for Hypotheses 1, 3 and 5</td>
<td>59</td>
</tr>
<tr>
<td>5.2</td>
<td>Summary of Findings for Hypotheses 2, 4 and 6</td>
<td>60</td>
</tr>
<tr>
<td>B.1</td>
<td>Two-sample t test with equal variances: \textit{CFN} vs. \textit{PFN} (user: ‘Mashable’)</td>
<td>80</td>
</tr>
<tr>
<td>B.2</td>
<td>Two-sample t test with equal variances: filtered \textit{CFN} vs. filtered \textit{PFN} (user: ‘Mashable’)</td>
<td>80</td>
</tr>
<tr>
<td>B.3</td>
<td>Two-sample t test with equal variances: \textit{CFN} vs. \textit{RFN} (user: ‘Mashable’)</td>
<td>81</td>
</tr>
<tr>
<td>B.4</td>
<td>Two-sample t test with equal variances: filtered \textit{CFN} vs. filtered \textit{RFN} (user: ‘Mashable’)</td>
<td>81</td>
</tr>
<tr>
<td>B.5</td>
<td>Two-sample t test with equal variances: \textit{PFN} vs. \textit{RFN} (user: ‘Mashable’)</td>
<td>82</td>
</tr>
<tr>
<td>B.6</td>
<td>Two-sample t test with equal variances: filtered \textit{PFN} vs. filtered \textit{RFN} (user: ‘Mashable’)</td>
<td>82</td>
</tr>
<tr>
<td>B.7</td>
<td>Two-sample t test with equal variances: \textit{CFN} vs. \textit{PFN} (user: ‘BoingBoing’)</td>
<td>84</td>
</tr>
<tr>
<td>B.8</td>
<td>Two-sample t test with equal variances: filtered \textit{CFN} vs. filtered \textit{PFN} (user: ‘BoingBoing’)</td>
<td>84</td>
</tr>
<tr>
<td>B.9</td>
<td>Two-sample t test with equal variances: \textit{PFN} vs. \textit{RFN} (user: ‘BoingBoing’)</td>
<td>85</td>
</tr>
<tr>
<td>B.10</td>
<td>Two-sample t test with equal variances: filtered \textit{PFN} vs. filtered \textit{RFN} (user: ‘BoingBoing’)</td>
<td>85</td>
</tr>
<tr>
<td>B.11</td>
<td>Two-sample t test with equal variances: \textit{PFN} vs. \textit{RFN} (user: ‘BoingBoing’)</td>
<td>86</td>
</tr>
<tr>
<td>B.12</td>
<td>Two-sample t test with equal variances: filtered \textit{PFN} vs. filtered \textit{RFN} (user: ‘BoingBoing’)</td>
<td>86</td>
</tr>
<tr>
<td>B.13</td>
<td>Two-sample t test with equal variances: \textit{CFN} vs. \textit{PFN} (user: ‘TechCrunch’)</td>
<td>88</td>
</tr>
<tr>
<td>B.14</td>
<td>Two-sample t test with equal variances: filtered \textit{CFN} vs. filtered \textit{PFN} (user: ‘TechCrunch’)</td>
<td>88</td>
</tr>
<tr>
<td>B.15</td>
<td>Two-sample t test with equal variances: \textit{CFN} vs. \textit{RFN} (user: ‘TechCrunch’)</td>
<td>89</td>
</tr>
<tr>
<td>B.16</td>
<td>Two-sample t test with equal variances: filtered \textit{CFN} vs. filtered \textit{RFN} (user: ‘TechCrunch’)</td>
<td>89</td>
</tr>
<tr>
<td>B.17</td>
<td>Two-sample t test with equal variances: \textit{PFN} vs. \textit{RFN} (user: ‘TechCrunch’)</td>
<td>90</td>
</tr>
<tr>
<td>B.18</td>
<td>Two-sample t test with equal variances: filtered \textit{PFN} vs. filtered \textit{RFN} (user: ‘TechCrunch’)</td>
<td>90</td>
</tr>
<tr>
<td>D.1</td>
<td>Summary Findings and Statistical Tests Used. (Source User: ‘Mashable’)</td>
<td>96</td>
</tr>
<tr>
<td>D.2</td>
<td>Summary Findings and Statistical Tests Used. (Source User: ‘BoingBoing’)</td>
<td>98</td>
</tr>
<tr>
<td>D.3</td>
<td>Summary Findings and Statistical Tests Used. (Source User: ‘TechCrunch’)</td>
<td>100</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

‘just setting up my twttr’¹
@jack Jack Dorsey

Twitter communities are groups of interconnected users that follow their interests through streams of information published in the Twitter social network and micro-blogging service. These information streams consist of flows of messages called tweets that are continuously posted by followees. Users manually adapt their streams by following such users and in this way try to follow their own interests. In some cases followers might receive, from within the information streams to which they are subscribed, messages they consider interesting and thus might decide to re-publish such messages in an event commonly referred to as a retweet.

User adoption of the Twitter service has skyrocketed since its launch in March of 2006. Currently, the service has more than 200 million users² together posting more than 140 million tweets per day.³ Tweets are short messages of 140 characters or fewer in length that contain text about various topics of potential interest and include username mentions, URLs and hashtags. The mechanics of spreading tweets in a social network of users who share similar interests has been described as follows:

“As a social network, Twitter revolves around ‘the principle of followers’. When you choose to follow another Twitter user, that user’s tweets appear in reverse chronological order on your main Twitter page. If you follow 20 people, you’ll see a mix of tweets scrolling down the page: breakfast-cereal updates, interesting new links, music recommendations, even musings on the future of education.”⁴ [our quotes and bold font]

Following someone means receiving their Twitter updates.⁵ This indicates an expression of interest towards that followee’s Twitter posts. However, it also means that the follower will receive each and every update from the given followee. Many tweets are bound to be uninteresting, as a result of adding more followees and hence

¹Twitter – The first tweet ever. https://twitter.com/#!/jack/status/20 (Retrieved Sep 2011)
(NB. Yes, ‘twitter’ was mispelled in the first ever tweet.)
⁴How Twitter Will Change the Way We Live – Time: http://www.time.com/time/printout/0,8816,1902604,00.html (Retrieved Aug 2011)
⁵Twitter Help Center – Frequently Asked Questions: https://support.twitter.com/entries/13920-frequently-asked-questions (Retrieved Sep 2011)
1.1 Motivation and previous work

receiving more tweets and also depending on the posting behavior of followees. Thus, Twitter users experience information overload. This defeats the purpose of following one’s interests on Twitter.6

Consequently, receiving updates from people by following them on Twitter can become overbearing. Users are increasingly flooded by minutiae and triviality. To make matters worse, Twitter does not provide user-friendly functionality for browsing through facets of tweets.7 Nor does Twitter enrich the semantics of posts and model users based on their micro-blogging activities in order to personalize their information streams. Furthermore, Twitter does not provide recommendations for news articles, despite the observation that over 85% of tweets posted everyday are related to news [31] and the observation that URLs extracted from a user’s close social group are more successful in content recommendation than the most popular URLs [15]. In this way the Twitter micro-blogging and social network service is broken. This leads to the main problem we will investigate:

Twitter does not provide support for automatically creating and maintaining lists of followees based on user behavior and understanding of user needs. Thus too many Twitter users and too many tweets quickly lead to information overload.

Recent Social Web research has aimed at improving upon these shortcomings of Twitter. For example, Celik et al. [14] investigated learning semantic relationships between entities on Twitter to support a faceted browsing experience that improves upon the chronologically ordered clutter option. Abel et al. [7] studied how to leverage Twitter activities for user modeling and evaluated the quality of user models in the context of recommending news articles. Abel et al. [8] introduced approaches for enriching the semantics of Twitter posts and modeling users based on their micro-blogging activities. These are just a few among a number of examples of research aimed at improving shortcomings of Twitter.

My thesis is that the shortcomings of Twitter can be improved by employing organizational principles to generate intelligent behavior useful in the construction of networks of followees. Examples of organizational principles include hierarchy and user interests; these are principles respectively embodied in hierarchical missing link prediction [17] and the link prediction approach to collaborative filtering [23]. I propose that organizational principles provide a better way than disconnected user intuitions to bring users together into networks of followers and followees, or what I call here ‘communities of interest.’ Thus this research is about the construction of communities. The improvements I propose are adaptive and personalized to the information needs of users and do not require maintenance from users, rather such improvements are inferred from collective user behavior. Furthermore, such improvements can be automated and relief users from having to manually maintain lists of followees as well as prevent them from information overload. Thus this research proposes algorithms to automate the construction of communities.

1.1 Motivation and previous work

In a prior literature survey we explored a novel perspective on recommender systems by surveying research on link prediction from the field of network analysis together with research on recommender systems. The novel perspective focused on looking at recommender systems through the lens of networks. The network lens transforms

---

6Follow your interests’ is Twitter’s motto: http://discover.twitter.com/ (Retrieved Aug 2011)
7While other software might provide such functionality, e.g. Tweetdeck, we focus here on the user experience offered by Twitter.
lists of users and items into interconnected communities of people and information. Communities are brought to life in social networks that evolve over time with fresh content, as happens every second of every minute in the Twitter micro-blogging and social networking service.

During discussions in research group meetings we identified an application of link prediction in which patterns of interaction in communities could be used to predict missing links that are unforeseeable now but that will likely arise in the future. Our motivation follows from this intuition. Namely we would like to uncover patterns of interaction that emerge from implicit connections between people and use these structures in the prediction of missing connections in a concrete and well-understood scenario that requires little interpretation. This scenario is captured in the concept of ‘communities of interest.’ Moreover, this scenario logically follows from the needs of user modeling and personalization in a fast changing environment. Thus our motivation is to develop a concept that is amenable and allows us to understand why new connections appear, why such connections were not present previously and why it is good to predict them.

In later discussions on research group meetings, it was suggested that the formulation of a scenario is important because it determines how relationships between entities are translated into a graph. Our motivation follows from this suggestion. We have identified translation as a key modeling step. Moreover, we realized that a concrete and well-understood scenario would allow us to evaluate success. Evaluating success is also part of our motivation, especially because it enables us to reflect on the effectiveness of our modeling strategy over the status quo (Twitter’s modeling strategy).

Previous work. In previous work we presented the most important discoveries, realizations and conclusions from our review of literature on recommender systems and link prediction. Our conclusions were based on comparing results and approaches found in literature with regards to the recommendation problem and the link prediction problem. Such conclusions serve as the connection between our survey of relevant literature and the present research project. As a continuation of our ideas for future research, the present research focuses on predicting new Twitter users to (automatically) follow based on the URLs that a particular user, her followees and followees-of-followees post on the Twitter social network and micro-blogging service (in what have been referred to as ‘social networks that matter’ [42]).

Motivation. Our motivation for the present research project follows from the following observations. First we observed that link prediction can be used to make recommendations. Second, we added that social networks provide a framework that allows us to leverage the connections between people and the content in which such people are interested. Third, we observed that such a framework can be extended by the application of link prediction methods. Fourth, from references suggesting that computational methods can inform social science theories we observed that the opposite might also hold. Social science theories can be used to evaluate computational methods such as recommender systems. More specifically, one such theory refers to the prevalence of hierarchy in social organization ([45], [46]), suggesting that knowledge of hierarchical structure can result in more accurate predictions for future relations in social networks. Another theory suggests that knowledge of user interests can result in more accurate predictions for future relations in social networks.8

---

8Although not a formal theory like that of hierarchy in social organization (e.g. [45]), Huang et al. [23] do show that the link prediction approach to collaborative filtering is effective in pre-
1.1 Motivation and previous work

Introduction

As explained above, our motivation is to formulate a concrete and well-understood scenario that allows us to reflect on the observations we made from literature. This scenario will focus on proposing a modeling strategy and evaluating its success. A central part of the modeling strategy is the application of a hierarchical missing link prediction algorithm [17] and the link prediction approach to collaborative filtering ([22], [23]). Thus our modeling strategy enables us to reflect back on the advantages of knowledge of hierarchical structure in social organization and the advantages of taking a link prediction approach to collaborative filtering, as has been suggested in literature.

Another source of motivation comes from the following observation of Simon [44]:

“A wealth of information creates a poverty of attention and a need to allocate that attention efficiently among the overabundance of information sources that might consume it.” [44]

As will be explained next, we see poverty of attention as one of the main limitations in Twitter. Our motivation is thus to overcome this limitation by engineering communities that can efficiently allocate attention by assisting users in finding information streams that are interesting to them.

1.1.1 Limitations of Conventional Information Streams

There are a number of shortcomings related to conventional information streams in Twitter. In the list below we bring forth the most significant limitations. Our aim is to compile a list of possible requirements and areas for improvement in Twitter’s modeling of information streams for its users.

- Manual follow/unfollow is not adaptive to the information needs of users.

- Information overload when following too many users or when followees produce too many updates as part of conventional information streams.

- Limited to non-existent attempt of understanding users information needs in terms of adaptive and personalized information streams based on user behavior.

- Recommendations for new followees are not explained to users.

- No preview (test) period for new followees.

- Not possible to select specific updates from followees based on user interests.
  - Not possible to select tweets or followees based on URLs posted on tweets.

- Not possible to learn from the (temporal) preferences of a user as expressed in lists of (latest) followees and (most recent) tweets.
  - Not possible to learn from patterns of URLs posted on tweets.

- No adaptive and/or personalized use of lists.

- No automated, adaptive and/or personalized construction of lists.

dicting future links in a network.
As the list above suggests there is room to improve upon the construction of information streams on Twitter. For instance, designing an algorithm for the construction of information streams such that streams become adaptive and personalized towards user interests could help address many of the limitations mentioned previously. Considering that user interests and user information needs change over time, it is sensible to make use of adaptive user modeling. Yet Twitter apparently ignores the needs of its users and the opportunity to use Social Web research results to improve its micro-blogging and social networking service. Until now, users must manually assemble information streams in order to follow their interests.

The same manual maintenance requirement applies when deconstructing or modifying information streams. Users must manually follow and unfollow other users in order to see changes in their information streams. And when it comes to lists, users must manually construct such lists by a similar process of adding and/or removing followees from such lists. At a more granular level, in both information streams and lists, it is not possible for users to select particular types of items they would like to see. Furthermore, it is not possible for users to express their interests and have Twitter automatically adjust to their preferences by presenting them with adapted and personalized information streams. This is the solution we seek, namely adaptive and personalized information streams that are automatically generated based on user behavior. We refer to these streams as alternative information streams and the larger whole to which they belong as alternative communities of interest.

1.2 Research gap

Recent research suggests that the structure of networks of followers and followees on Twitter reflects interpolation between properties of social and information networks ([31], [41]). For instance, friendship relations are mostly social while directed relations are mainly informational, but when information is forwarded among friends then interpolation happens and new friendships might arise as a result. So far, studies involving interpolation between social and information network properties on Twitter have focused primarily on the phenomenon of retweeting ([19], [41], [50]). Despite interesting results, other processes of interpolation between social and information network properties have hardly been studied. Neither has research focused on the effects of interpolation on social network structure and evolution. However, users interact every day on Twitter with friends and other users and interpolate URLs and meaningful content on their information streams by tweeting.

Recent research has found Twitter to be an unconventional social network with news media and information network characteristics ([31], [41]). For instance, Kwak et al. [31] found that Twitter exhibits properties that deviate from known characteristics of conventional social networks [37] (cited in [31]). Romero and Kleinberg [41] observed that Twitter is among social media sites that increasingly interpolate between the properties of social and information networks [41]. Among Twitter’s information network capabilities, the most studied has been the directed informational process of retweeting ([31], [50], [19], [41]).

Much research has focused on the phenomenon of retweeting. For example, Zaman et al. [50] worked on predicting retweets based on the interestingness of tweets. There have also been studies about predicting the URLs that are included in interesting tweets that are then retweeted [19]. Some researchers have studied the diffusion of retweets by constructing so-called ‘retweet trees’, which have been found to be shallow and with power-law distributed populations ([31], [19]). Other researchers, motivated by the analogy between link copying within Twitter’s informational follower network and the process of triadic closure in social networks,
have focused on link formation in Twitter’s follower network [41]. Yet others have studied the impact of network structure on the unfollowing of users [25].

Romero and Kleinberg [41] implicitly focused on retweets,9 probably because retweeting is the most common way in which underlying relationships among people in Twitter’s follower network are exposed. As explained in Twitter’s blog when the feature was introduced, retweeting clearly exhibits triadic closure through link copying within separate information streams:

“... Let’s say you follow @jessverr, @biz (that’s me), and @gregpass but you don’t follow @ev. However, I do follow @ev and the birth of his baby boy was so momentous that I retweeted it to all my followers.”10

A similar observation holds for research focused on the prediction of interesting content in Twitter. Not much progress has been made outside of prediction models and methods applied to retweets. For example, Galuba et al. [19] only focus on URLs in retweets despite the fact that URLs are also included in regular tweets. This in turn means that predictions that could be broader and more applicable by considering all tweets are confined to only look at retweets. Along the same lines, Zaman et al. [50] focus on predicting which content will be retweeted thus missing all of the other content on Twitter that could be interesting to users but is not retweeted.

Despite interesting results there has been little progress on other processes of interpolation in Twitter and their influence on social network structures and evolution. One exception is Chen et al. [16], who found that tie-strength-based algorithms perform significantly better in recommending conversations in online social streams for people who use Twitter for social purposes than for those who use it for informational purposes. However, the content of conversations in online social streams, including embedded URLs, has not been used to identify (meaningful) connections and make recommendations between people based on those connections. We propose that there is an opportunity to leverage content, more specifically embedded URLs and semantically enriched content [8], to identify connections between users and henceforth predict or recommend alternative information streams on Twitter. The problem we aim to solve is avoiding information overload and understanding user needs in an adaptive way based on the content users post.

Our proposed solution is based on the construction of triadic (user-object-user) networks where followers and followees are connected implicitly through embedded URLs and semantic entities, which are the conceptual ‘objects’ of discussion in tweets. The extracted URLs and semantic entities are then aggregated in dyadic (user-user) networks which are used to predict or recommend connections inferred based on user interaction through embedded URLs and in a more refined scenario, based on user interaction through embedded URLs found in tweets with semantically enriched content. Patterns of embedded URLs and semantic entities are then extracted from user profiles over time and used to measure interestingness of content on the network.

1.2.1 Research goal

In this research we study two alternative ways of building communities of interest. Namely we study predictions of followees made by the hierarchical link prediction algorithm of Clauset et al. [17] and predictions of followees made by a collaborative filtering recommendation algorithm. The reason we consider two approaches

---

9The word ‘retweet’ is not mentioned at all in [41]  
is because in literature there are indications that link prediction approaches can significantly outperform collaborative filtering approaches. However, it is not clear from literature whether hierarchical link prediction can outperform collaborative filtering or vice-versa. Furthermore, research has not been conducted to establish whether Twitter communities are hierarchically organized.

Huang et al. [22] found that path-based and neighbor-based approaches can significantly outperform the standard user-based and item-based collaborative filtering algorithms. However, there is no comparison in [22] of such approaches to hierarchical link prediction algorithms like the one of Clauset et al. [17]. On the other hand, Clauset et al. [17] found that the hierarchical method of link prediction outperforms link prediction methods, including path-based and neighbor-based approaches. However, there is no comparison in [17] between hierarchical link prediction and collaborative filtering. Therefore, we decided to undertake this comparison, albeit with a focus on determining which of the two methods is better for building communities of interest.

Our goal is to investigate whether different modeling strategies that are not focused on retweeting can be used to obtain more interesting content for Twitter users. We aim to accomplish this by comparing the interestingness of content found in patterns of tweets. These patterns were collected from different networks that we constructed based on the different modeling strategies, from two to three times per day, every day over a period from two weeks. These networks have been constructed through:

1. Explicit followee relations between Twitter users. (Defined manually by users in their respective followee lists.)

2. Predicted connections between Twitter users using the hierarchical missing link prediction algorithm [17].

3. Predicted connections between Twitter users using the link prediction approach to collaborative filtering ([22], [23]).

The first of these networks comes from the baseline provided by Twitter, which we consider a structural ground truth that provides us with a mechanism to structure networks. The second network was obtained by running the hierarchical link prediction algorithm of Clauset et al. [17]. The third network was obtained by running a collaborative filtering recommender algorithm applying the link prediction approach to collaborative filtering in which transitive user and item similarities are used to address the sparsity problem in collaborative filtering. This approach has been described as follows:

“It is our approach, collaborative filtering is studied in bipartite graphs. One set of nodes represents products, services, and information items for potential consumption. The other set represents consumers or users. The transactions and feedback are modeled as links connecting nodes between these two sets. Under this graph-based framework, we apply associative retrieval techniques, including several spreading activation algorithms, to explicitly generate transitive associations, which in turn are used in collaborative filtering.” [22]

“Our research dealt with the sparsity problem under a different framework. Instead of reducing the dimension of the consumer-product interaction matrix A (thus, making it less sparse), we proposed to explore the transitive interactions between consumers and items to augment the matrix A and make it meaningfully dense for recommendation purposes.” [23]
1.3 Introduction to communities of interest on Twitter

As an overlay pattern on the recommended and predicted networks we filtered tweets based on entities that were extracted from the community of followees of a Twitter user using the TweetUM webservice [14]. The overlay pattern is considered a network where interests are more refined based on semantic entities. These entities are used in a filtering mechanism that only considers tweets containing words from such entities. Therefore, URLs extracted from the different networks have a greater chance of being semantically related to a user’s interests as expressed by the community of users she follows.

Overall, our idea is to propose complementary filtering and routing mechanisms for Twitter based on the preference patterns of users as observed through the URLs they post on Twitter, the entities that semantically express their interests, and as predicted by a hierarchical missing link prediction algorithm and recommended by a collaborative filtering recommender system employing the link prediction approach to collaborative filtering.

1.2.2 Research questions

In this research we use the terms ‘community of interest’ and ‘network of followers and followees’ interchangeably. We construct conventional and alternative communities of interest over time. These are represented as networks in which we can study organizational principles, which are essential primitives underlying how a network is constructed. The two types of alternative networks we study are constructed through different organizational principles, namely user interest and hierarchy.

User interest is the basic organizational principle underlying predictions about future states of a network constructed through collaborative filtering. From here onwards we refer to this approach as ‘CF-based.’ Hierarchy is the basic organizational principle underlying predictions made about future states of a network constructed through hierarchical missing link prediction. From here onwards we refer to this approach as ‘LP-based.’ Thus, user interest and hierarchy are essential primitives. Our goal is to study whether algorithms employing these primitives over time can outperform the manual way of organizational networks of followers and followees which Twitter currently employs.

To evaluate how well the different networks perform with respect to organizational principles and in terms of interesting content we compare the relevance of content in different communities over time. This leads to our research questions, which we have formulated as follows:

1. Is the CF-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?
   1.1. Is the filtered CF-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?

2. Is the LP-based approach more effective in terms of the interestingness of content produced in communities of interest over time?
   2.1. Is the filtered LP-based approach more effective in terms of the interestingness of content produced in communities of interest over time?

1.3 Introduction to communities of interest on Twitter

Communities of interest on Twitter are groups of users who share common interests, which they explore by following each other and possibly other users. In this research we delimit communities of interest to only include users until followees-of-followees. Therefore, we extract URLs embedded in tweets only until the third level in a
network starting from a root user. We also extract the interests of the root user and from her followees through the TweetUM web service [48]. In combination, the extracted URLs embedded in tweets and the extracted user profiles of the root user and her followees enable us to construct networks of (user, item) pairs. Such networks contain the temporal interests of a group of users that we refer to as a community of interest.

We would like to compare the performance of conventional and alternative information streams in terms of the interestingness of content shared in corresponding communities of interest. The idea is to propose alternative information streams for a user and see if over time those streams would provide more interesting content for that user. However, it is important to make note of two conditions. First, our experiment covers an entire community of users. In other words, we will measure interestingness of content based on the URLs embedded in tweets that are posted by users in a community. Therefore, the measure of interestingness is reflective of the content that has been posted at the community level.

Second, the other condition we wish to note is that alternative information streams will be constructed each time we collect information. As mentioned previously such collection process will start from the root user. The consequence of constructing the alternative information streams each time is that the community of interest for a root user (also referred to as alternative community of interests) will change each time a model is constructed based on the alternatives that are found and used. We knew this beforehand and consciously decided to model alternative information streams in this way, especially because it introduces the key advantage of adaptiveness into our models. As user interests change over time, so will the predicted and recommended followees who share similar interests. As noted previously, the lack of adaptiveness and automation is a fundamental modeling and design flaw in Twitter information streams.

1.3.1 Modeling technology micro-bloggers

For this research project we selected three user accounts from competing technology news micro-bloggers (Tech Bloggers). We chose to model the activities and interests of these micro-bloggers because they are highly active on Twitter, both on the social network and micro-blogging aspects of the service. As show in Figure 1.2, these users post approximately 20 tweets per day, every day. Most of these tweets contain embedded URLs linking to relevant news about technology, from different perspectives (e.g. BoingBoing has a self-declared political interest in news items). The reason it is interesting to model these users is because they are leaders whose similarities and differences make them hard to compare, nevertheless each has its own active and evolving community of interest.

Another reason it is interesting to model technology bloggers is because it gives us impartial insights into the topics they tweet about and therefore into their interests. For example, BoingBoing has a self-proclaimed interest in left-wing politics, but what does this actually mean in terms of the items they post on Twitter? Mashable is supposedly the leader on social media news, but how is it different to TechCrunch in terms of interests within social media? The question of whether followees have any influence on the interests of Mashable, TechCrunch or BoingBoing, might also provide insights into the posting behavior of such users.

For our analyses we selected the following accounts:

1. Mashable (@Mashable)

“Digital, social media, business, tech, entertainment and mobile news from Mashable.com, the top resource for web culture. Updates from @mashable staff. http://mashable.com”

1
2. TechCrunch (@TechCrunch)


3. BoingBoing (@BoingBoing)

“The official BoingBoing.net feed: headlines and links to each blog post, video, and feature we publish. http://www.boingboing.net”

As shown in Figure 1.1, the user Mashable has the highest numbers with respect to number of tweets, number of followees, number of followers, and number of lists in which the account has been added. Next in numbers is TechCrunch, with lower numbers in every category in comparison to Mashable. BoingBoing on the other hand has the lowest numbers in all categories except number of tweets. In Figure 1.2 we see the tweets per day of the users ‘Mashable’ (Figure 1.2a), ‘TechCrunch’ (Figure 1.2b) and ‘BoingBoing’ (Figure 1.2c).

From the numbers in Figures 1.1 and 1.2, it seems that the Mashable user has the highest engagement with Twitter in all regards. And because of large differences in some numbers and odd disparities in others it would appear that comparing the performance of conventional and alternative information streams across these accounts would not be possible. However, in this research we have devised a strategy that enables us to compare different information streams across different users on Twitter. We will explain this strategy at length in the following chapters.

1.4 Relevance of this research

1.4.1 Scientific relevance

This research is scientifically relevant because it is an organized inquiry into the relevance of information and how well it meets the information needs of users in online social and information networks. Even though we focus on Twitter, the approaches and methods we propose in this research are applicable to any online social or information network. Furthermore, adding to the relevance of this research,
we propose alternate ways by which to obtain more relevant content from Twitter, thereby expressing an interest in challenging and changing the status quo.

This research aims to integrate and complement research in user modeling and personalization on Twitter ([1], [8], [7], [9], [48], [14], [6]). In addition, our general approach is aimed at bringing together management science, information science and computer science thus increasing the scientific relevance of this research. Furthermore, directions for future research in this project propose to employ methods of artificial intelligence, a branch of computer science that is relevant to both management science and information science.

1.4.2 Practical relevance

The information needs of users and the relevancy of content are intrinsically tied to how people communicate and organize into groups. Knowledge of such needs and requirements are thus essential components for managing emergent situations, which increasingly occur with the help of social media. An example that has become relevant with the recent London riots is inferring the organization of riots and criminal activities using social media. Through an understanding of how user’s organize into communities and how they sustain their behavior over time we can develop early warning systems for suspicious activities.

Inferring patterns of organization and behavior also raises ethical questions about user privacy and security. These are also essential components of a practical inquiry into social organization. Our approach inherently favors the analysis of behavior as it is occurring instead of shutting down social networks as some have recently proposed. We advocate the use of technology to monitor and make predictions about future states of social networks which can be extrapolated to real-world organizational settings.

One such setting is found in social media marketing. We believe this research is practically relevant because it can aid marketeers in making predictions about future states of a social network, which in turn can be used for market forecasting. Seeing how people organize and being able to make predictions opens up a range of possibilities in marketing. We believe this research contributes towards making this a reality.

Lastly, the social relevance of this research is found in its contemplation of the problems of recommendation and link prediction. We have developed a system that compares different algorithms and has the unifying goal of aggregating user preferences to inform individuals about future states of their social network. Given this opportunity we have also had a chance to carefully consider and even experience some of the subtle consequences that take place when sophisticated information systems are injected into a complex social problem (as suggested in [18]).

1.5 Structure of the report

This report is structured as follows. In Chapter 2 we survey research in user modeling and personalization, which serves as the foundation for this project. Chapter 3 explores the modeling of alternate information streams as networks using implicit, predicted and recommended connections (to be) found in online social and information networks. In Chapter 4 we construct these networks and setup data collection from Twitter. Then in chapter 5 we discuss the results from our measurements of interestingness of content in different communities of interest. Chapter 6 provides conclusions and proposes directions for future research.

The following questions provide an overview of the topics that will be explored in the next chapters:
1.5 Structure of the report

- **Chapter 2 - What are user modeling and personalization?**
  - What is hierarchical link prediction?
  - What is the link prediction approach to collaborative filtering?
  - How are user modeling and personalization related to link prediction and collaborative filtering?

- **Chapter 3 - How can we design communities of interest?**
  - How are hierarchical link prediction and the link prediction approach to collaborative filtering used in design models?

- **Chapter 4 - How can we engineer communities of interest?**
  - How are hierarchical link prediction and the link prediction approach to collaborative filtering used in such engineering process?
  - What challenges are encountered in engineering communities of interest and how can they be solved?

- **Chapter 5 - How can we measure the interestingness of information streams in communities of interest?**
  - What are the results of measuring the interestingness of alternate information streams in the communities of interest of selected Twitter users?

- **Chapter 6 - Are alternative communities of interest better or worse than conventional communities of interest?**
  - What are interesting directions for further research that can be proposed based on alternative communities of interest and alternative information streams?
Chapter 2

Related Work

‘Standing on the shoulders of giants.’

Isaac Newton

In this chapter we review state-of-the-art Social Web research focused on Twitter and state-of-the-art research on link prediction and recommender systems. We also examine Social Web methods and techniques, from which we will synthesize models for designing communities of interest. Link prediction is an essential component in these models. In this chapter we review the literature underlying the method of hierarchical link prediction and the link prediction approach to collaborative filtering.

2.1 Background on Social Web research focused on Twitter

The Social Web is becoming a highly interconnected Web in which applications produce, exchange, and reuse data from various sources. Chief among data sources are user activities and content. From systematic observations of user behavior and content produced by users it is possible to obtain knowledge about a user. For example, it is possible to learn about a user’s friendships, interests and preferences. The collection of knowledge about a user is stored in a user profile.

Among the data that can be collected for the construction of user profiles there are click-through data, search query logs, and data about user interaction with content and with other users. These are, among other types of data, the most important sources of knowledge about different aspects of users and their activities on the Social Web. Abel and Houben [1] focus on such data in their research. They consider Social Web data as a raw source of knowledge that is growing in number and scale. For instance, they write:

“The amount of user data available on the Web is tremendously growing so that sharing and mining these heterogenous data corpora distributed on the Web is a non-trivial problem that poses several challenges to the Web engineering community.”

Abel and Houben [1] identify two main challenges: information overload and understanding the demands of users [1]. The first challenge can be countered through adaptation and personalization, two mechanisms that allow a system to learn about the differences between individuals. In order for such mechanisms to actuate on heterogenous data corpora distributed on the Web, they must be provided with an understanding of user demands [1].
Social Web research focuses on helping overcome the challenges of information overload and understanding the demands of users, along with a number of associated challenges. These challenges can be summarized as follows:

- Analyzing user modeling within and between (across) Social Web systems ([7], [10]).
- Personalization of user experiences through different modeling strategies ([5], [9]).
- Learning semantic relationships between entities ([14], [6]).
- Application of user modeling strategies in the provision of services that exploit user activities to infer semantically meaningful profiles, such as the TUMS service (Twitter User Modeling Service) [48].

A seminal Social Web technique is semantic user modeling. The idea of semantic user modeling is to enrich content by referring to linked sources and then using enriched content in the construction of user profiles [8]. Semantic user modeling has proven especially useful in addressing the challenges of information overload and understanding of user demands on Twitter. This was recognized and proposed by Abel et al. [8]:

“Representing the semantics of individual Twitter activities and modeling the interests of Twitter users would allow for personalization and therewith counteract the [challenge of] information overload.” [8]

Among the main benefits of semantic user modeling on Twitter activities is the obtainment of high precision and coverage in the process of establishing relations between tweets and news articles [8]. Such enrichment of tweets provides a semantic extension on the construction of user profiles for the Social Web that enables the representation of individual Twitter activities in a semantically meaningful way [8].

2.2 Social Web research focused on Twitter

In this section we review state-of-the-art Social Web research focused on Twitter. We also examine hypotheses under investigation found in the publications we reviewed.

2.2.1 Semantic Enrichment for User Profile Construction

Abel et al. [8] introduce approaches for semantic enrichment of Twitter posts. As mentioned previously the main idea behind semantic enrichment is to enhance content by referring to linked sources extracted from URLs in Twitter posts or inferred from their content and then using the enriched content in the construction of user profiles [8]. In this way, Abel et al. [8] aim to support Twitter users in selecting information streams to follow or particular items within a stream which they may like or in which they may be peripherally interested. The need for such support is in line with the challenges of information overload and understanding of user needs in the Social Semantic Web. Furthermore, given factors that are specific to the Twitter micro-blogging service, Abel et al. [8] identified clear benefits in applying the method of semantic enrichment of Twitter posts. These factors include the variety and recency of topics discussed, the limited (140 characters) length of Twitter posts and the realtime nature and news media characteristics of the Twitter micro-blogging service [31].

The hypotheses under investigation in [8] include:
1. “The news-related user modeling strategy, which benefits from the linkage of Twitter messages with news articles, creates more valuable profiles than the tweet-based strategy.” [8]

2. “The semantic expressiveness of profiles generated with the news-based user modeling strategy is much higher than for the hashtag-based profiles.” [8]

2.2.2 Analysis of User Modeling

Abel et al. [7] study how to leverage Twitter activities for user modeling and evaluate the quality of user models in the context of recommendations for news articles. The rationale for focusing on recommendations for news articles comes from the observation that over 85% of tweets posted everyday are related to news [31] (cited in [7]). By focusing on news recommendations, Abel et al. [7] continue the investigation into semantic user modeling from [8]. There the authors proposed strategies for enrichment and contextualization of Twitter posts based on extracted URLs and, when no URLs were found, based on inferred content, both of which would ultimately link tweets to news items.

As discussed by Abel et al. [7], the focus on personalized news recommendations does not mean that the aim in [7] is to optimize recommendation quality. Rather, the goal of Abel et al. [7] in employing content-based or hybrid approaches is to analyze and compare the applicability of a set of proposed user modeling strategies in the context of news recommendations. Thus the context of ‘news recommendations’ is used as a means to compare the effectiveness of different types of user profiles. This is accomplished by keeping the recommendation algorithm constant and only changing the user modeling strategies (e.g. the user profiles used), and subsequently comparing the results.

The motivation of Abel et al. [7] is to go beyond hashtag-based user profiles and bag-of-words representations of tweets which are predominant in related research (e.g. [15] (cited in [7])). Furthermore, going beyond such representations in the construction of user profiles allows for exploration of how different semantic entity- and topic-based user modeling strategies impact the accuracy of recommending news articles. The analyses of Abel et al. [7] show that enriching tweets with URLs that refer to news articles significantly impacts user modeling and improves upon the construction of meaningful profiles than the strategy based on tweets alone.

Analysis of temporal dynamics

Abel et al. [7] and Abel et al. [9] analyze temporal dynamics in Twitter user profiles. Despite recent research showing that the consideration of temporal dynamics impacts recommendation quality significantly [30], the authors identified a lack of research on the impact of temporal characteristics of Twitter-based user profiles on recommendation performance. Furthermore, they found that little research has been done on understanding the semantics of individual Twitter activities and inferring (temporal) user interests from these activities. Abel et al. [9] suggest that such user interests can for example be used in modeling users as basis for personalized recommendations in different applications in the Web.

The hypotheses under investigation in [7] and [9] include:

1. Regardless whether the semantic enrichment method (introduced in [8]) might introduce a certain degree of noise – it impacts the quality of user modeling and personalization positively [7].

2. “The time-sensitive strategy characterizes the actual demands and concerns of a user better than the non-time-sensitive baseline strategy.” [9]
3. “Weekend profiles differ significantly from weekday profiles.” [9]

**TUMS: Twitter-based User Modeling Service**

Tao et al. [48] present a Twitter-based user modeling web service (TUMS) that exploits the activities of Twitter users with the aim to extract and infer user profiles that are semantically meaningful [48]. Various user modeling strategies are made available through the TUMS web service.\(^1\) These strategies are based on related research in semantic enrichment of individual tweets [14] and user modeling strategies [7]. Furthermore, Tao et al. [48] discuss the rationale for making the TUMS web service publicly available:

- *Giving Twitter users the ability to quickly grasp the content of their profiles* [48].
- *Enabling web applications to consume Twitter-based profiles in RDF format and thus help such applications deal with sparsity problems* [48].
- *Providing services with “realtime” or very fresh profile information from Twitter users* [48].

### 2.2.3 Analysis of cross-system User Modeling

Abel and Houben [1] provide initial motivation for cross-system user modeling and personalization based on solutions for user identification, social graph APIs, as well as frameworks developed for mashing up profile information ([2], [3], [4] (cited in [10])) and which furthermore facilitate the handling of aggregated user data. Abel and Houben [1] observe that given these developments it is important and feasible to analyze the nature of distributed user profiles and their impact on personalization in Social Web systems [1].

Abel et al. [10] observed that individual users are nowadays active in various social tagging systems. This distributed activity results in tag-based user profiles that are spread across different systems. Furthermore, Abel et al. [10] observed that other studies have not investigated the impact of user profiles on personalization and have not tested approaches on Social Web data where individual user interactions are performed across different systems and domains. In relation to these observations Abel et al. [10] questioned:

- *How user profiles differ or overlap across systems* [10].
- *How cross-system user modeling can support the construction of tag-based profiles* [10].
- *How such profiles impact the performance of recommender systems in cold-start situations in which user profiles are sparse* [10].

Abel et al. [10] investigate tagging behavior and user modeling across system boundaries with the aim of providing support towards the engineering of Social Web systems that aim for personalization. The researchers studied the behavior of a set of users active across a number of Social Web systems including Twitter, Delicious and Flickr. Their primary motivation was to beyond prior studies which have only been conducted in the context of particular (individual) systems (e.g. only on Delicious) [10]. The hypotheses that Abel et al. [10] investigated include:

\(^1\)The TUMS web application is accessible at: http://wisserver.st.ewi.tudelft.nl:8080/um-twitter-service/ (Retrieved Sep 2011)
1. “Cross-system user modeling strategies, within the scope of tag and resource recommendations in cold-start settings, have significant impact on the performance of the recommendation quality.” [10]

2. “Cross-system user modeling by means of aggregating the user-specific profiles from different platforms results in significantly more valuable profiles.” [10]

2.2.4 Applications of Learning Semantic Relationships between Entities

Celik et al. [14] discuss the lack of advanced browsing functionality in Twitter as one of the main motivations for their research. They identified a need for guidance from the users’ point of view in the exploration of entities in Twitter. Together with the feasibility of identifying entities and relations between entities in Twitter, Celik et al. [14] observed that learning semantic relations is an area of research that had not been studied yet.

Celik et al. [14] investigate learning semantic relationships between entities in Twitter. The authors identified semantic links between persons, products, events and other entities on Twitter. Along the same lines as proposed in [8], Celik et al. [14] found that the links they had discovered between entities become even more meaningful upon semantic enrichment. Thus they enhanced content in Twitter posts by referring to linked sources extracted from URLs and then used the entities that could be identified and exploited them for faceted search. In addition, establishing relations between such entities enabled the provision of recommendations among related entities.

Faceted Search

Abel et al. [6] propose strategies for inferring facets and facet values on Twitter by enriching the semantics of individual Twitter messages. The authors also present different methods, including personalized and context-adaptive methods, for making faceted search on Twitter more effective. The hypotheses under investigation in [14] include:

1. “The consideration of news articles improves recall and precision of the relation discovery clearly” [14].

2. Accuracy depends strongly on the type of relationships to be learnt [14].

2.3 Link Prediction

In this section we review the literature underlying the method of hierarchical link prediction. First we discuss how the link prediction problem has been formulated in research on networks. Then we go into depth on hierarchical link prediction, including some background on organizing principles in social networks. We discuss the hierarchical structure model and its relation to missing link prediction. As well presented in this section are limitations and advantages of hierarchical missing link prediction.

2.3.1 Link Prediction Problem

Link prediction is a way to formulate a problem such that methods of graph theory and social network analysis can be used to build predictive models. Predictive models look to the future, where links are predicted for objects that may not have been previously linked but for which predictions are based on observed links. Predicted
2.3 Link Prediction Related Work

Links may overlap with previously observed links or may be entirely new, meaning that missing links in networks are also candidates for prediction. Such overlap is mostly a matter of structure in the network. There are models that evaluate a wide range of network structures and within such structures sample the set of all possible network arrangements in accordance to an organizing principle (e.g., hierarchy in [17]).

The link prediction problem is the problem of predicting the existence of a relationship between two entities, based on object attributes and observed links in a network at a given time [38]. The links that are predicted can represent relationships that should be present but for some reason were not observed [20]. An example is inferring a complete network when only part of it could be observed [24]. Such a case of link prediction is called network completion [24]. Predicted links can also represent other types of relationships, such as new relationships, in which case the prediction task is referred to as missing link prediction [17], [38]. Depending on the type of network and model, missing link prediction and network completion may be different tasks. When time is part of the model, then missing link prediction and network completion refer to the same task which is to predict the evolution of a network in terms of new edges that will be added in the future.

Most research on link prediction refers to the more general formulation of the link prediction problem [38]. This formulation does not distinguish between predictions of missing links that should be in a network and those that are new to the network. Nowell and Kleinberg [38] formulate the link prediction problem as follows:

“Given a snapshot of a social network at time t, we seek to accurately predict the edges that will be added to the network during the interval from time t to a given future time t’” [38]

Link prediction is an estimate of the likelihood or probability of the future occurrence of a link in a graph. For example, a maximum likelihood approach is used in missing link prediction based on a model of how links are organized in a network. This model considers all the possible arrangements of a given network and the distribution of such arrangements across a range of possible network structures [17]. A maximum likelihood approach can also be used to predict false positives, which are links that are present but should not be present in a network. This is accomplished by looking at the minimum likelihood (lowest probability) of a link in a graph. A defining element of link prediction (as in [38], [17] and [13]) is that prediction methods are based purely on graph structure and focus on network evolution.

There are other elements that are important in link prediction. For example the type of network, the kind of edges (undirected vs. directed), and the nature of nodes in the network (homogeneous vs. heterogenous). Furthermore, links can be weighted or unweighted, and can be defined based on a certain threshold. The element of time is also crucial in the evolution of a network and hence on the relevancy and applicability of link prediction. For example the time period used to separate different snapshots of a network in order to study their evolution is an important modeling decision.

Another relevant aspect in link prediction is the class of nodes in a network. It is important to consider the entities that nodes represent (e.g., people, news, books, music, products), in order to appropriately make use of theories that explain things in the network. For example, homophily is the theory that in social networks people will make connections with others who are similar to them. Such a theory gives an expectation of behavior in a network. Kleinberg [28] proposes that theories from sociology and the social sciences can be informed by a computational perspective [28], which is to say that they can be employed and at the same time tested.
Related Work

2.3 Link Prediction

Networks and relationships. The main types of networks found in literature are social and information networks [18]. Furthermore, the link prediction problem was initially formulated for social networks [27]. Kleinberg [27] proposes that future research extend link prediction to information networks. In such networks, relationships between users would be established through intermediary entities such as items or products (e.g., items or products that a user might purchase). These networks are modeled using bipartite graphs, but can also be modeled using graphs and digraphs. Many of the link prediction methods in this section can be applied on social and information networks (e.g. [17], [13], [24]).

2.3.2 Hierarchical Link Prediction

Clauset et al. [17] present a general method for inferring hierarchical structure from network data that can be used in the prediction of missing links. The inference of hierarchical structure in the algorithm of Clauset et al. [17] is based on sampling a large number of dendograms with probability proportional to their likelihood [33]. The prediction of missing links is then calculated as the probability that two nodes are connected over all the sampled dendograms. The procedure is essentially a form of semi-supervised learning [43], in which network data represents labeled examples that are used to generate unlabeled network data represented in dendograms. From sampled dendograms the hidden hierarchical organization of the network can be uncovered and missing links can be predicted [33].

Clauset et al. [17] explain the usefulness of their technique for predicting missing links by showing that it can simultaneously explain and quantitatively reproduce the most important topological properties observed in networks. This includes properties such as right-skewed degree distributions (e.g., power-law distribution), high clustering coefficients and short path lengths [17] (which are characteristic of small world networks [49]). The technique proposed by Clauset et al. [17] hinges on hierarchy as the central organizing principle of complex networks. They propose that hierarchy can help explain the formation of communities in networks. Clauset et al. [17] find that the organizing principle of hierarchy is widespread in networks, to the extent that it leads to prediction of missing links in a range of different types of networks with more accuracy than competing techniques [38].

Organizing principles in social networks

Before the work of Clauset et al. [17] others have considered organizing principles in networks in order to explain formation and evolution (e.g. [35], [49], [26], [27]). The pioneer was Milgram [35], who conducted the first experiment to investigate the ‘small world problem.’ He found that information known locally to individuals forwarding a letter through acquaintances was effective enough to eventually have the letter delivered after only six hops. This finding became known as the principle of ‘six degrees of separation’, which says that everyone in the social network of the world can be reached through six intervening acquaintances, hence the name small-world. The small world property has been researched in online social networks like Facebook, for which the average separation was found to be 5.73 degrees. In a study of the Microsoft Instant Messaging (MSN) network, the average separation was found to be 6.6 degrees [32].

Milgram’s experiment and follow-up studies have demonstrated that there is an abundance of short paths that link individuals to each other and that groups

---

2 Wikipedia: Six degrees of separation

3 Hence the authors approximated separation to seven degrees and proposed the number six in ‘6 degrees of separation’ be changed.
of individuals are collectively very effective at finding short paths without having global knowledge of the network [18]. Further research continued to focus on short paths. For instance, Watts and Strogatz [49] set out to explain how short paths come into existence in networks. In doing so they explored the continuum between fully structured networks and random networks. Watts and Strogatz [49] found that the middle ground between such networks is a form of self-organization that had been previously found to be present in a variety of structures. For instance many biological, technological and social networks that lie somewhere in this continuum display high clustering coefficients and short average path lengths [49].

Watts and Strogatz [49] modeled small-world phenomena in networks. The term ‘small-world’ refers to situations in which the majority of links in a network are local but have at the same time resulted from long-range shortcuts. Watts and Strogatz [49] modeled such phenomena by introducing randomness into links using a ‘random rewiring procedure.’ The procedure starts from a ring lattice with \( n \) nodes and \( k \) links per node, and rewires nodes at random with a probability of \( p \) [49]. As a result of this procedure, Watts and Strogatz [49] found that a small number of long-range links out of each node would be added to destinations chosen uniformly at random, while the majority of new links would be created locally. Thus, the model of Watts and Strogatz [49] became known as the ‘small-world’ network model.

Structure in social networks. Easley and Kleinberg [18] describe modeling the process of decentralized search in social networks. They write that introducing a small adaptation to the Watts and Strogatz [49] model helps in understanding how structure influences decentralized search. The ‘small-world’ network model of Watts and Strogatz [49] has short paths that can be found using decentralized search. Kleinberg [29] found that by introducing a quantity that controls the “scales” spanned by long-range weak ties, the ‘small-world’ network model can generate networks in which decentralized search is more efficient. The following quote summarizes how changes in this parameter influence the structure of social networks such that the efficiency of decentralized search is impacted:

“When this correlation is near a critical threshold, the structure of the long-range connections forms a type of gradient that allows individuals to guide a message efficiently towards a target. As the correlation drops below this critical value and the social network becomes more homogeneous, these cues begin to disappear; in the limit, when long-range connections are generated uniformly at random, the result is a world in which short chains exist but individuals, faced with a disorienting array of social contacts, are unable to find them.” [29]

Applications in social networks. Easley and Kleinberg [18] write that social networks are conduits through which ideas and innovations flow through groups of people. They describe that people in groups are able to reach socially distant others, and in doing so engage in a kind of “focused search” [18]. Furthermore, as a result of the small-world phenomenon these groups are structurally similar: highly clustered and connected by very short paths. Through these paths individuals can conduct a focused search that is much more targeted than in other phenomena in networks, like diffusion.

The influence of organizing principles on the structure of social networks and on the effectiveness of social networks in distributed computations like decentralized search is apparent in the findings of Kleinberg [29]. One could say that Kleinberg [29] applied knowledge about the structure of social networks to the problem of decentralized search. Similarly, knowledge about the structure of social networks can be applied to other problems such as the recommendation problem. Research into
Related Work

2.3 Link Prediction

the hierarchical structure model [33] suggests this would be possible. Knowledge about structure has been applied in problems such as the missing link prediction problem. For instance, Clauset et al. [17] write:

“We further show that knowledge of hierarchical structure can be used to predict missing connections in partly known networks with high accuracy, and for more general network structures than competing techniques.” [17]

While problems that can be addressed by knowledge of structure in a social network vary in their aim and depend on certain characteristics which differentiate each problem, most problems have in common the orientation towards prediction of future states in a network. Taking this more general aim into account and considering that hierarchical structure has been found to be present along a range of different networks, one could say that hierarchical link prediction can be applied to many problems of prediction in networks. Lü and Zhou [33] propose this possibility and explain that hierarchical link prediction is a promising alternative to other methods that have been used for problems of prediction.

Hierarchical structure model

The hierarchical random graph model generates artificial networks with specified hierarchical structure [17]. The ability to generate hierarchical networks is used to sample the space of all possible hierarchical random graphs that occur in accordance to the likelihood that they generate a given network. In other words, a weight can be assigned to the probability that a certain model generates an observed network. And by sampling the space of all possible models (also called dendograms) it is possible to estimate the parameters that generate the given network.

Clauset et al. [17] explain that the comprehensive sampling of models allows the hierarchical random graph model to sense the different types of edges that can occur in a network. For instance, the model can differentiate between assortative and disassortative structures [17]. These are structures for which short paths are common within groups of nodes and sparse between them, in assortative networks, and the opposite with long paths in disassortative networks. The hierarchical random graph model is hence sensitive to the context of a network in terms of short or long edges. Small-world networks are a specific example in this continuum, situated towards the end of assortative networks.

Another aspect of the hierarchical random graph model is that it allows for hierarchical decomposition of a network [17]. This is a side effect of the ability to get a contextualized grasp of the structure of a network. Furthermore, hierarchical decomposition enables the resampling and matching of statistical properties like degree distributions, clustering coefficients, and distributions of shortest path lengths between pairs of nodes [17]. And this provides a secondary source of information that can be used to test the generating model and see if it has captured the key structural features of a network, thereby also attesting to its explanatory power [17].

Applications of the hierarchical structure model. Clauset et al. [17] discuss two main applications of the hierarchical structure model (which they call hierarchical random graph model). The first is a consensus structure that aggregates all of the sampled models into a single representation. This representation, called consensus dendogram, captures topological features that are consistently present in all models and therefore provides a better summary of network structure than a single model (dendogram) [17]. Furthermore, Clauset et al. [17] present an example
in which the consensus structure (dendogram) reveals communities and subcommunities. The second application of the hierarchical structure model is missing link prediction, which we discuss next.

**Missing link prediction.** The prediction of missing interactions in networks is another application of the hierarchical structure model [17]. Clauset et al. [17] propose the use of hierarchical structure in the prediction of missing interactions in biological networks. It is very costly to discover which interactions are missing and thus prediction based on known interactions can sharply reduce experimental costs [33].

In addition, missing link prediction can be used to predict the links that may appear in the future of evolving networks [33]. This is not explicitly affirmed or acknowledged in [17] but neither is it denied. Furthermore, Lü and Zhou [33] suggest evolution as a problem that fits naturally with link prediction methods such as missing link prediction. They further propose that information science and computer science can benefit from approaches employed in the study of complex networks, such as maximum likelihood-based methods like missing link prediction [33].

Clauset et al. [17] discuss the advantages of the hierarchical structure model. Among the advantages is more accurate prediction for a wider range of network structures than other methods [17]. This includes online social networks where not yet existent links can be recommended as promising friendships [33]. Furthermore, similar techniques can be applied to evaluate related evolving mechanisms that are related to friendship [33].

**The method of missing link prediction**

Clauset et al. [17] explain the workings of their proposed method for predicting missing interactions in networks, as follows:

1. Given a network generate a set of hierarchical random graphs that fit its structure.
2. Evaluate pairs of vertices with a high probability of connection within the sampled hierarchical random graphs.
3. Rank the results by sorting based on the probability of their occurrence.

Furthermore, Clauset et al. [17] also explain how their missing link prediction method can be evaluated. They propose using a standard metric for quantifying the accuracy of prediction algorithms, commonly referred to as the AUC statistic. The AUC acronym stands for ‘Area Under the ROC curve’, and is a metric commonly used in the machine learning and medical communities [17], [33]. The evaluation method under this metric consists of seeing the extent to which missing link prediction performs better than chance. Clauset et al. [17] conducted this test for three different networks and found that their algorithm indeed performs far better than chance. This result indicates that hierarchy is a strong general predictor of missing structure [17].

Clauset et al. [17] also compared the performance of their method of missing link prediction to other methods of link prediction, specifically those surveyed by Kleinberg [27]. The results of Kleinberg [27] are shown in Appendix A, in Figures A.1, A.2, and A.3, respectively in terms of common neighbors, graph distance and random predictor. Similarly, the results of Clauset et al. [17] are also shown in Appendix A, in Figures A.4, A.5 and A.6, respectively in terms of common neighbors, graph distance and random predictor.
As can be seen in Figures A.4 and A.6, hierarchical missing link prediction outperforms all other methods of link prediction. For these results Clauset et al. [17] carefully selected networks that exhibit structural properties which are characteristically assortative in some networks and disassortative in others. The choice of networks in [17] was meant to underscore the distinguishing characteristic of method of link prediction based on hierarchical structure, namely that it is sensitive to the structural context found at different levels in a network. Furthermore, this distinguishing factor explains the conclusion that “[t]he hierarchical method thus makes accurate predictions for a wider range of network structures than the previous methods” [17].

The reason the conclusion in [17] needs some explanation is that in other publications, such as [33], accuracy is discussed as a limitation of the missing link prediction method of Clauset et al. [17]. We discuss this limitation next, but first the explanation. The explanation is that Clauset et al. [17] refer to accuracy in terms of predicting links where such links should be predicted because of the structure of the network in which they are at work. Lü and Zhou [33], on the other hand, refer to accuracy in terms of a comparison against link prediction in networks where such distinction is not necessary and thus not made explicit, such as in the strongly assortative collaboration networks considered in [18] (as discussed in [17]).

Next we discuss advantages and limitations in more detail. We start with limitations.

Limitations

1. Time consuming - Algorithms can handle networks of up to a few thousand nodes.
2. Inaccurate - Methods perform poorly in comparison to similarity-based methods in terms of accuracy.
3. Produces poor predictions in some cases - When there is a lack of hierarchical structure, methods may produce poor predictions.

Lü and Zhou [33] describe the limitations of missing link prediction based on hierarchical structure (the method proposed in [17]). First, they write that algorithms for maximum likelihood like that in [17] can be very time consuming. Such algorithms can handle networks with up to a few thousand nodes, but are not able to scale up to millions of nodes [13]. Limitations in scalability are illustrated by the average-case complexity of order $O(N^2)$ needed to process sample dendograms for a Markov chain method, which in the worst-case has an exponential complexity. These limitations make algorithms like missing link prediction very time consuming, depending on the scenario and data being used [33].

Another limitation described in [33] is that maximum likelihood-based methods are not the most accurate. For example, when compared to similarity-based methods methods like missing link prediction based on hierarchical structure generally perform poorly [33]. It is important to note here our previous discussion regarding assortative and disassortative networks and the fact that accuracy has a different meaning in both. This appears to not have been considered by Lü and Zhou [33].

One more limitation discussed by Lü and Zhou [33] with regards to missing link prediction, is that when networks do not have a clearly identifiable hierarchical structure, the methods may produce poor predictions for missing links. Here again there is the risk of not carefully considering what good results mean and what they imply (similar to accuracy having a different meaning as discussed previously). Furthermore, Lü and Zhou [33] do not seem to take into account that the method of missing link prediction can be extended to incorporate domain-specific information [17]. This would probably be useful in cases in which there is no clearly identifiable
hierarchical structure and where domain-specific information could help fill that gap.

**Advantages**

1. *Uncovers hidden hierarchical structure* - Through its sampling approach, missing link prediction uncovers hidden hierarchical structure [33].

2. *Good performance for a wider range of networks* - Methods based on hierarchical structure have been found to outperform unsupervised link prediction methods by making accurate predictions for a wider range of network structures [17].

3. *Explicitly acknowledges many plausible hierarchical representations* - Missing link prediction explicitly acknowledges that most real-world networks have many plausible hierarchical representations of roughly equal likelihood [17].

4. *Ability to predict false positives* - Missing link prediction can predict false positive in networks where such links may be present [17].

5. *Extensibility to incorporate domain-specific information* - Missing link prediction can be extended to incorporate domain-specific information such as behavioral or node-specific features. [17].

An advantage of hierarchical structure model is that it uncovers the hidden hierarchical structure of networks it samples [33]. Moreover, through its sampling approach the hierarchical structure is a very good heuristic to employ. This is especially true based on it being used to quantitatively reproduce the most important structural properties in networks [17]. Furthermore, for a wider range of networks missing link prediction has been reported to outperform unsupervised link prediction methods by making more accurate predictions (note the previous discussion on the meaning of accuracy) [17]. This might be especially relevant upon reductions in network size or network summarization, as often advocated in literature [36], [38], [17], [18], because such reductions might interfere with the assortative or disassortative nature of the networks being studied.

Other advantages of missing link prediction include its explicit acknowledgement of many plausible hierarchical representations. By sampling these representations, missing link prediction avoids overfitting and enables understanding of various topological features [17]. Another advantage is that missing link prediction can predict false positives. This can be accomplished by looking for vertices with a low probability of connection which are nevertheless connected in an observed network [17]. In some cases connections can be explained by features outside of hierarchical structure. Missing link prediction caters to such need by allowing extensibility to incorporate domain-specific information from node features or behavioral characteristics in the observed network.

### 2.4 Collaborative Filtering

Collaborative filtering systems are based on the idea of identifying similarities between users through their ratings on items. These systems try to predict the utility of items for different users based on the items previously rated by other users [11]. Collaborative filtering systems have been defined as systems concerned with the process of identifying similar users and recommending what similar users like [40]. The main limitation of collaborative filtering systems is the dependence on abundant relations between users and items. When relations are sparse collaborative filtering systems perform poorly [11].
In this section we focus on a class of collaborative filtering systems which employ associative retrieval techniques in order to address the sparsity problem [22]. These systems apply the link prediction approach to collaborative filtering [22]. Such approach is based on exploring transitive user and item similarities in order to address the sparsity problem [23].

2.4.1 Link prediction approach to collaborative filtering

The link prediction approach to collaborative filtering is a combination of link prediction and collaborative filtering. The approach has two related parts or sides. The first is the application of graph-based algorithms to explore transitive user-item associations [22]. Huang et al. [23] describe this exploration as an attempt to make relations meaningfully dense for recommendation purposes. This aspect of the approach is meant to adapt collaborative filtering for link prediction, including representing relations in a graph. As described by Huang et al. [22]:

"Under this graph representation, the recommendation problem can be viewed as a task of selecting unobserved links for each user node, and thus can be modeled as a link prediction problem." [22]

The second aspect of the link prediction approach is the adaptation of link prediction algorithms to enable them to make collaborative filtering recommendations. This aspect is meant to adapt link prediction to collaborative filtering. Huang et al. [22] adapted a number of link prediction algorithms for making recommendations. They focused on the majority of algorithms surveyed in Kleinberg [27]. After adaptation of the link prediction algorithms, Huang et al. [22] studied linkage measures on a dataset about books and show that link prediction algorithms perform better than standard user-based and item-based collaborative filtering algorithms.

Huang et al. [22] describe their approach to collaborative filtering as follows:

"In our approach, collaborative filtering is studied in bipartite graphs. One set of nodes represents products, services, and information items for potential consumption. The other set represents consumers or users. The transactions and feedback are modeled as links connecting nodes between these two sets. Under this graph-based framework, we apply associative retrieval techniques, including several spreading activation algorithms, to explicitly generate transitive associations, which in turn are used in collaborative filtering." [22]

2.4.2 Limitations

The main limitations of the link prediction approach to collaborative filtering are the following:

1. **New User Problem** - new users must first rate sufficient items before recommendations can be issued.

2. **New Item Problem** - new items depend on past ratings by users in order to be recommended.

3. **Loss of information** - information is lost in the process of aggregation from triadic networks into dyadic networks.

The first two limitations are related to those described by Adomavicius and Tuzhilin [11] for collaborative filtering systems. The extent of both problems depends on the amount of historical data available users and items and also on the
extent of newness of such users and items. The third problem, namely loss of information, has been suggested to arise when information from bi-partite graphs is summarized into normal (one-mode) graphs [18]. In addition, assigning equal weights to edges in a graph regardless of historical importance (e.g. new edges might be more important than old edges) has been identified as an area for improvement graph-based representations and predictions [18].

2.4.3 Advantages

The main advantages of the link prediction approach to collaborative filtering are the following:

1. **Reduced sparsity** - exploring transitive relations between users and items reduces sparsity.

2. **Quality-based filtering** - preferences are more granular, despite being aggregated.

3. **Amenable to analyses over time** - preferences expressed over time lead to richer data.

4. **Serendipitous recommendations** - provide recommendations that users find valuable but which they did not expect.

The first advantage comes from the use of graph-based algorithms for exploring transitive user-item associations [22]. The reduction in sparsity comes from applying associative retrieval techniques to explore such transitive relations among users based on their past behavior [22]. The second and third advantages were identified by Kleinberg [27], who pointed out that bi-partite graphs have richer information that over-time analyses might be able to leverage even when such graphs are aggregated into normal (one-mode) graphs. The fourth advantage, namely serendipitous recommendations, was suggested by Herlocker et al. [21] and Aggarwal et al. [12] for collaborative filtering systems. It also applies to systems employing the link prediction approach to collaborative filtering because, like regular collaborative systems, such systems are also content agnostic and can have access to all items rated by users. Thus link prediction can take advantage of all relations between users and items in order to make predictions.
Chapter 3

Designing Communities of Interest

‘Everyone designs who devises courses of action aimed at changing existing situations into preferred ones.’

Herbert Simon

In this chapter we present our observations from related work. From these observations we synthesized a model and approach for designing communities of interest. We present and explain our model in this chapter. Our approach is based on key design decisions whose rationale we motivate and explain.

The aim of this chapter is to introduce our models for designing alternative communities of interest. These models hinge on the use of organizational principles for generating intelligent (communal) behavior. We focus on two organizational principles: hierarchy and user interest. Particular attention is paid to how those principles lead to different (competing) designs for alternative communities of interest, which we will compare in later chapters.

3.1 Observations from related work

Our observations concern two major topics: (1) Social Web research focused on Twitter and (2) research on link prediction and recommender systems. We treat each major topic in turn. First, within Social Web research, we examine the seminal method of semantic enrichment [8]. We continue by inspecting various more focused methods and techniques that depart from semantic enrichment and branch-off into specific investigation of other Social Web research interests related to Twitter. Finally, we examine hierarchical link prediction [17] and the link prediction approach to collaborative filtering recommender systems [22], [23].

3.1.1 Semantic enrichment of tweets to generate user profiles

Our observations are the following:

1. **Semantic enrichment, as presented in [8], is performed through linked sources that have been extracted from URLs embedded on tweets.**

2. **Semantic enrichment produces semantic entities that express the interests of users better than hash-tags or tweets alone (as shown in [8]).**

---

1It is also possible to infer linked sources, which in [8] is another way of ‘extracting’ URLs from tweets.
3.1 Observations from related work

3. Semantic entities are used in the construction of user profiles (as shown in [7], [8]).

4. Semantic enrichment provides a semantic extension to the construction user profiles (as applied in [7]).

5. Semantic user profiles from followees can be combined to infer the interests of a user based on the interests of followees.

Extrapolation.

User interests are expressed in particular followees that users have chosen to follow, and the content such users receive as a result of their followings is therefore interesting to them. By combining user profiles and employing semantic enrichment of tweets we can construct user profiles that can be used to infer communal interests. Such interests can in turn be used to establish linkages between users and henceforth observe organization in resulting communities of interest.

3.1.2 Consideration of temporal profile patterns

Our observations are the following:

1. Short-term interests are different than long-term interests, but users can express both types of interest at any time.\(^2\)

2. Short-term interests are modeled as topics and long-term interests as entities ([7], [9]).

3. Abel et al. [7] and Abel et al. [9] do not combine both short-term and long-term user profiles, nor do they consider both short-term and long-term profiles simultaneously.

Extrapolation.

User interests are temporally expressed in particular followees that users have chosen to follow, and the content such users receive as a result of their followings is therefore interesting to them at a given moment in time. By combining short-term and long-term interests and considering both types of interests simultaneously we can construct user profiles that benefit from advantages of both long-term and short-term temporal knowledge.

3.1.3 Adaptive faceted search based on URLs

Our observations are the following:

1. Adaptive faceted search can be performed through semantic enrichment (As suggested in [6]).

2. Faceted search depends on the extraction and expansion of (shortened) URLs.\(^3\)

3. URL facets enable faceted search to become adaptive. As user interests change, URL facets change.

---

\(^2\)In other words what differs is the frequency of expression of short-term vs. long-term interests.

\(^3\)This is not discussed in [6], but is an important step in this research as will be explained in Chapter 4.
4. The interests of users are organized around facets which include URLs. Therefore, user interests are to some extent organized around URLs.

5. The common ground in the search experience of different users is found in facets which include URLs. Therefore, users that share interests (what we call here ‘communities of interest’) can to some extent be found around URLs.

**Extrapolation.**

User interests are expressed in particular followees that users have chosen to follow, and the content, including URLs, such users receive as a result of their followings is therefore interesting to them. By extracting and expanding URLs we can employ them as facets and use them to search for new followees and thus obtain an understanding of user interests and of how communities around those interests are organized.

### 3.1.4 Link prediction and recommender systems

Our observations are the following:

1. The recommendation problem and the link prediction problem are not really different. Thus we can cast the recommendation problem into a link prediction problem.

2. The formulation of the recommendation problem in a graph is a framework that allows us to combine content-based and collaborative filtering systems.

3. The formulation of the recommendation problem in a graph is a framework that allows us to extend combinations of content-based and collaborative filtering systems through hierarchical link prediction [17] and through the link prediction approach to collaborative filtering [22].

4. Recommender systems can help inform social science theories and social science theories can help evaluate recommender systems.

5. Knowledge of hierarchical structure is powerful in link prediction because it can result in more accurate predictions of relations in a social network (as shown in [17]).

6. Knowledge of user interests is powerful in link prediction because it enables us to employ the link prediction approach to collaborative filtering which can in turn result in more accurate predictions of relations in a social network.

**Extrapolation.**

User interests obtained through semantic enrichment can be leveraged to identify implicit (conceptual) connections between followers and followees. By combining semantic enrichment with link prediction in social networks we can produce social networks, that represent communities with one source for cohesion (linkages), and extend such social networks into semantic social networks, which represent communities with two sources for cohesion (linkages and concepts).

---

4I acknowledge the commonality nowadays of graph-based recommender systems like PageRank or FolkRank. However, what I wish to show here is my thought process: (1) algorithms can be combined and (2) combinations can be extended.

5As proposed by Easley and Kleinberg [18].

6This is our observation. What we mean is that (informed) social science theories can set expectations for recommender systems.
3.2 Synthesized design approach

Having reviewed related work we are now ready to synthesize models for designing communities of interest. These models consist of key design decisions that are motivated by our observations of state-of-the-art literature and which we have extrapolated into models for designing communities of interest. Our models for designing communities of interest combine methods and techniques from the two major topics we studied in Chapter 2, namely (1) Social Web research focused on Twitter and (2) research on link prediction and recommender systems. The idea is to build a pipeline of methods that produce as output an alternative community of interest. First we explain our key design decisions, then in Section 3.2.2 we explain how these decisions are present in communities of interest and then in Section 3.2.3 we describe the resulting pipeline of methods.

3.2.1 Key design decisions

Below we list the key design decisions that were motivated by our observations of related work:

1. Collect latest followees and their user profiles and consider those as long-term patterns of interest.

2. Combine user profiles from a number of followees to infer the long-term interests of a user based on the long-term interests of followees.

3. Collect user profiles over time in order to adapt to long-term patterns of interest.\(^7\)

4. Collect most recent tweets and embedded URLs and consider those as short-term patterns of interest.

5. Collect tweets over time in order to adapt to short-term patterns of interest.

6. Construct networks according to the organizational principles of hierarchy and user interests.

7. Combine semantic enrichment with link prediction in social networks to produce semantic social networks.\(^8\)

As described in chapter 1, the mechanics of spreading tweets in a community of users who share similar interests revolves around the so-called ‘principle of followers.’ In this research we propose the use of organizational principles (hierarchy and user interests) to construct communities of followers and followees. We compare the performance of these different ‘principles’ based on the measure of interestingness of content (more details will be provided in Chapters 4 and 5). To accomplish this we collect tweets with embedded URLs and construct networks of (user, item) pairs which represent the state of a community of interest at a given time. We repeat this process over time in order to gain an improved understanding of the evolution of such state.

\(^7\) A longer time period of observation might be required for adaptation.

\(^8\) The idea here is to extend social networks based on implicit URLs to produce social networks based on semantic concepts.
3.2.2 Communities of interest

Communities of interest are groups of users that share the same interests. User interests are expressed in particular followees that users have chosen to follow, and the content such users receive as a result of their followings is therefore considered interesting to them. User interests are also expressed in the most recent tweets and embedded URLs that users post on Twitter.

Communities of interest can be designed in different ways depending on the organizational principle used to define their basic structure. Based on the key design decisions identified above we propose two alternative ways to design communities of interest. The reason we propose two alternative ways instead of only one is because the literature on link prediction is not clear as to whether hierarchical link prediction or the link prediction approach to collaborative filtering results in better predictions in social networks. There are indications that hierarchical link prediction is a superior method, but such indications carry strong requirements for the networks that are subject to prediction. While we believe that hierarchical link prediction results in more accurate predictions (as claimed in [17]), we question whether this method performs better than the link prediction approach to collaborative filtering. Our questioning is particularly focused on limitations of hierarchical link prediction and on the structure of communities of interest.

Communities of semantic interest

Communities of semantic interest are groups of users that share the same conceptual interests. Such interest are determined through semantic filtering. As before, user interests are expressed in particular followees that users have chosen to follow, and the content such users receive as a result of their followings is therefore considered interesting to them. Content is semantically filtered to determine its interestingness to users in a community. User interests are expressed in the most recent tweets and embedded URLs that users post on Twitter, which are also semantically filtered.

Steps to design communities of interest.

1. Collect a user’s list of latest followees and their user profiles (key design decision 1). We consider the interests of followees as long-term patterns of interest.

2. Combine semantic entities extracted from user profiles to form a user’s profile of long-term interests (key design decision 2). We believe that the interestingness of content will be strongly influenced by a user’s followees.

3. Collect user profiles over time in case followees change (key design decision 3). We consider this step necessary to introduce adaptation to long-term patterns of interest.

4. Collect the most recent tweets with embedded URLs from a user and her followees (key design decision 4). We consider the URLs as short-term patterns of interest.

5. Collect tweets over time (key design decision 5). We believe this step is necessary to adapt to short-term patterns of interest.

---

9 This is specified in observations 1-3 in the key design decisions list.
10 This is specified in observation 4-5 in the key design decisions list.
11 This means that only URLs on tweets with semantically interesting content will be considered for further analysis.
12 We extract the user profile of each user in a community of interest using the TweetUM web service [48].
3.3 Information Streams

6. Construct networks and make predictions based on the organizational principles of hierarchy and user interests (key design decision 6). We believe this step is necessary for adaptation of communities of interest.

7. Combine semantic enrichment with link prediction (key design decision 7). We believe this step is necessary for personalization of communities of interest.

3.2.3 Pipeline of methods

Next, we sketch out the pipeline of methods that we have put together based on key design decisions. Below we briefly present our proposed pipeline of methods.

A. Consideration of temporal profile patterns

• Periodically collect the latest thirty followees of a user and use them in the construction of that user’s community of interest, from which we will subsequently collect tweets.

• Periodically collect the three most recent tweets with embedded URL entities from a user’s community of interest.

B. Semantic enrichment of tweets to generate user profiles

• Periodically filter the three most recent tweets with embedded URL entities from a user’s community of interest. The entities extracted from these profiles are used for semantic filtering.

C. Adaptive faceted search

• Periodically generate facets based on URL entities embedded in tweets and use these facets to search for other users who posted the same URL entities in their tweets, and to whom the source (facet) user is thus implicitly connected.

D. Adaptation and Personalization of temporal networks of interest

• Periodically run a hierarchical link prediction algorithm based on implicit connections between users to obtain a predicted followee network.\(^{14}\)

• Periodically run a collaborative filtering recommender system algorithm based on implicit connections between users to obtain a predicted followee network.\(^{15}\)

3.3 Information Streams

3.3.1 Conventional Information Streams

Conventional information streams in Twitter are flows of tweets received by a user based on manually defined followees. This means that users who choose to follow others must manually define their followees on Twitter. As a consequence, unfollowing a user must also be done manually. Otherwise, the follower user will continue receiving each and every tweet produced by followees who are potentially no longer interesting.

In the long-term, follower users are interested in what they think a given followee represents. However, when user interests change over time then such subjective

\(^{13}\)Based on the user profiles extracted using the TweetUM web service [48].

\(^{14}\)We refer to these networks as LP-based networks.

\(^{15}\)We refer to these networks as CF-based networks.
judgements might also change; both in terms of the follower and followee. The same happens in the short-term. However, short-term interests are more volatile and noisy. Nevertheless, short-term interests can change consistently. In such cases, subjective judgements about interestingness are also open to change.

Long-term interests can be extracted as semantic entities through semantic enrichment [8]. Such entities can be used to model the long-term interests of users. Short-term interests, on the other hand, can be extracted from URLs embedded in tweets. Similarly, they can be used to model the short-term interests of users.

We refer to groupings of conventional information streams as conventional communities of interest. These are networks that over time represent the interests of users who utilize conventional means to structure their networks of followees. We refer to such networks as T networks. An illustration of a T network is shown in Figure 3.1. As shown, the solid circles surrounding groups of users represent the static relations that are established through fixed followings. Users receive content from their followees without any filters to evaluate the interestingness of content, thus all users are considered equal. This is illustrated by showing all followee users in the same (blue) color.

![Figure 3.1: Conventional Followee Network (T)](image)

### 3.3.2 Alternative Information Streams

Alternative information streams in Twitter are flows of tweets received by a user based on predicted followees. Followees are predicted through the hierarchical link prediction algorithm of Clauset et al. [17] (LP-based) and through the link prediction approach to collaborative filtering algorithm of Huang et al. [22] (CF-based). Predictions for the former are based on the hierarchical structure found in the implicit followee network. For the latter, predictions are based on user interests inferred through transitive user-item associations [23] found in the implicit followee network. The implicit followee network is constructed from URLs embedded in the tweets of a user and the URLs embedded in the tweets of implicit followees, up to a third level of depth in the network.

For both the LP-based and the CF-based predicted followee network, followees are automatically added to a user’s follower list. This means that users do not
3.3 Information Streams

Designing Communities of Interest

need to maintain their follower list, since such list will be maintained automatically and will thus change according to the long-term and short-term interests of users. Next we describe the prediction factors that uniquely differentiate the two types of alternative information streams and their filtered extensions.\(^\text{16}\)

**LP-based predicted followee network**

The following points describe what is special about the hierarchical link prediction (LP-based) alternative.

- *LP-based can simultaneously explain and quantitatively reproduce the most important topological properties observed in networks [17].* This includes properties such as right-skewed degree distributions (e.g. power-law distribution), high clustering coefficients and short path lengths. Consequently, predictions will likely identify (tweet) leaders related to short-term interests identified through URLs.

- *Hierarchy can help explain the formation of communities in networks [17].* This means that communities, sub-communities and smaller scale groupings will be identifiable within implicit followee networks. Thus, predicted connections will target followees who share the same short-term interests identified through URLs.

- *Hierarchy is widespread in networks, to the extent that it leads to prediction of missing links in a range of different types of networks with more accuracy than competing techniques [38].* This means that it will potentially be possible to make predictions across different types of implicit followee networks.

We refer to groupings of alternative information streams as alternative communities of interest. In the case of the LP-based approach, these are networks that over time represent the interests of users as determined through hierarchical relations which are used to structure their networks of followees. We refer to such networks as LP-based networks. An illustration of an LP-based network is shown in Figure 3.2. As shown, the broken circles surrounding groups of users represent the dynamic relations that are established through hierarchically predicted followings. Users receive content from their hierarchically predicted followees, who in turn receive content from their fixed followees. Thus hierarchy is used as a filter to evaluate the interestingness of content as determined from the structure of a network. This is illustrated by showing hierarchically predicted followee users in green and their fixed followees in blue.

**Filtered LP-based predicted followee network**

One of the main advantages of hierarchical link prediction is its extensibility to incorporate domain-specific information. This means that hierarchical link prediction can be extended to incorporate domain-specific information such as behavioral or node-specific features [17]. Node specific features, in the form of semantic entities extracted through TweetUM [48] are available for all networks in this project. Therefore, we can make use of such semantic information to further scrutinize relations in networks.

\(^\text{16}\)N.B. These are not assumptions or hypotheses that we will investigate in our analyses. Rather what follows after this note are observations from literature on hierarchical link prediction and the link prediction approach to collaborative filtering. These observations are meant to explain what is special about the LP-based and CF-based approaches we have chosen and thus serve as a motivation for why we chose these approaches.
CF-based predicted followee network

The following points describe what is special about the link prediction approach to collaborative filtering (CF-based) alternative.

1. **CF-based results in reduced sparsity.** This means that exploring transitive relations between users and items will result in reductions in sparsity and potentially improved quality in predictions.

2. **CF-based results in quality relations.** This means that preferences are more granular despite being aggregated.

3. **CF-based is amenable to analyses over time.** This means that preferences expressed over time lead to richer data about relations and interests over time.

4. **CF-based caters to serendipitous recommendations.** This means that CF-based can produce recommendations that users find valuable but which they did not expect.

We refer to groupings of alternative information streams as *alternative communities of interest*. In the case of the CF – based approach, *these are networks that over time represent the interests of users as determined through content relations which are used to structure their networks of followees*. We refer to such networks as *CF – based networks*. An illustration of a CF – based network is shown in Figure 3.3. As shown, the broken circles surrounding groups of users represent the dynamic relations that are established through predicted followings based on user interests. Users receive content from their predicted followees, who in turn receive content from their fixed followees. Thus user interest is used as a filter to evaluate the interestingness of content as determined from the structure of a network. This is illustrated by showing predicted followee users in purple and their fixed followees in blue.
One of the main advantages of the link prediction approach to collaborative filtering is that it summarizes a network based on domain-specific information. This means that collaborative filtering can be extended to incorporate domain-specific information such as behavioral or node-specific features through an associative retrieval framework. Node specific features, in the form of semantic entities extracted through TweetUM are available for all networks in this project. Therefore, we can make use of such semantic information to further scrutinize relations in networks.
Chapter 4

Engineering Communities of Interest

‘A wealth of information creates a poverty of attention and a need to allocate that attention efficiently among the overabundance of information sources that might consume it.’

Herbert Simon

In this chapter we present our approach towards engineering communities of interest. Our goal is the efficient allocation of attention in communities of interest. In coming up with this approach we encountered various challenges for which solutions were sought. We describe both in this chapter. In addition, we also describe the algorithms we designed for the construction of communities of interest. Design considerations included meeting time and complexity constraints, thus we describe those aspects as well. Furthermore, the algorithms we designed focus on automation of predictions, thus these aspects are also described in detail. Throughout this chapter we include code excerpts and where relevant we make reference to particular lines of code. All of the source code for our implemented algorithms is available in Appendix C.

4.1 Challenges and Solutions

Here we describe the main challenges and solutions in the construction of communities of interest. For some of the challenges we will include Python code and will describe the corresponding solution in reference to the code.

4.1.1 Search Strategy

In chapter 1 we formulated our main research problem as follows:

Twitter does not provide support for automatically creating and maintaining lists of followees based on user behavior and an understanding of user needs. Thus too many Twitter users and too many tweets quickly lead to information overload.

The solution we propose involves searching through tweets that are implicitly interconnected to other tweets through embedded URLs.\(^1\) However, the number

\(^1\)Users are the authors of tweets and thus they are also implicitly interconnected with other users, through embedded URLs.
of implicitly interconnected tweets and users grows exponentially and thus requires that we subdivide the problem into a tractable subproblem. The solution to the problem of intractability is described next in section 4.1.2. Another problem involves searching in the (reduced) space of interconnected tweets. Here we describe the search strategy employed after having defined a tractable subproblem.

Solution for search strategy. In order to implement a search strategy we devised a relaxed problem in which we only collect the latest followees of each user. This way we get a glimpse into her latest interests. A solution to the relaxed problem involved searching for implicit links between users. We selected depth first search as the search strategy used to solve the relaxed problem. We chose depth first search because temporal constraints are crucial, e.g. collecting the interests of second and third degree users immediately after we discover an implicit connection.

We applied depth first search as the search strategy for all three of our algorithms (T-based, CF-based and LP-based). More specifically, we implemented an iterative depth-first search algorithm to search through implicitly interconnected users and tweets. In our code, we iterate through the following functions:

- ProcessTweets (lines 1 to 14 in Figure 4.1)
- ProcessFriendsTweets (lines 16 to 46 in Figure 4.1)
- ProcessFriendsOfFriendsTweets (lines 48 to 72 in Figure 4.1).

Snow-ball data collection. As shown in Figure 4.1 (in the function calls in lines 9, 14, 40, 44, 68 and 72) networks are built in a snow-ball manner using the userid of both a user and her implicit followees (which are passed as parameters to each function). Which particular followees are selected depends on organizational principles implemented in the different algorithms. This means that we collect tweets from users, followees and followees-of-followees, according to different principles of followers (baseline, hierarchy, user interest) in depth-first order. Thus, our collection of tweets from a user leads us to other users (followees) who in turn lead us to yet other users (followees-of-followees).

4.1.2 Tractability

Addressing our main research problem quickly becomes an intractable problem itself, since the time required to solve instances of the problem grows exponentially with the size of the instances [43]. For example, when we collect followees from users there might be users with hundreds or even thousands of followees. Followees themselves might, in turn, have hundreds or thousands of followees (which we call followees-of-followees). Rather than solving this we chose to relax our problem by collecting only the latest followees of a root user and her followees. This approach, in turn, led to an insight for temporal user modeling, which we describe below.

Solution for tractability. In order to limit the space and time complexity of our algorithms, we devised a relaxed problem in which we only collect the latest followees of each user. This makes our solution tractable because we only consider a fixed number of followees for each user encountered through our search strategy. As well,
def ProcessTweets(self, user_id, tweets):
    self.output_logs.write("Entering ProcessTweets... \n")
    for tweet in tweets:
        if not(len(tweet.entities['urls']) == 0):
            # Add it to the unique URLs list
            if not(len(tweet.entities['urls'][0]['url']) == 0) and not(tweets_with_url > 3):
                # Extract URL from Tweet
                url = self.GetURLFromTweet(tweet)
                self.AddToItemList(user_id, tweet.user_id, url)
                tweets_with_url = tweets_with_url + 1
                filtered_tweet = self.FilterContent(tweet)
                if not(filtered_tweet is None):
                    self.AddToFilteredItemList(user_id, tweet.user_id, url)

def ProcessFriendsTweets(self, user_id):
    self.output_logs.write("Entering ProcessFriendsTweets... \n")
    try:
        friends_ids = self.GetUserFriends(user_id)
    except:
        pass
    if not(friends_ids is None):
        friend_circle = 0
        # For each friend, get their tweets and then extract URLs and add them uniquely to list
        for followee in friends_ids:
            try:
                followee_timeline = self.GetUserTimeline(followee.id)
            except:
                continue
            if not(followee_timeline is None):
                friend_circle = friend_circle + 1
                tweets_with_url = 0
                for tweet in followee_timeline:
                    if not(len(tweet.entities['urls']) == 0):
                        # Add it to the unique URLs list
                        if not(len(tweet.entities['urls'][0]['url']) == 0) and not(tweets_with_url > 3):
                            # Extract URL from Tweet
                            url = self.GetURLFromTweet(tweet)
                            self.AddToItemList(user_id, tweet.user_id, url)
                            tweets_with_url = tweets_with_url + 1
                            filtered_tweet = self.FilterContent(tweet)
                            if not(filtered_tweet is None):
                                self.AddToFilteredItemList(user_id, tweet.user_id, url)
            # Process the tweets of friends of friends of user
            self.ProcessFriendsOfFriendsTweets(followee.id)

def ProcessFriendsOfFriendsTweets(self, user_id):
    # Get the friends ids of a user
    try:
        friends_ids = self.GetUserFriends(user_id)
    except:
        pass
    if not(friends_ids is None):
        friend_circle = 0
        # For each friend, get their tweets and then extract URLs and add them uniquely to list
        for followee in friends_ids:
            try:
                followee_timeline = self.GetUserTimeline(followee.id)
            except:
                continue
            if not(followee_timeline is None):
                friend_circle = friend_circle + 1
                tweets_with_url = 0
                for tweet in followee_timeline:
                    if not(len(tweet.entities['urls']) == 0):
                        # Add it to the unique URLs list
                        if not(len(tweet.entities['urls'][0]['url']) == 0) and not(tweets_with_url > 3):
                            # Extract URL from Tweet
                            url = self.GetURLFromTweet(tweet)
                            self.AddToItemList(user_id, tweet.user_id, url)
                            tweets_with_url = tweets_with_url + 1
                            filtered_tweet = self.FilterContent(tweet)
                            if not(filtered_tweet is None):
                                self.AddToFilteredItemList(user_id, tweet.user_id, url)

Figure 4.1: Snow-ball collection excerpt from CFN.py
we only consider a fixed number of tweets from each user. Therefore, our solution is predictable in terms of time and space. Furthermore, our solution for tractability also introduces fairness into the comparison of different types of networks, since the same number of users and tweets are considered for each network.

As shown in lines 7, 37, and 66 in Figure 4.1, we limit the number of tweets with embedded URLs to no more than three (3). Similarly, as shown in lines 28 and 56 in Figure 4.1, we limit the number of implicit connections (or a circle of implicit friends) to no more than 30 followees.

**Temporal User Preference data collection.** As a result of relaxing our problem definition we devised a temporal data collection method. In other words, our data collection method takes time into consideration. First, when followees change over time our method collects the new followees and disposes the previous followees. Second, when users post new tweets we only collect the most recent of such tweets. In such case it is as if we ‘stretch time’ so that the same pseudo-time period applies to all users, where time is measured by tweets.

### 4.1.3 API limitations

The Twitter REST API is rate limited.\(^4\) As described in the ‘Rate Limiting’ documentation from Twitter:

- *Unauthenticated calls are permitted 150 requests per hour. Unauthenticated calls are measured against the public facing IP of the server or device making the request.*
- *OAuth calls are permitted 350 requests per hour and are measured against the oauth_token used in the request.*

Twitter’s Streaming API offers advantages with regards to rate limiting and access levels, but requires long-lived connections.\(^5\) Since we focus on short-lived connections we did not consider use of the Streaming API as an appropriate solution.

**Solution to API limitations.** Our solution to API limitations was to rotate among four accounts (using modular arithmetic\(^6\)) for the collection of tweets. Even though this is probably not considered proper use of the Twitter REST API, our focus here is on the technical solution in a proof-of-concept scenario. As shown in the code excerpt in Figure 4.2, we keep a count of the number of API requests (lines 6 - 9 in Figure 4.2) and when a limit of 325 API requests is reached we perform modulo addition on the account number (line 14 in Figure 4.2), which results in a ‘wrap-around’ among the four accounts used for collection.

### 4.1.4 Bag-of-words for semantic entities

As explained in chapter 3 (section 3.2.1), one of our key design decisions is to combine semantic enrichment with link prediction in social networks in order to produce semantic social networks. In the implementation of this design decision we used the TweetUM web service [48] to extract semantic entities from followees and combine them into a single ‘bag-of-words’ consisting of semantic entities (which we

---

\(^4\)Twitter developers – REST API Rate Limiting: https://dev.twitter.com/docs/rate-limiting#rest (Retrieved Sep 2011)

\(^5\)Twitter developers – Streaming API: https://dev.twitter.com/docs/rate-limiting#streaming (Retrieved Sep 2011)

\(^6\)Wikipedia – Modular Arithmetic: http://en.wikipedia.org/wiki/Modular_arithmetic
also call a ‘bag-of-semantic-entities’). In addition, another requirement we put forth was for a data structure that enabled the best look-up time for a large number of entities.

**Solution to bag-of-words for semantic entities.** As shown in the code excerpt in Figure 4.3, we build a query for the TweetUM webservice (lines 8 - 11 in Figure 4.3) and pattern match the results to extract semantic entities from the result set (lines 18 - 19 in Figure 4.3). As a data structure we chose sets in Python because we are not concerned with ordering the elements in any particular way and because sets, like dictionaries in other programming languages, offer constant time look-up [47].

### 4.1.5 Timeouts

One of the problems with Twitter’s REST API is that programs accessing the API have the risk of hanging unexpectedly. To avoid this we needed a timeout function that could be used across all the methods we use to access the Twitter REST API.

**Solution for Timeouts** We obtained the base timeout function from an advanced Python tutorial:

> “The idea is to have a general way to specify a maximum amount of time a function is allowed to run. (...) We want the final result to be a decorator which we could use with any function.”

The base timeout function is shown in the code excerpt in Figure 4.4. We modified the base function so that it would allow functions with arguments to be interrupted after a time period (line 13 in Figure 4.4). This way, methods that access the Twitter REST API would timeout after a specified time. Using this solution we were able to fully automate data collection for the different types of networks (T, LP − based, CF − based) without experiencing timeouts. Avoiding timeouts was thus a crucial solution for data collection.

---

4.1 Challenges and Solutions

Engineering Communities of Interest

Figure 4.3: Bag-of-semantic-entities excerpt from ICF.py

```python
def timeout(timeout_time, default):
    def timeout_function(f):
        def f2(*args):
            def timeout_handler(signum, frame):
                raise TimeoutException()
            old_handler = signal.signal(signal.SIGALRM, timeout_handler)
            signal.alarm(timeout_time) # triger alarm in timeout_time seconds
            try:
                retval = f(*args)
            except TimeoutException:
                print "TIMEOUT!"
                return None
            finally:
                signal.signal(signal.SIGALRM, old_handler)
                signal.alarm(0)
            return retval
        return f2
    return timeout_function
```

Figure 4.4: Timeouts excerpt from CFN.py

```python
def timeout(function):
    def f2(*args):
        def timeout_handler(signum, frame):
            raise TimeoutException()
        old_handler = signal.signal(signal.SIGALRM, timeout_handler)
        signal.alarm(timeout_time) # triger alarm in timeout_time seconds
        try:
            retval = function(*args)
        except TimeoutException:
            print "TIMEOUT!"
        finally:
            signal.signal(signal.SIGALRM, old_handler)
            signal.alarm(0)
        return retval
    return f2
```
4.1 Challenges and Solutions

4.1.6 Centralized processing of recommendations and evaluations

We faced the need to centralize collected networks in order to process recommendations and to evaluate precision in the different networks. This requirement could not be fulfilled in two of the three servers we used for data collection, because in those servers we did not have access to a java virtual machine. Thus we decided to centralize all processing of recommendations and evaluations on a single server. However, this brought forth a new requirement, namely that of non-interactive login for our network processing Python scripts (T, LP-based, CF-based).

Solution for centralized processing of recommendations and evaluations

We implemented non-interactive SSH login between all three servers. This allowed our Python scripts to automatically login and obtain recommendations and evaluations from collected networks. As shown in the code excerpt in Figure 4.5, we login through SSH and run the UserBasedEvaluator.java (code available in Section C.9 of Appendix C) which produces recommendations and evaluations (lines 13 - 14 in Figure 4.5).

4.1.7 Monitoring and logging

Since our collection scripts were running up to three times per day for the three different kinds of networks (T, LP-based, CF-based), this brought forth the challenge of monitoring and logging. We needed to continuously check all the information produced by our scripts in order to ensure that in a period of two weeks data collection would make progress without any problems.

Solution for monitoring and logging

We implemented e-mail notification and logging of important information to files. This allowed our scripts to append log files that we would later review upon collection of networks. Furthermore, once each script finished it would send an email summary as shown in Figure 4.6. This functionality was implemented as shown in the code excerpt in Figure 4.7.

4.1.8 URL extraction and expansion

All of the URLs embedded in tweets are shortened in some way. The URLs are often shortened by users themselves and in all cases URLs are shortened by Twitter.
as URLs under the t.co domain. This means that there can be several layers of shortened URLs. For example, a user can shorten a URL with bit.ly and the shortened URL will in turn be shortened with t.co. Thus we faced the challenge of going through several layers of URL shortening in order to extract the real URLs embedded in tweets.

**Solution for URL extraction and expansion** We were able to extract URLs embedded in tweets by accessing the tweet’s entity fields. These fields contain the shortened URL and in cases when such URL was only shortened with t.co then the expanded URL would also be available. However, when another shortening layer was used (e.g. bit.ly) then we needed to further expand that shortened URL. We accomplished further expansion of shortened URLs by using three expansion

---

8bitly – Shorten, share and track your links: https://bitly.com/ (Retrieved Sep 2011)
9Twitter – t.co: http://t.co/ (Retrieved Sep 2011)
10Twitter developers – Twitter API: https://dev.twitter.com/docs/tweet-entities (Retrieved Sep 2011)
def GetURLFromTweet(self, tweet):
    if not(len(tweet.entities['urls']) == 0):
        if not(len(tweet.entities['urls'][0]['url']) == 0):
            if (self.expander_service == 0):
                real_url = self.ExpandURL(url_posted)
            elif (self.expander_service == 1):
                real_url = self.LongURL(url_posted)
            elif (self.expander_service == 2):
                real_url = self.LongURLPlease(url_posted)
        else:
            url_posted = tweet.entities['urls'][0]['url']
            real_url = url_posted
        if (real_url == url_posted):
            if (self.expander_service == 0):
                real_url = self.ExpandURL(url_posted)
            elif (self.expander_service == 1):
                real_url = self.LongURL(url_posted)
            elif (self.expander_service == 2):
                real_url = self.LongURLPlease(url_posted)
    else:
        url_posted = tweet.entities['urls'][0]['url']
        real_url = url_posted
    if (real_url == url_posted):
        try:
            shortened_url = urllib2.urlopen(url_posted)
            real_url = shortened_url.url
        except:
            real_url = url_posted
            expanded_url = real_url

Figure 4.8: URL extraction and expansion excerpt from CFN.py

However, a further challenge with the use of expansion services was that such services are often rate limited. Thus we needed to detect when such rate limiting had occurred and switch to another expansion service without loosing any URLs in the process. The solution we implemented is similar to the one above for Twitter’s rate limited REST API. Our solution was to rotate among four expansion services (using modular arithmetic\(^{14}\)) for the expansion of shortened URLs extracted from tweets. Even though this is probably not considered proper use of the expansion services, our focus here was on the technical solution in a proof-of-concept scenario. This functionality was implemented as shown in the code excerpt in Figure 4.8.

4.1.9 URL filtering

In the construction of networks of interest (\(T\), \(LP\)-based, \(CF\)-based) we were interested in assigning unique identifiers to users and the URLs embedded in their tweets. For users this assignment was straightforward as each user on Twitter

\(^{11}\)http://expandurl.appspot.com/ (Retrieved Sep 2011)
\(^{12}\)http://longurl.org/ (Retrieved Sep 2011)
\(^{13}\)http://www.longurlplease.com/ (Retrieved Sep 2011)
\(^{14}\)Wikipedia – Modular Arithmetic: http://en.wikipedia.org/wiki/Modular_arithmetic
already has a unique user identifier. Thus, for users we extracted their **userid** and used it as a unique identifier. For URLs, however, this was not as straightforward.

There are several ways in which to assign unique identifiers to URLs. Expanded URLs sometimes include paths, parameters and fragments. The problem is that users shorten URLs in different ways. Some users shorten full URLs (including paths, parameters and fragments), while other users shorten only part of the full URL.

**Solution for URL filtering.** After considering alternative pre-processing solutions (e.g. shortest match and longest match) we decided to filter URLs without pre-processing. We came to this solution after reviewing preliminary data, in which we found that differences in expanded URLs were few and thus did not merit the effort to implement URL pre-processing functionality. Therefore, the solution we implemented consists of checking whether a URL is already in the unique URLs list and if not then adding it to the list. This functionality was implemented as shown in the code excerpt in Figure 4.9.

**Solution for semantic URL filtering.** A similar solution was implemented for semantic URL filtering. As described in section 4.1.4, we extract semantic entities using the TweetUM web service and use those entities to build a set. For semantic URL filtering we break down a tweet into words and check whether those words can be found in the bag-of-semantic-entities (the set mentioned previously). If found then the URL is added to the semantic URL filtered list. This functionality was implemented as shown in the code excerpt in Figure 4.9.
def AddToItemList(self, parent_user_id, child_user_id, url):
    try:
        self.output_logs.write("From " + "%s" %child_user_id + "'s profile \n")
        print "From " + "%s" %child_user_id + "'s profile"
        self.output_logs.write("URL: " + "%s" %url + "\n")
        print "URL: " + "%s" %url + "\n"
    except:
        pass

# Add user_id to Users list, if not there already
if (self.users.count(child_user_id) == 0):
    self.users.append(child_user_id)

# Put URL in items list if it's not there already
if (self.items.count(url) == 0):
    self.items.append(url)

# Add URL to total items
self.total_items.append(url)

if not(url is None):
    try:
        self.output.write("%s" %child_user_id + "%,s" %self.items.index(url) + "\n")
        self.output.write("%s" %parent_user_id + "%,s" %self.items.index(url) + "\n")
    except:
        pass

def AddToFilteredItemList(self, parent_user_id, child_user_id, url):
    try:
        self.output_filtered_logs.write("[FILTERED] From " + "%s" %child_user_id + "'s profile \n")
        print "[FILTERED] From " + "%s" %child_user_id + "'s profile"
        self.output_filtered_logs.write("URL: " + "%s" %url + "\n")
        print "URL: " + "%s" %url + "\n"
    except:
        pass

# Add user_id to Users list, if not there already
if (self.filtered_users.count(child_user_id) == 0):
    self.filtered_users.append(child_user_id)

# Put URL in items list if it's not there already
if (self.filtered_items.count(url) == 0):
    self.filtered_items.append(url)

# Add URL to total filtered items
self.total_filtered_items.append(url)

if not(url is None):
    try:
        self.output_filtered.write("%s" %child_user_id + "%,s" %self.filtered_items.index(url) + "\n")
        self.output_filtered.write("%s" %parent_user_id + "%,s" %self.filtered_items.index(url) + "\n")
    except:
        pass

def FilterContent(self, tweet):
    words_in_tweet = tweet.text.split(' ')
    for word in words_in_tweet:
        if word in self.tweetum_entities:
            try:
                self.output_filtered_logs.write("TweetUM Match! Matched with word: %s" %word + "\n")
            except:
                pass
    return tweet

Figure 4.9: URL filtering excerpt from CFN.py
4.2 Engineering Information Streams

4.2.1 Conventional Information Streams

To build a $T$ network we collect the 30 latest followees of a given (source) user and extract the three (3) most recent tweets with embedded URLs. The reason we chose to extract three tweets each time is that we wish to restrict the amount of time it will take to build a $T$ network. On average we found that it takes between three to five hours to collect a complete $T$ network.

The complete implementation of the $T$ network in Python is included in Appendix C (Section C.1). A $T$ network is built up as follows:

1. Collect the three most recent tweets with embedded URLs posted by a source (root) user.
2. Collect the thirty latest followees of a given user.
3. Collect the three most recent tweets with embedded URL entities from each of the latest followee users based on the search results from step 2.
4. Repeat steps 1 - 3 for each second level followee user.
5. Stop at the third level of followee users. Only collect the three most recent tweets with embedded URLs.

Figure 4.10 shows an overview of the data set collected for the user ‘mashable.’ Data was collected three times per day, every day during two weeks.

![Figure 4.10: ($T$-based network) - Dataset Overview](image)

4.2.2 Alternative Information Streams

LP-based network

In the construction of an $LP$-based network we use as input an implicit followee network ($IFN$). This is the network of implicit connections obtained through the
iterative search of URLs embedded in the most recent tweets of a group of users. Once we have constructed the IFN, we use it as input for the hierarchical link prediction algorithm of Clauset et al. [17]. This algorithm gives us a third network, which we refer to as the \( LP - \text{based} \) network. In such network, we have access to the ranked predicted connections for all users from which we extract followees in the order shown below:

1. We look for the first 30 followees with \( P > 0.0 \)
2. We look for the first 30 followees of implicit followees with \( P > 0.0 \)
3. We use the 30 implicit followees of the source user when there are no followees found in steps 1 and 2.

The resulting \( LP - \text{based} \) network is the ordered list of users with whom a given user will likely be connected in the future. As explained above in the absence of predictions we use that user’s implicit followees (from the IFN) as the source for predictions and otherwise we use the implicit followees themselves as predictions. Predicted connections are the input for a collection mechanism that is similar to that of the \( T - \text{based} \) network. The mechanism is similar in the sense that we take the 30 latest followees of each of the predicted followees and we collect their three (3) most recent tweets with embedded URLs. The complete implementation of the \( LP - \text{based} \) network algorithm in Python is included in Appendix C (Section C.3).

An \( LP - \text{based} \) network is built up as follows:

1. Collect the three (3) most recent tweets with embedded URLs.
2. Construct the IFN.
3. Convert the (user*item) pairs in matrix \( A \) into (user*user) pairs (e.g. by matrix multiplication: \( IFN2PFN = A \ast A^T \)). This means that users who were implicitly connected through shared items will now be explicitly connected.
4. Run the hierarchical link prediction algorithm on \( IFN2PFN \), the result will be an \( LP - \text{based} \) network.
5. Go through the \( LP - \text{based} \) network and extract:
   - The first 30 followees with \( P > 0.0 \)
   - The first 30 followees of implicit followees with \( P > 0.0 \)
   - The first 30 implicit followees in the absence of predictions
6. Collect the three most recent tweets with embedded URLs from each of the latest 30 predicted followees.
7. Repeat steps 1 - 3 for each second level followee user (ie. followees of predicted followees).
8. Stop at the third level of followee users (ie. followees of predicted followees). Only collect the three (3) most recent tweets with embedded URL entities from such followees.

Figure 4.11 shows an overview of the data set collected for the user ‘mashable’ using the \( LP - \text{based} \) strategy. Data was collected three times per day, every day during two weeks.
4.2 Engineering Information Streams

Engineering Communities of Interest

Figure 4.11: (LP-based network) - Dataset Overview

CF-based network

After having constructed the IFN we use it as input for a link recommendation algorithm. This algorithm employs a collaborative filtering recommender system, which gives us a second alternative network that we refer to as the CF-based network. In such network, we have access to the predicted connections for all users in the CF-based network from which we extract the first 30 recommended followees. The complete implementation of the CF-based network algorithm in Python is included in Appendix C (Section C.1).

A CF-based is built up as follows:

1. Collect the three (3) most recent tweets with embedded URLs.
2. Construct the IFN.
3. Convert the (user*item) pairs in matrix A into (user*user) pairs (e.g. by matrix multiplication: \( IFN2RFN = A \times A^T \)). This means that users who were implicitly connected through shared items will now be explicitly connected.
4. Run the link recommendation algorithm on IFN2RFN, the result will be a CF-based network.
5. Go through the CF-based network and extract:
   - The first 30 followees.
   - The first 30 implicit followees in the absence of recommendations.
6. Collect the three most recent tweets with embedded URL entities from each of the latest 30 predicted followees.
7. Repeat steps 1 - 3 for each second level followee user (ie. followees of predicted followees).
8. Stop at the third level of followee users (i.e., followees of predicted followees). Only collect the three most recent tweets with embedded URL entities from such followees.

Figure 4.12 shows an overview of the data set collected for the user ‘mashable’ using the $CF_{-based}$ strategy. Data was collected three times per day, every day during two weeks.

![Figure 4.12: (CF-based network) - Dataset Overview](image-url)
Chapter 5

Evaluating Communities of Interest

“If you can measure it, you can improve it.”
Lord Kelvin

In this chapter we describe our research methodology. We explain how and why we measure interestingness of content over time in conventional and alternative information streams found in communities of interest. Furthermore, we explain the evaluation strategy employed in this research. Following from the research methodology we present and discuss the results of this research.

5.1 Research Methodology

In this section we concern ourselves with answering the following question:

How do the different algorithms for constructing communities of interests impact the behavior and performance of (collaborative) recommender systems?

We focus on the behavior and performance of (collaborative filtering) recommender systems because they can be straightforwardly operationalized and measured. Furthermore, we use the behavior and performance of (collaborative filtering) recommender systems as proxies for interestingness of content. Answering the question above will help us answer the research questions posed in Chapter 1 by comparing the different algorithms for constructing communities of interest that were proposed in Chapters 3 and 4.

In our experiment, a collaborative filtering recommender system predicts the behavior of users in relation to the content they find interesting. As we observed in Section 3.2.1, relevance judgements can be automated based on the structural characteristics and semantics found in the information streams that comprise a network. Therefore, the performance of (collaborative) recommender systems can be used to evaluate the interestingness of content that would result from behavioral predictions about future followings. In other words, by evaluating the precision in predicting items that would be consumed by the user if she would be organized within the predicted network we can compare different networks to see which is better in terms of interestingness of content.
5.1 Research Methodology

5.1.1 Measuring Relevance

Manning et al. [34] describe various approaches to information retrieval (IR) system evaluation. The approach they consider to be standard is based on the notion of relevant and non-relevant documents [34]. In this approach a collection of documents and one or more information needs expressible as queries are used together with a set of relevance judgements to assess whether query-document pairs are relevant. Manning et al. [34] explain that the “gold standard” or ground truth judgement of relevance in IR is the classification of documents in a test collection as being either relevant or non-relevant with respect to a user information need. Based on such classification they explain the advantage of IR system evaluation:

“The advantage of system evaluation, as enabled by the standard model of relevant and nonrelevant documents, is that we have a fixed setting in which we can vary IR systems and system parameters to carry out comparative experiments” [34]

Social Web research focused on Twitter employs a flavor of IR system evaluation. For example, Abel et al. [7] measure quality in user profiles by keeping a fixed recommendation algorithm in which content considered to be relevant for a specific user is obtained from linkages to preselected sources (preselected by the researchers). This allows the researchers to compare the quality of different types of user profiles based on measures such as MRR (Mean Reciprocal Rank) and S@k (Success at rank k). To achieve such comparison the recommendation problem is cast into a search and ranking problem in which user profiles are interpreted as queries. The ground-truth in [7], is obtained through a prejudgement of relevance which classifies tweets based on whether they link to relevant news sources. Russell and Norvig [43] refer to this kind of judgement as a “human relevance judgement” [43].

5.1.2 Measuring Relevance in Information Streams

We take a different approach towards IR system evaluation. Instead of conducting an expansive and time consuming user study, we automatically determine relevance for users according to the structure of networks of (predicted) followees in which users are organized. In other words, we prejudge all items that are incoming from the information streams of (predicted) followees, as relevant. This design choice follows from our observation (found in Section 3.2.1 under ‘communities of interest’) that user interests are expressed in particular followees that users have chosen to follow, and the content such users receive as a result of their followings is therefore interesting to them.

In a more refined version of our approach, we automatically determine relevance for users according to the semantics of networks of (predicted) followees in which users are organized. In other words, we prejudge all items that are incoming from semantically filtered information streams of (predicted) followees, as relevant. This design choice was also specified in our observations in Section 3.2.1 under ‘communities of semantic interest’.

Evaluation Approach

Our approach is based on conducting a simulation in which it is possible to:

Structure a network such that we can measure the precision in predicting items that would be consumed by the user if she would be organized within

\(^1\text{Manning et al. [34] use the term ‘gold standard.’}\)
the different networks of (predicted) followees, namely the $T$, $LP$-based and $CF$-based networks.

We chose this approach because it allows us to automatically determine relevance in networks without the need for conducting an expansive and time consuming user study that would require human relevance judgements [43] across thousands of users who constantly change in networks of predicted followees. By following this approach we are actually measuring potential precision, which we call ‘interestingness’:

The (potential) precision in predicting items that would be consumed by the user if she would be organized within a given (predicted) followee network.

In contrast to conducting a user study, we do not define good/relevant recommendations beforehand. The question that follows is what exactly is a good recommendation? Owen et al. [39] provide an answer to this question for the Mahout framework, which is the recommender technology used in this research. They write:

"But what exactly is a good recommendation here? The framework was asked to decide – it didn’t receive a definition. Intuitively, the most highly preferred items to the test set are the good recommendations, and the rest aren’t." [39]

"The issue is further complicated when the preferences are Boolean and contain no preference value. There isn’t even a notion of relative preference on which to select a subset of good items. The best the test can do is randomly select some preference items as the good ones." [39]

The approach advocated by Owen et al. [39] for evaluating boolean recommenders is congruent with our research approach. The reason it is congruent is because we have decided in this research to not conduct a user study in order to determine the relevance of items.

Evaluation Strategy for Communities of Interest

Our evaluation strategy is to compare the interestingness of content over time for all users in structural ground-truth ($T$) and $CF$-based and $LP$-based networks. These are networks of (user, item) pairs that are made up of the unique user identifiers of a root user, followees and followees-of-followees, and the unique item identifiers that are assigned (upon extraction) to the URLs embedded in the tweets of these users. Together, users and their common interests comprise a ‘community of interest.’

In our evaluation we compare interestingness of content over time in networks constructed based on different strategies, namely a baseline strategy, $CF$-based strategy and $LP$-based strategy. These strategies are similar in that they are used to determine the followees of a user. Strategies differ in that the baseline strategy collects the latest followees from a user’s manually configured profile on Twitter, while the $CF$-based and $LP$-based strategies, respectively, collect the latest automatically predicted followees of a user. As explained in Chapter 3, predictions for followees are based on the organizing principles of hierarchy and user

---

2 These unique identifiers are the identifiers assigned to users by Twitter.
3 These are unique identifiers that we assign ourselves to items (URLs) that we extract from tweets.
interests. Therefore, networks differ in how they are constructed and how well they are constructed in terms of the interestingness of content.\footnote{Since the distinguishing characteristic of networks is how they are constructed, which is something that is dictated by the strategy used, we use the same names for networks and strategies: \(T\) baseline, \(CF - \text{based}\), and \(LP - \text{based}\).}

The items in baseline \((T)\), \(CF - \text{based}\), and \(LP - \text{based}\) networks are URLs extracted from the most recent tweets found in information streams of followees and followees-of-followees that are prescribed by the different strategies. The determination of relevance in items is inferred from the relevance of information streams, which is a process that is shared among the three strategies we evaluate. Information streams are classified as relevant when they meet a user information need that is defined according to the two rules below:

- The URLs embedded in tweets that flow in information streams of users are considered relevant to the users themselves.
- The URLs embedded in tweets that flow in information streams of followees are considered relevant to the users who follow them.

We refer to the determination of relevance in this research as the \textit{structural ground truth}. In \textit{structural ground truth} the same relevance criteria apply to all users. Therefore, a relevant item for a given user is an item found in the information stream of a user or her followees during a given time. \textit{Structural ground truth} makes it possible to measure the potential precision in predicting items that would be consumed by the user if she would be organized in a given (predicted) followee network.\footnote{This is the evaluation approach we described previously.}

5.2 Results

Below we formulate our research hypotheses. We have organized these hypotheses into three groups around the main networks being compared (e.g. \(T\) vs. \(LP - \text{based}\)). For each group we will report on the results for each of the three users we selected, namely: ‘Mashable’, ‘BoingBoing’, and ‘TechCrunch.’

**Group1** (section 5.2.1)

**Hypothesis 5.1.** Precision of \(LP - \text{based}\) is higher than that of the \(T\).

\[ T_\mu < LP - \text{based}_\mu \]

**Hypothesis 5.2.** Precision of filtered \(LP - \text{based}\) is higher than that of the filtered \(T\).

\[ T^\text{filtered}_\mu < LP - \text{based}^\text{filtered}_\mu \]

**Group2** (section 5.2.2)

**Hypothesis 5.3.** Precision of \(CF - \text{based}\) is higher than that of the \(T\).

\[ T_\mu < CF - \text{based}_\mu \]
Hypothesis 5.4. Precision of filtered CF-based is higher than that of the filtered T.

\[ T_{\mu}^{filtered} < CF_{\mu}^{filtered} \]

Group3 (section 5.2.3)

Hypothesis 5.5. Precision of LP-based is higher than that of the CF-based.

\[ LP_{\mu} > CF_{\mu} \]

Hypothesis 5.6. Precision of filtered LP-based is higher than that of the filtered CF-based.

\[ LP_{\mu}^{filtered} > CF_{\mu}^{filtered} \]

5.2.1 \( T_{\mu}^{(f)} > LP_{\mu}^{(f)} \): Hypotheses 1 and 2

An independent-samples t-test was conducted to compare precision in the \( T \) and \( LP \) networks and filtered versions of these networks, for each of the users we selected. The hypotheses we respectively tested for are shown below.

1. Precision of LP-based is higher than that of the \( T \):

\[ T_{\mu} < LP_{\mu} \]

2. Precision of filtered LP-based is higher than that of the filtered \( T \):

\[ T_{\mu}^{filtered} < LP_{\mu}^{filtered} \]

Findings

For hypotheses 1 and 2 our findings differ from one user to the other (but not among all users). For users ‘Mashable’ and ‘TechCrunch’ our findings are contrary to our expectations, namely we found that \( T_{\mu} > LP_{\mu} \) and \( T_{\mu}^{filtered} > LP_{\mu}^{filtered} \). For user ‘BoingBoing’ we found the opposite, which is in accordance to our expectations, namely that \( T_{\mu} < LP_{\mu} \) and \( T_{\mu}^{filtered} < LP_{\mu}^{filtered} \). These results are summarized in Appendix D, respectively for each user, in Tables D.1 and D.2 and D.3. A more general summary of the results is provided in Tables 5.1 and 5.2 in this chapter. We will interpret these results in chapter 6.

5.2.2 \( T_{\mu}^{(f)} < CF_{\mu}^{(f)} \): Hypotheses 3 and 4

An independent-samples t-test was conducted to compare precision in the \( T \) and \( CF \) networks and filtered versions of these networks, for each of the users we selected. The hypotheses we respectively tested for are shown below.

3. Precision of CF-based is higher than that of the \( T \):

\[ T_{\mu} < CF_{\mu} \]

4. Precision of filtered CF-based is higher than that of the filtered \( T \):

\[ T_{\mu}^{filtered} < CF_{\mu}^{filtered} \]
5.2 Results Evaluating Communities of Interest

Findings

For hypotheses 3 and 4 our findings are the same from one user to the other, but differ with respect to the user ‘TechCrunch.’ For users ‘Mashable’ and ‘BoingBoing’ our findings are in accordance to our expectation, namely we found that $T_\mu < CF - based_\mu$ and $T_\mu^{filtered} < CF - based_\mu^{filtered}$. For user ‘TechCrunch’ we found the opposite, which is contrary to our expectations, namely that $T_\mu > CF - based_\mu$ and $T_\mu^{filtered} > CF - based_\mu^{filtered}$. These results are summarized in Appendix D, respectively for each user, in Tables in tables D.1 and D.2 and D.3. A more general summary of the results is provided in Tables 5.1 and 5.2 in this chapter. We will interpret these results in chapter 6.

5.2.3 $LP - based_\mu^{(f)} > CF - based_\mu^{(f)}$: Hypotheses 5 and 6

An independent-samples t-test was conducted to compare precision in the $LP - based$ and $CF - based$ networks and filtered versions of these networks, for each of the users we selected. The hypotheses we respectively tested for are shown below.

5. Precision of LP-based is higher than precision of CF-based:

$$LP - based_\mu > CF - based_\mu$$

6. Precision of filtered LP-based is higher than precision of filtered CF-based:

$$LP - based_\mu^{filtered} > CF - based_\mu^{filtered}$$

Findings

For hypotheses 5 and 6 our findings are the same from one user to the other (for all three users), but are contrary to our expectations. For all three users our findings are contrary to our expectations, namely we found that $LP - based_\mu < CF - based_\mu$ and $LP - based_\mu^{filtered} < CF - based_\mu^{filtered}$. These results are summarized in Appendix D, respectively for each user, in Tables D.1 and D.2 and D.3. A more general summary of the results is provided in Tables 5.1 and 5.2 in this chapter. We will interpret these results in chapter 6.
Summary of Findings for unfiltered networks

<table>
<thead>
<tr>
<th>Mashable</th>
<th>TechCrunch</th>
<th>BoingBoing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $T_\mu &gt; LP - \text{based}_\mu$</td>
<td>$T_\mu &gt; LP - \text{based}_\mu$</td>
<td>$T_\mu &lt; LP - \text{based}_\mu$</td>
</tr>
<tr>
<td>3. $T_\mu &lt; CF - \text{based}_\mu$</td>
<td>$T_\mu = CF - \text{based}_\mu$</td>
<td>$T_\mu &lt; CF - \text{based}_\mu$</td>
</tr>
<tr>
<td>5. $LP - \text{based}<em>\mu &lt; CF - \text{based}</em>\mu$</td>
<td>$LP - \text{based}<em>\mu &lt; CF - \text{based}</em>\mu$</td>
<td>$LP - \text{based}<em>\mu &lt; CF - \text{based}</em>\mu$</td>
</tr>
<tr>
<td>$LP - \text{based}<em>\mu &lt; T</em>\mu &lt; CF - \text{based}_\mu$</td>
<td>$LP - \text{based}<em>\mu &lt; T</em>\mu = CF - \text{based}_\mu$</td>
<td>$T_\mu &lt; LP - \text{based}<em>\mu &lt; CF - \text{based}</em>\mu$</td>
</tr>
</tbody>
</table>

Table 5.1: Summary of Findings for Hypotheses 1, 3 and 5
### Summary of Findings for Hypotheses 2, 4 and 6

<table>
<thead>
<tr>
<th>Mashable</th>
<th>TechCrunch</th>
<th>BoingBoing</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. ( T_{\mu}^{(f)} &gt; LP - based_{\mu}^{(f)} )</td>
<td>( T_{\mu}^{(f)} &gt; LP - based_{\mu}^{(f)} )</td>
<td>( T_{\mu}^{(f)} &lt; LP - based_{\mu}^{(f)} )</td>
</tr>
<tr>
<td>4. ( T_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( T_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( T_{\mu}^{(f)} = CF - based_{\mu}^{(f)} )</td>
</tr>
<tr>
<td>6. ( LP - based_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( LP - based_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( LP - based_{\mu}^{(f)} = CF - based_{\mu}^{(f)} )</td>
</tr>
<tr>
<td>( LP - based_{\mu}^{(f)} &lt; T_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( LP - based_{\mu}^{(f)} &lt; T_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} )</td>
<td>( T_{\mu}^{(f)} &lt; CF - based_{\mu}^{(f)} &lt; LP - based_{\mu}^{(f)} )</td>
</tr>
</tbody>
</table>

Table 5.2: Summary of Findings for Hypotheses 2, 4 and 6
5.3 Discussion

In this section we discuss the results of this research in relation to possible alternative explanations, limitations in our research approach and initial ideas for further research.\footnote{We will discuss further research extensively in the next chapter, here we will just introduce some ideas related to the discussion of results.} Our intention here is to explain our results and to reflect on them by considering the possibility of alternative explanations, either in breadth because of limitations or in depth because of aspects we did not consider.

5.3.1 Explanation of Results

In order to explain our results we refer again to our evaluation method. Our method is based on the following approach:

Structure a network such that we can measure the precision in predicting items that would be consumed by the user if she would be organized within the different networks of (predicted) followees, namely the $T$, $LP$–based and $CF$ – based networks.

Given that we have a structural ground truth and a semantical ground truth that are automatically inferred from the community of interest of a user, the precision value we measure for each network describes the potential performance (or potential precision), which we call ‘interestingness’:

The (potential) precision in predicting items that would be consumed by the user if she would be organized within a given (predicted) followee network.

The explanation of our results for the two kinds of networks constructed in this research are summarized below:

Structural networks

Structural networks are the subject of hypotheses 1, 3 and 5. These are networks that are unfiltered, which means that all items published by (predicted) followee users are considered relevant for a given user. Our results indicate that the $CF$–based networks outperform the $LP$–based and $T$ networks.

Semantic networks

Semantic networks are the subject of hypotheses 2, 4 and 6. These are networks that are filtered, which means that from the items published by (predicted) followee users only those that are found to match semantic entity-based profiles are considered relevant for a given user. Our results indicate that filtered $CF$–based networks outperform filtered $LP$–based and $T$ networks.

5.3.2 Alternative Explanations

The results of this research support the belief that communities of interest on Twitter are more effectively organized around user interests. This is based on our findings which support the $CF$ – based construction of networks. However, could this same result also be found for other users? Or would different results be found because
user interests in other communities are organized differently? For instance, it could be the case that user interests in other communities are organized hierarchically.

Our answer to the previous questions is related to the limitations of this research and the type of conclusions we can make based on the evidence found. This answer is provided in sub-section 5.3.3. We also have a more practical answer that is related to the limitations of the hierarchical missing link prediction algorithm of Clauset et al. [17], which we discuss in Section 5.3.4. And we have an answer based on our experience in analyzing the data collected for this research, which we discuss on the Section 5.3.5.

5.3.3 Limitations of this Research

The main limitation of this research comes from our data collection strategy. We chose to only collect data from the perspective of three different users. This means we did not collect information from the perspective of all the other users found in the communities of interest that we constructed. Consequently, we can not make conclusions at the level of a community. Our conclusions are therefore only indications of what could happen at the level of a community.

Another limitation of this research is related to the URLs that users post on tweets. These URLs are often shortened by the users themselves and in all cases the URLs are shortened by Twitter as URLs under the t.co domain. Even though we employed three different URL expansion services, we found that URLs are shortened at different levels. Some users shorten URLs without including paths, parameters and fragments. But other users shorten only part of or the entire URL. For example, there is a difference between expanded URLs for a webpage that includes fragments (which refer to an internal section of a web document) from one that does not. For us this difference implies that we completely miss an implicit connection between users when URLs are expanded to different versions of a baseline URL.

Despite the limitations brought forth by shortened URLs, we see possible solutions. One solution is to pre-process URLs so that we can bring them to a baseline. For example, if we consider again a URL with parameters, we would pre-process expanded URLs to strip away any parameters. Another solution is to pre-process URLs to determine which URLs refer to the same resource. More complex solutions, like following and counting links between shortened URLs could also be employed as possible solutions should they provide some clear advantage.

5.3.4 Generalizability of Results

The results of this research can be generalized across users similar to those we analyzed here. For example, based on our findings we would be inclined to believe that CF-based networks would offer more relevant content for technology microbloggers or users maintaining microblogs on particular subjects. It is yet to be determined if our results can be generalized to normal users. However, since our methods are adaptive to user interests we argue that as long as users express their interests consistently over time, then we can improve their user experience. Furthermore, we believe that technology microbloggers are leaders in their respective communities of interest and thus we argue that what applies to them also applies to normal users who follow them. Next steps in our research, aimed at improving the generalizability of our results, involve testing the threshold of influence of followees on users and testing the threshold of effectiveness of CF-based followee recommendations and automated network construction.

Despite generalizability across similar users, our results cannot be generalized to entire communities of interest. As explained previously, this would require re-

7By normal users, we mean users who do not consistently express the same interests.
peating the same analyses for each member of a community of interest, a task that was considered overwhelming for this project given time and resource constraints. However, this might become more feasible with Twitter’s streaming API and access to dedicated resources (servers) for the collection of data.\(^8\) Primarily, such resources would be focused on constructing one type of network so that they could maintain a dedicated connection to Twitter’s streaming API servers. Parallel resources would then be dedicated to constructing different networks over time.

5.3.5 Generalizability of Methods

The methods in this research can be applied to any user on Twitter. Applying these methods at a larger scale, for instance at the level of communities, is also possible as we have explained previously. Furthermore, our research methods can be applied across different time periods. This enables the researcher to strive for generalizability along various dimensions: user, community and time.

In addition, the methods in this research can also be extended to study triadic relations (user-object-user) on objects that are not URLs. For instance, as a direction of further research, discussed in more detail in Chapter 6 (Section 6.4), we propose to perform the same analyses on semantic entities. We believe this is a way to add semantics to social network analysis. Furthermore, by applying the methods in this research periodically in conjunction with other methods it is possible to learn the relevance of objects (URLs, semantic entities) over time. We believe such extended analyses will improve personalization of information streams on Twitter. We provide more motivation for these points in more detail in Chapter 6.

5.3.6 Further Research

We foresee the possibility of further research linked to our results through user studies and monitoring of interaction in communities of interest. To gain further proof of our results and analyze how real users would actually behave if they could benefit from being organized in the CF-based community of interest that is generated by our methods, it would be possible to deploy the proposed algorithms and conduct a user study. Jointly, or independently of the user study, further research could focus on monitoring how users will be interacting in the automatically generated communities of interests. We will discuss these and other directions for further research in more detail in Chapter 6.

---

\(^8\) Twitter developers Twitter Streaming API: [https://dev.twitter.com/docs/streaming-api](https://dev.twitter.com/docs/streaming-api) (Retrieved October 2011)
Chapter 6

Conclusions

‘Birds of a feather flock together.’

Popular proverb

In this chapter we draw conclusions from our research on link prediction and the evolution of communities on Twitter. We conducted an extensive research project in which we employed information retrieval and extraction, data mining, statistical analyses and link prediction in networks. These methods and techniques allowed us to extensively analyze communities consisting of around 1,000 users: a root user, her followees and the followees-of-followees. These users were situated in conventional and alternative communities of interest. The source of such communities were the root users ‘Mashable’, ‘BoingBoing’ and ‘TechCrunch.’

6.1 Research Model

Our research model is based on the construction of different communities of interest on Twitter. These communities are groups of users that share the same interests and come together in different ways. In conventional communities users come together based on their followings, which are defined using the conventional methods provided by Twitter. As users manually add followees to their Twitter profile they build their own community of interest. In alternative communities, we developed communities based on organizational principles inferred through connections brought about by the URLs embedded in the tweets of users. These communities also consist of followees and followees-of-followees, but differ because of the way in which the communities are designed and constructed. We employed two main algorithms for this purpose, namely the hierarchical link prediction algorithm of Clauset et al. [17] and an algorithm implementing the link prediction approach to collaborative filtering [22].

Hierarchy and user interests stand in this research as alternative organizational principles used to automatically build communities of interest on Twitter. We identified the importance of principles in community building on Twitter by studying the mechanics of spreading tweets in a social network of users who share similar interests, a process which has been described as follows:

“As a social network, Twitter revolves around ‘the principle of followers’. When you choose to follow another Twitter user, that user’s tweets appear in reverse chronological order on your main Twitter page. If you follow 20 people, you’ll see a mix of tweets scrolling down the page:
6.2 Answering the research questions

Thus, our motivation for this research was to compare the performance of network construction algorithms in terms of the interestingness of content shared in alternative communities of interest which are constructed on the basis of different organizational principles. Namely, these principles are: user intuitions, hierarchy and user interests. In order to test the performance of these principles we constructed networks of \((\text{user}, \text{item})\) pairs which were subsequently used to measure interestingness of content. The items in these pairs are URLs embedded in tweets. We claim that users interact through these URLs by expressing their interest in the topics and entities related to such URLs. Based on interactions in URLs embedded in tweets we crawled the Twitter social network to construct conventional and alternative communities of interest. We refer to the snapshots of these communities, respectively, as conventional and alternative followee networks.

6.1.1 Conventional Followee Network

To build the conventional followee network \((T)\) we extracted embedded URLs from the three most recent tweets of the root user, her 30 latest followees and the 30 latest followees-of-followees.

6.1.2 Alternative Followee Network

To build the alternative followee networks \((LP - \text{based} \text{ and } CF - \text{based})\) we first built an intermediate ‘implicit followee network’ \((IFN)\). The construction an \(IFN\) was similar to that of the \(T\) network. We extracted embedded URLs from the three most recent tweets of the root user, her 30 latest (implicit) followees and the (30 latest) followees-of-(implicit) followees. This pre-processing step allowed us to come up with a \((\text{user}, \text{user})\) network that is needed in order to apply the link prediction approach to collaborative filtering ([22] and which is also needed as input for the hierarchical link prediction algorithm of Clauset et al. [17].

LP-based and CF-based Followee Networks

Once we had a network of \((\text{user}, \text{user})\) pairs we ran the hierarchical missing link prediction algorithm of Clauset et al. [17] and a recommender algorithm implementing the link prediction approach to collaborative filtering.\(^2\) The result was a set of predicted followees which we used to construct the \(LP - \text{based} \text{ and } CF - \text{based}\) followee networks in the same way in which we had constructed the conventional followee network \((T)\). Therefore, we extracted embedded URLs from the three most recent tweets of the root user, her 30 latest (predicted or recommended) followees and the (30 latest) followees-of-(predicted or recommended) followees.

6.2 Answering the research questions

We are now ready to answer the research questions that were posed in chapter 1.

---

\(^1\)How Twitter Will Change the Way We Live – Time
http://www.time.com/time/printout/0,8816,1902604,00.html (Retrieved Aug 2011)

\(^2\)Implemented in Apache Mahout using the Log likelihood similarity algorithm and nearest-n user neighborhood algorithm.
6.2.1 Answer to research question 1

1. *Is the CF-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?*

Yes, the CF-based approach consistently outperforms (T-based) conventional communities of interest and LP-based alternative communities of interest in terms of interestingness of content consumed in such communities of interest over time.

1.1. *Is the filtered CF-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?*

Yes, the filtered CF-based approach consistently outperforms (T-based) conventional communities of interest and LP-based alternative communities of interest in terms of interestingness of content consumed in such communities of interest over time.

6.2.2 Answer to research question 2

1. *Is the LP-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?*

No, the LP-based approach consistently underperforms when compared to (T-based) conventional communities of interest and CF-based alternative communities of interest in terms of interestingness of content consumed in such communities of interest over time.

2.1. *Is the filtered LP-based approach more effective in terms of the interestingness of content consumed in communities of interest over time?*

No, the filtered LP-based approach consistently underperforms when compared to (T-based) conventional communities of interest and CF-based alternative communities of interest in terms of interestingness of content consumed in such communities of interest over time.

6.3 Implications and Reflection

This section focuses on two related aspects. First, we discuss the implications of this research in relation to its results and how they can be understood and used. Second, we include reflections on what our research means for users, researchers and organizations interested in Twitter.

6.3.1 Implications

The results of this research support the belief that there is room for improvement in the Twitter social network and micro-blogging service. Our results show that the organizational principle of user interests performs better than current ways of finding interesting followees for technology bloggers. This implies that micro-bloggers who focus on particular topics will also benefit from recommendations for interesting followees. Moreover, we believe our approach is better than the conventional approach of manually maintaining lists of followees for two reasons:
Further Research

(1) our approach is adaptive and (2) it can be automated. This implies that it can be extended and applied to all users on Twitter.

In comparison to the current method of manually building communities of followers, our method performs better and has clear advantages. This implies that Twitter would benefit from implementing our methods and techniques as a feature for automated list maintenance in their social network and micro-blogging service. Alternatively, it would be possible to build a shadow Twitter social network and micro-blogging service implementing our methods and techniques for maintenance of followers. We believe such a shadow network would stand a fair chance of overtaking Twitter in popularity because of two reasons: (1) ease of follower maintenance and (2) because of the feasibility to control information overload based on an understanding of user needs (which can in part be provided by the users themselves through parameters).

6.3.2 Reflection

Twitter has been used to organize protests, revolutions, communications among groups of people, interactive TV viewing, marketing campaigns, among many other social situations where interaction between people can be shaped for momentous accomplishments. However, despite the importance of Twitter as a medium for organization, the role of the service as a means to negative ends, such as organizing riots and other criminal pursuits, is also tangible. Drawing the line between good and bad is not an easy task. Neither is it straightforward, or fair for that matter, to claim that events have been organized on Twitter without having an understanding of what it means to organize a social undertaking on Twitter and the extent to which the social network actually plays a role in such organization and mobilization of people to a specific common goal.

In spite of the lack of evidence people are quick to point out events that have been ‘organized on Twitter.’ But what exactly does it mean for some event to be organized on Twitter? For instance, what does it take to organize a revolution on Twitter? To what extent is Twitter the medium that people use to organize themselves? And if Twitter indeed plays a relevant role, why do more conventional organizations, like companies or universities, not use it instead of their explicit organizational structures? Or, on the other hand, is it the case that there is an implicit and informal structure that is organically evolving and for which Twitter provides a means of communication? If so, how does such organization work? Can we make it explicit in order to understand it? Does it follow any principles? Can we predict its features or its outcomes? In this research we explored the latter questions and offered approaches to find answers.

6.4 Further Research

We have divided our suggestions for further research in three main directions. We discuss each direction in the next sections.

6.4.1 Online social networks

Our approach of designing and engineering communities of interest based on implicit connections between users can be applied in the study of other social networks. On social networks like Facebook and Google+ it would be interesting to research whether communities organized around user interests would be more vibrant and cohesive than communities organized based on the other two approaches studied in this research, namely hierarchy and user intuitions. Below we describe two suggested research projects.
Understanding music communities on Facebook/Spotify. Facebook users can now interact with objects through apps that are integrated to the Open Graph. The results of user actions can be published on different social channels on the Facebook platform. It would be interesting to gather data on actions and/or objects and use it to construct communities of interest. For example, using the Spotify app users can ‘listen’ to music and share song titles their listening to. Communities of interest could be constructed around music genres, artists or songs. Then it would be possible to measure interestingness as proposed in this research and test whether communities organized using the $CF - based$ approach perform better than communities organized using the $LP - based$ or $T - based$ approaches. As well, it would be possible to measure other organizational phenomena and perform regression analysis to explain different aspects of communities.

Understanding search communities. On Google+ it is now possible to get access to real-time search results. It would be interesting to gather data on search queries and use it to construct communities of interest. For example, when a user types in an event as a search query she would become part of a community of users who typed the same or similar search queries. Communities of interest could be constructed around events, news, products, or anything else that can be referenced using a search query. Then it would be possible to measure interestingness of content as proposed in this research and test whether communities organized using the $CF - based$ approach perform better than communities organized using the $LP - based$ or $T - based$ approaches.

As well, it would be possible to study the influence of organizational principles in Google+ communities of interest on the outputs of social media production. In other words, if we think of search queries in Google+ and if we count the number of clicks on search results (click-through rate), then we could measure the relationship between communities of interest and popular content shared on Google+. Doing something similar on a search engine, e.g. the Google search engine, we could measure the success of an online advertising campaign by counting the number of clicks on advertisements associated to search queries coming from users in a community of interest.

Web service for construction of communities of interest. Build a web service that constructs communities of interest over time and which implements results as lists in the Twitter social network and micro-blogging service. In terms of research, the idea would be to make the service public and study usage of engineered lists of information streams in comparison to conventional Twitter information streams, for all users in a community of interest.

Shadow Twitter network. Build a shadow social network for Twitter which constructs communities of interest for any Twitter user. The new recommendation features in the WebDSL model-driven language could be used for implementation. Features could include automated real-time construction of communities of interest based on: (1) search query terms, (2) actions and objects, and (3) semantic entities, hashtags or URLs. In terms of research, the shadow network could be used to collect usage data for later analysis of communities of interest.

---

3Facebok - Open Graph: https://developers.facebook.com/docs/beta/opengraph/ (Retrieved October 2011)

4Google+: Real-time search and improved hashtag support: https://plus.google.com/107117483540235115863/posts/dXovwc1hSyY (Retrieved October 2011)

6.4 Further Research

6.4.2 Semantic social networks

The following directions for further research focus on leveraging semantic enrichment to generate semantic social networks, learning relevance of semantic entities in such networks over time, and personalizing information streams. These directions for further research build upon each other and could be considered as part of one larger research project.

How can semantics be added to social network analysis? Adding semantics to social network analysis means that we use semantic entities to establish (otherwise implicit) social relations between users. This process results in a social network that summarizes semantic connections between users. For example in Twitter we can create an alternate social graph by establishing relations between users based on the co-occurrence of semantic entities extracted using the TUMS web service [48] from semantically enriched tweets [8] over time. This social graph is different than the conventional social graph created manually by users through their followings in that relations in the alternate social graph can be implicitly, dynamically and automatically determined through the semantic content of tweets. We search for such content on Twitter and use the resulting tweets to identify users who have posted related content. As this content changes so do the connections between users. This translation of semantic interaction into a social graph is our key modeling step for networks of followers and followees on Twitter.

How can machine-learning methods be employed to learn the relevance of semantic entities over time? Once we can build semantic social graphs, the next step is to construct a perspective of evolution. A perspective of evolution describes how a graph changes over time from the point of view of a given user. Having such perspective implies we have data that can be used as training examples for machine learning of the relevance of semantic entities over time. Therefore, machine learning methods are employed to learn the relevance of semantic entities over time from the perspective of a given user.

How can information streams be personalized using semantic entities? We argue that in the process of summarizing semantic connections between users we do not incur in loss of information because the summarized connections are only indications of the interests of a user at a certain moment in time. The personalization of information streams comes from recommending users to follow based on connections summarized in a semantic social network (as discussed above).

How can information streams be personalized using machine-learned relevance of semantic entities? Having access to machine learned relevance of semantic entities means that we have more detailed information about the interests of a user based on the weights learned from training examples. From there on the process of personalization of information streams comes, similar to above, from recommending users to follow based on connections summarized in weighted semantic social networks.

6.4.3 Confirmatory research

The next two directions for further research focus on validation and confirmation of the results of this research.

Regression analysis using measurements from other organizational phenomena. Measure other organizational phenomena on communities of interest
and conduct regression analysis to see whether the measure of interestingness can be used to make predictions about other organizational phenomena. For example, measure the clustering coefficient on networks of interest and do regression analysis on clustering coefficient using measurements on interestingness. A result in which interestingness would be found to be closely correlated to clustering coefficient would be a confirmation of the results in this research because it would indicate the level of interestingness of content shared in communities depending on their clustering coefficient.

**Randomized K-fold cross-validation.** Introduce randomness into the selection of training and test data as suggested by Owen et al. [39] (pg. 22). Perform (k-times) repeated measurements and average the results, then compare against the results of this research. A result in which interestingness were found to be higher in $CF$-based networks than in $LP$-based or $T$-based networks would confirm and validate the results of this research because it would show that content that is more interesting is shared in communities of interest constructed using the $CF$-based approach.

**User study.** We foresee the possibility of further research linked to our results through user studies and monitoring of interaction in communities of interest. To gain further proof of our results and analyze how real users would actually behave if they could benefit from being organized in the $CF$-based community of interest that is generated by our methods, it would be possible to deploy the proposed algorithms and conduct a user study. Jointly, or independently of the user study, further research could focus on monitoring how users will be interacting in the automatically generated communities of interests.
Appendix A

A.1 Link Prediction

![Diagram showing relative performance ratio versus common neighbors predictor]

Figure A.1: Link prediction vs. common neighbors predictor [38]
Relative performance ratio versus graph distance predictor

Figure A.2: Link prediction vs. graph distance predictor [38]
Appendix A

A.1 Link Prediction

Figure A.3: Link prediction vs. random predictor [38]
A.2 Hierarchical Link Prediction

Figure A.4: Link prediction vs. common neighbors predictor [17]

Figure A.5: Link prediction vs. graph distance predictor [17]
Appendix A

A.2 Hierarchical Link Prediction

Figure A.6: Link prediction vs. random predictor [17]
Appendix B

B.1 Tables for user ‘Mashable’
B.1 Tables for user ‘Mashable’

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std Err</th>
<th>Std Dev</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>56</td>
<td>.1358144</td>
<td>.0039227</td>
<td>.0293552</td>
<td>.1279531 .1436758</td>
</tr>
<tr>
<td>PFN</td>
<td>55</td>
<td>.1013269</td>
<td>.0201715</td>
<td>.1495961</td>
<td>.0608854 .1417684</td>
</tr>
<tr>
<td>combined</td>
<td>111</td>
<td>.118726</td>
<td>.0102741</td>
<td>.1082448</td>
<td>.0983651 .139087</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>.0344875</td>
<td>.0203772</td>
<td>.0748743</td>
<td></td>
</tr>
</tbody>
</table>

Degrees of freedom: 109

\[ Ho : mean(0) - mean(1) = diff = 0 \]

\[ Ha : diff < 0 \quad Ha : diff \neq 0 \quad Ha : diff > 0 \]
\[ t = 1.6925 \quad t = 1.6925 \quad t = 1.6925 \]
\[ P < t = 0.9533 \quad P > t = 0.0934 \quad P > t = 0.0467 \]

Table B.1: Two-sample t test with equal variances: CFN vs. PFN (user: ‘Mashable’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std Err</th>
<th>Std Dev</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>56</td>
<td>.2260434</td>
<td>.0082114</td>
<td>.0614484</td>
<td>.2095874 .2424994</td>
</tr>
<tr>
<td>filtered PFN</td>
<td>55</td>
<td>.1673389</td>
<td>.0275532</td>
<td>.2043398</td>
<td>.1120981 .225797</td>
</tr>
<tr>
<td>combined</td>
<td>111</td>
<td>.1969556</td>
<td>.0144743</td>
<td>.1524961</td>
<td>.1682709 .2256402</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>.0587045</td>
<td>.0285335</td>
<td>.002152</td>
<td>.1152569</td>
</tr>
</tbody>
</table>

Degrees of freedom: 109

\[ Ho : mean(0) - mean(1) = diff = 0 \]

\[ Ha : diff < 0 \quad Ha : diff \neq 0 \quad Ha : diff > 0 \]
\[ t = 2.0574 \quad t = 2.0574 \quad t = 2.0574 \]
\[ P < t = 0.9790 \quad P > t = 0.0420 \quad P > t = 0.0210 \]

Table B.2: Two-sample t test with equal variances: filtered CFN vs. filtered PFN (user: ‘Mashable’)
### Appendix B  
#### B.1 Tables for user ‘Mashable’

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>56</td>
<td>.1358144</td>
<td>.0039227</td>
<td>.0293552</td>
<td>.1279531 .1436758</td>
</tr>
<tr>
<td>RFN</td>
<td>53</td>
<td>.1501967</td>
<td>.0041936</td>
<td>.0305297</td>
<td>.1417817 .1586117</td>
</tr>
<tr>
<td>combined</td>
<td>109</td>
<td>.1428076</td>
<td>.0029363</td>
<td>.0306556</td>
<td>.1369874 .1486278</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.0143823</td>
<td>.005736</td>
<td></td>
<td>-.0257533 -.0030112</td>
</tr>
</tbody>
</table>

Degrees of freedom: 107

<table>
<thead>
<tr>
<th>Ho</th>
<th>mean(0) − mean(1) = diff = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ha : diff &lt; 0</td>
<td>Ha : diff ! = 0</td>
</tr>
<tr>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>-.2.5073</td>
<td>-.2.5073</td>
</tr>
<tr>
<td>P &lt; t</td>
<td>P &gt; t</td>
</tr>
<tr>
<td>.0068</td>
<td>.0137</td>
</tr>
</tbody>
</table>

Table B.3: Two-sample t test with equal variances: CFN vs. RFN (user: ‘Mashable’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>56</td>
<td>.2260434</td>
<td>.0082114</td>
<td>.0614484</td>
<td>.2095874 .2424994</td>
</tr>
<tr>
<td>filtered RFN</td>
<td>53</td>
<td>.2681347</td>
<td>.0124446</td>
<td>.0905981</td>
<td>.2431628 .2931066</td>
</tr>
<tr>
<td>combined</td>
<td>109</td>
<td>.2465098</td>
<td>.0076155</td>
<td>.0795086</td>
<td>.2314145 .2616051</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.0420913</td>
<td>.0149096</td>
<td></td>
<td>-.071708 -.0124747</td>
</tr>
</tbody>
</table>

Degrees of freedom: 107

<table>
<thead>
<tr>
<th>Ho</th>
<th>mean(0) − mean(1) = diff = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ha : diff &lt; 0</td>
<td>Ha : diff ! = 0</td>
</tr>
<tr>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>-.2.8523</td>
<td>-.2.8523</td>
</tr>
<tr>
<td>P &lt; t</td>
<td>P &gt; t</td>
</tr>
<tr>
<td>.0026</td>
<td>.0052</td>
</tr>
</tbody>
</table>

Table B.4: Two-sample t test with equal variances: filtered CFN vs. filtered RFN (user: ‘Mashable’)
### B.1 Tables for user ‘Mashable’

#### Appendix B

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>95% Conf. Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PFN</strong></td>
<td>55</td>
<td>0.1013269</td>
<td>0.0201715</td>
<td>0.1495961</td>
<td>0.0608854 - 0.1417684</td>
</tr>
<tr>
<td><strong>RFN</strong></td>
<td>54</td>
<td>0.1474153</td>
<td>0.004967</td>
<td>0.0364998</td>
<td>0.1374527 - 0.1573778</td>
</tr>
<tr>
<td><strong>combined</strong></td>
<td>109</td>
<td>0.1241597</td>
<td>0.0106569</td>
<td>0.1112618</td>
<td>0.1030358 - 0.1452836</td>
</tr>
<tr>
<td><strong>diff</strong></td>
<td></td>
<td>-0.0460883</td>
<td>0.0209455</td>
<td></td>
<td>-0.0876104 - 0.0045663</td>
</tr>
</tbody>
</table>

Degrees of freedom: 107

**Ho**: mean(0) − mean(1) = diff = 0

**Ha**: diff < 0  
**Ha**: diff = 0  
**Ha**: diff > 0  

$$t = -2.2004$$  
$$t = -2.2004$$  
$$t = -2.2004$$  

$$P < t = 0.0150$$  
$$P > t = 0.0299$$  
$$P > t = 0.9850$$  

Table B.5: Two-sample t test with equal variances: **PFN** vs. **RFN** (user: ‘Mashable’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>95% Conf. Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>filtered PFN</strong></td>
<td>55</td>
<td>0.1673389</td>
<td>0.0275532</td>
<td>0.2043398</td>
<td>0.1120981 - 0.2225797</td>
</tr>
<tr>
<td><strong>filtered RFN</strong></td>
<td>54</td>
<td>0.2631692</td>
<td>0.0131829</td>
<td>0.096874</td>
<td>0.2367277 - 0.2896107</td>
</tr>
<tr>
<td><strong>combined</strong></td>
<td>109</td>
<td>0.2148145</td>
<td>0.0159701</td>
<td>0.1667324</td>
<td>0.183159 - 0.2464699</td>
</tr>
<tr>
<td><strong>diff</strong></td>
<td></td>
<td>-0.0958304</td>
<td>0.030724</td>
<td></td>
<td>-0.1567372 - 0.0349235</td>
</tr>
</tbody>
</table>

Degrees of freedom: 107

**Ho**: mean(0) − mean(1) = diff = 0

**Ha**: diff < 0  
**Ha**: diff = 0  
**Ha**: diff > 0  

$$t = -3.1191$$  
$$t = -3.1191$$  
$$t = -3.1191$$  

$$P < t = 0.0012$$  
$$P > t = 0.0023$$  
$$P > t = 0.9988$$  

Table B.6: Two-sample t test with equal variances: **filtered PFN** vs. **filtered RFN** (user: ‘Mashable’)
B.2 Tables for user ‘BoingBoing’
### Table B.7: Two-sample t test with equal variances: CFN vs. PFN (user: 'BoingBoing')

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>43</td>
<td>0.0541214</td>
<td>0.0031269</td>
<td>0.0205047</td>
<td>0.047811 .0604319</td>
</tr>
<tr>
<td>PFN</td>
<td>42</td>
<td>0.1330803</td>
<td>0.0131615</td>
<td>0.0852966</td>
<td>0.1065 .1596606</td>
</tr>
<tr>
<td>combined</td>
<td>85</td>
<td>0.0931364</td>
<td>0.0079249</td>
<td>0.0730638</td>
<td>0.0773769 .1088959</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.0789588</td>
<td>0.0133852</td>
<td>0.105814</td>
<td>-.0523363 .0523363</td>
</tr>
</tbody>
</table>

Degrees of freedom: 83

\[ Ho : \text{mean}(0) - \text{mean}(1) = \text{diff} = 0 \]

\[ Ha : \text{diff}<0 \quad Ha : \text{diff}>0 \]
\[ t = -5.8990 \quad t = -5.8990 \]
\[ P < t = 0.0000 \quad P > t = 1.0000 \]

Table B.8: Two-sample t test with equal variances: filtered CFN vs. filtered PFN (user: 'BoingBoing')

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>43</td>
<td>.137097</td>
<td>.0170461</td>
<td>.1117784</td>
<td>.1026967 .1714974</td>
</tr>
<tr>
<td>filtered PFN</td>
<td>42</td>
<td>.274337</td>
<td>.050274</td>
<td>.325813</td>
<td>.1728066 .3758675</td>
</tr>
<tr>
<td>combined</td>
<td>85</td>
<td>.2049097</td>
<td>.0271866</td>
<td>.2506484</td>
<td>.1508461 .2589733</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.13724</td>
<td>.0525886</td>
<td>.2418366</td>
<td>-.0326434 .0326434</td>
</tr>
</tbody>
</table>

Degrees of freedom: 83

\[ Ho : \text{mean}(0) - \text{mean}(1) = \text{diff} = 0 \]

\[ Ha : \text{diff}<0 \quad Ha : \text{diff}>0 \]
\[ t = -2.6097 \quad t = -2.6097 \]
\[ P < t = 0.0054 \quad P > t = 0.9946 \]
### Table B.9: Two-sample t test with equal variances: CFN vs. RFN (user: ‘BoingBoing’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>43</td>
<td>.0541214</td>
<td>.0031269</td>
<td>.0205047</td>
<td>[.047811, .0604319]</td>
</tr>
<tr>
<td>RFN</td>
<td>40</td>
<td>.1684375</td>
<td>.0104676</td>
<td>.066203</td>
<td>[.1472648, .1896103]</td>
</tr>
<tr>
<td>combined</td>
<td>83</td>
<td>.1092135</td>
<td>.0082158</td>
<td>.0748494</td>
<td>[.0928648, .1255574]</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.1143161</td>
<td>.0105996</td>
<td>-.135406</td>
<td>[.0932261, .1255574]</td>
</tr>
</tbody>
</table>

Degrees of freedom: 81

\[
\begin{align*}
Ho &: mean(0) - mean(1) = diff = 0 \\
Ha &: diff < 0 \\
t &= -10.7849 \\
P < t &= 0.0000
\end{align*}
\]

Table B.10: Two-sample t test with equal variances: filtered CFN vs. filtered RFN (user: ‘BoingBoing’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>43</td>
<td>.137097</td>
<td>.0170461</td>
<td>.1117784</td>
<td>[.1026967, .1714974]</td>
</tr>
<tr>
<td>filtered RFN</td>
<td>40</td>
<td>.2689958</td>
<td>.0414509</td>
<td>.2621584</td>
<td>[.1851534, .3528381]</td>
</tr>
<tr>
<td>combined</td>
<td>83</td>
<td>.2006627</td>
<td>.0228888</td>
<td>.2085268</td>
<td>[.1551296, .2461958]</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.1318987</td>
<td>.0436973</td>
<td>-.2188426</td>
<td>-.0449548</td>
</tr>
</tbody>
</table>

Degrees of freedom: 81

\[
\begin{align*}
Ho &: mean(0) - mean(1) = diff = 0 \\
Ha &: diff < 0 \\
t &= -3.0185 \\
P < t &= 0.0017
\end{align*}
\]

Table B.10: Two-sample t test with equal variances: filtered CFN vs. filtered RFN (user: ‘BoingBoing’)

85
### Table B.11: Two-sample t test with equal variances: PFN vs. RFN (user: ‘BoingBoing’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFN</td>
<td>42</td>
<td>.1330803</td>
<td>.0131615</td>
<td>.0852966</td>
<td>.1065 .1596606</td>
</tr>
<tr>
<td>RFN</td>
<td>40</td>
<td>.1684375</td>
<td>.0104676</td>
<td>.066203</td>
<td>.1472648 .1896103</td>
</tr>
<tr>
<td>combined</td>
<td>82</td>
<td>.1503277</td>
<td>.0086314</td>
<td>.0781609</td>
<td>.1331539 .1675016</td>
</tr>
<tr>
<td>diff</td>
<td>-</td>
<td>-.0353573</td>
<td>.0169199</td>
<td>-.069029</td>
<td>-.0016855</td>
</tr>
</tbody>
</table>

Degrees of freedom: 80

**Ho**: mean(0) − mean(1) = diff = 0

**Ha**: diff < 0  
**Ha**: diff = 0  
**Ha**: diff > 0

\[ t = -2.0897 \]  
\[ t = -2.0897 \]  
\[ t = -2.0897 \]

\[ P < t = 0.0199 \]  
\[ P > t = 0.0398 \]  
\[ P > t = 0.9801 \]

---

### Table B.12: Two-sample t test with equal variances: filtered PFN vs. filtered RFN (user: ‘BoingBoing’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered PFN</td>
<td>42</td>
<td>.274337</td>
<td>.050274</td>
<td>.325813</td>
<td>.1728066 .3758675</td>
</tr>
<tr>
<td>filtered RFN</td>
<td>40</td>
<td>.2689958</td>
<td>.0414509</td>
<td>.2621584</td>
<td>.1851534 .3528381</td>
</tr>
<tr>
<td>combined</td>
<td>82</td>
<td>.2717315</td>
<td>.0325408</td>
<td>.2946699</td>
<td>.2069855 .3364776</td>
</tr>
<tr>
<td>diff</td>
<td>-</td>
<td>.0053413</td>
<td>.065504</td>
<td>-1.250157</td>
<td>.1356983</td>
</tr>
</tbody>
</table>

Degrees of freedom: 80

**Ho**: mean(0) − mean(1) = diff = 0

**Ha**: diff < 0  
**Ha**: diff = 0  
**Ha**: diff > 0

\[ t = 0.0815 \]  
\[ t = 0.0815 \]  
\[ t = 0.0815 \]

\[ P < t = 0.5324 \]  
\[ P > t = 0.9352 \]  
\[ P > t = 0.4676 \]

Table B.12: Two-sample t test with equal variances: filtered PFN vs. filtered RFN (user: ‘BoingBoing’)
B.3 Tables for user ‘TechCrunch’
### Table B.13: Two-sample t test with equal variances: CFN vs. PFN (user: ‘TechCrunch’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>24</td>
<td>.1582968</td>
<td>.0099724</td>
<td>.0488547</td>
<td>.1376673 .1789264</td>
</tr>
<tr>
<td>PFN</td>
<td>22</td>
<td>.0885927</td>
<td>.0200651</td>
<td>.0941136</td>
<td>.0468651 .1303204</td>
</tr>
<tr>
<td>combined</td>
<td>46</td>
<td>.1249601</td>
<td>.0119716</td>
<td>.0811952</td>
<td>.1008481 .1490721</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>.0697041</td>
<td>.0218401</td>
<td>.0256882</td>
<td>.0256882 .1137199</td>
</tr>
</tbody>
</table>

Degrees of freedom: 44

\[ Ho: \text{mean}(0) - \text{mean}(1) = \text{diff} = 0 \]

\[ Ha: \text{diff} < 0 \quad Ha: \text{diff} = 0 \quad Ha: \text{diff} > 0 \]

\[ t = 3.1916 \quad t = 3.1916 \quad t = 3.1916 \]

\[ P < t = 0.9987 \quad P > t = 0.0026 \quad P > t = 0.0013 \]

### Table B.14: Two-sample t test with equal variances: filtered CFN vs. filtered PFN (user: ‘TechCrunch’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>24</td>
<td>.1939135</td>
<td>.0142286</td>
<td>.0697055</td>
<td>.1644794 .2233476</td>
</tr>
<tr>
<td>filtered PFN</td>
<td>22</td>
<td>.1125253</td>
<td>.0299094</td>
<td>.1402876</td>
<td>.0503252 .1747253</td>
</tr>
<tr>
<td>combined</td>
<td>46</td>
<td>.1549887</td>
<td>.0170404</td>
<td>.1155739</td>
<td>.1206675 .1893099</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>.0813882</td>
<td>.0322429</td>
<td>.0164069</td>
<td>.1463096</td>
</tr>
</tbody>
</table>

Degrees of freedom: 44

\[ Ho: \text{mean}(0) - \text{mean}(1) = \text{diff} = 0 \]

\[ Ha: \text{diff} < 0 \quad Ha: \text{diff} = 0 \quad Ha: \text{diff} > 0 \]

\[ t = 2.5242 \quad t = 2.5242 \quad t = 2.5242 \]

\[ P < t = 0.9924 \quad P > t = 0.0153 \quad P > t = 0.0076 \]
### Appendix B

B.3 Tables for user ‘TechCrunch’

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFN</td>
<td>24</td>
<td>.1582968</td>
<td>.0099724</td>
<td>.0488547</td>
<td>.1376673 -.1789264</td>
</tr>
<tr>
<td>RFN</td>
<td>22</td>
<td>.1653415</td>
<td>.0091366</td>
<td>.0428544</td>
<td>.1463409 -.184342</td>
</tr>
<tr>
<td>combined</td>
<td>46</td>
<td>.161666</td>
<td>.0067399</td>
<td>.0457122</td>
<td>.1480911 -.1752408</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.0070446</td>
<td>.0136036</td>
<td></td>
<td>-.0203717 .0344609</td>
</tr>
</tbody>
</table>

Degrees of freedom: 44

\[
H_0 : \text{mean}(0) - \text{mean}(1) = \text{diff} = 0
\]

\[
H_a : \text{diff} < 0 \quad H_a : \text{diff} ! = 0 \quad H_a : \text{diff} > 0
\]

\[
t = -0.5178 \quad t = -0.5178 \quad t = -0.5178
\]

\[
P < t = 0.3036 \quad P > t = 0.6072 \quad P > t = 0.6964
\]

Table B.15: Two-sample t test with equal variances: CFN vs. RFN (user: ‘TechCrunch’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered CFN</td>
<td>24</td>
<td>.1939135</td>
<td>.0142286</td>
<td>.0697055</td>
<td>.1644794 .2233476</td>
</tr>
<tr>
<td>filtered RFN</td>
<td>22</td>
<td>.2348203</td>
<td>.0225086</td>
<td>.1055745</td>
<td>.1880112 .2816295</td>
</tr>
<tr>
<td>combined</td>
<td>46</td>
<td>.2134776</td>
<td>.0132794</td>
<td>.090065</td>
<td>.1867316 .2402236</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.0409068</td>
<td>.0261674</td>
<td></td>
<td>-.0936437 .0118301</td>
</tr>
</tbody>
</table>

Degrees of freedom: 44

\[
H_0 : \text{mean}(0) - \text{mean}(1) = \text{diff} = 0
\]

\[
H_a : \text{diff} < 0 \quad H_a : \text{diff} ! = 0 \quad H_a : \text{diff} > 0
\]

\[
t = -1.5633 \quad t = -1.5633 \quad t = -1.5633
\]

\[
P < t = 0.0626 \quad P > t = 0.1252 \quad P > t = 0.9374
\]

Table B.16: Two-sample t test with equal variances: filtered CFN vs. filtered RFN (user: ‘TechCrunch’)

89
### Table B.17: Two-sample t test with equal variances: $PFN$ vs. $RFN$ (user: ‘TechCrunch’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$PFN$</td>
<td>22</td>
<td>.0431382</td>
<td>.0534064</td>
<td>.2504981</td>
<td>-.0679264 .1542029</td>
</tr>
<tr>
<td>$RFN$</td>
<td>22</td>
<td>.1653415</td>
<td>.0091366</td>
<td>.0428544</td>
<td>.1463409 .184342</td>
</tr>
<tr>
<td>combined</td>
<td>44</td>
<td>.1042398</td>
<td>.0283493</td>
<td>.1880482</td>
<td>.0470679 .1614117</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.1222032</td>
<td>.0541823</td>
<td>-.2315475</td>
<td>-.012859</td>
</tr>
</tbody>
</table>

Degrees of freedom: 42

$Ho : mean(0) - mean(1) = diff = 0$

$Ha : diff < 0 \quad Ha : diff \neq 0 \quad Ha : diff > 0$

$t = -.2554 \quad t = -2.2554 \quad t = -2.2554$

$P < t = 0.0147 \quad P > t = 0.0294 \quad P > t = 0.9853$

---

### Table B.18: Two-sample t test with equal variances: filtered $PFN$ vs. filtered $RFN$ (user: ‘TechCrunch’)

<table>
<thead>
<tr>
<th>Group</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Err.</th>
<th>Std. Dev.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>filtered $PFN$</td>
<td>22</td>
<td>.0670707</td>
<td>.0587181</td>
<td>.2754121</td>
<td>-.0550402 .1891816</td>
</tr>
<tr>
<td>filtered $RFN$</td>
<td>22</td>
<td>.2348203</td>
<td>.0225086</td>
<td>.1055745</td>
<td>.1880112 .2816295</td>
</tr>
<tr>
<td>combined</td>
<td>44</td>
<td>.1509455</td>
<td>.0336039</td>
<td>.2229033</td>
<td>.0831767 .2187143</td>
</tr>
<tr>
<td>diff</td>
<td></td>
<td>-.1677496</td>
<td>.0628844</td>
<td>-.294655</td>
<td>-.2946555 .0408438</td>
</tr>
</tbody>
</table>

Degrees of freedom: 42

$Ho : mean(0) - mean(1) = diff = 0$

$Ha : diff < 0 \quad Ha : diff \neq 0 \quad Ha : diff > 0$

$t = -2.6676 \quad t = -2.6676 \quad t = -2.6676$

$P < t = 0.0054 \quad P > t = 0.0108 \quad P > t = 0.9946$
Appendix C

C.1 Python Code for CFN (T-based network)

The Python code for CFN is also available at:
http://in5000.googlecode.com/svn/trunk/CFN.py

C.2 Python Code for IFN (Meta-level network)

The Python code for IFN is also available at:
http://in5000.googlecode.com/svn/trunk/IFN.py

C.3 Python Code for PFN (LP-based network)

The Python code for PFN is also available at:
http://in5000.googlecode.com/svn/trunk/PFN.py

C.4 Python Code for RFN (CF-based network)

The Python code for RFN is also available at:
http://in5000.googlecode.com/svn/trunk/RFN.py

C.5 Python Code for NM (Administration script)

The Python code for NM is available at:
http://in5000.googlecode.com/svn/trunk/NM.py

C.6 Python Code for IFN2PFN (Meta-level network conversion utility)

The Python code for IFN2PFN is available at:
http://in5000.googlecode.com/svn/trunk/IFN2PFN.py

C.7 Python Code for ICF (Semantic entity extractor utility)

The Python code for ICF is available at:
http://in5000.googlecode.com/svn/trunk/ICF.py
C.8 Java Code for ItemBasedRecommender

The Java code for the ItemBasedRecommender is available at:
http://in5000.googlecode.com/svn/trunk/ItemBasedRecommender.java

C.9 Java Code for UserBasedEvaluator

The Java code for the UserBasedEvaluator is available at:
http://in5000.googlecode.com/svn/trunk/UserBasedEvaluator.java
Appendix D

D.1 Results

In this appendix we report the results from two-tailed t-tests with equal and unequal variances.

D.1.1 Results for hypotheses 1 and 2

Results for hypotheses 1 and 2 for user ‘Mashable’

- There was a significant difference in the scores for precision in \( T \) (\( M = .1358144, SD = .0293552 \)) and precision in \( LP - based \) (\( M = .1013269, SD = .1495961 \)); \( t(109) = 1.6925, p = 0.0467 \). Therefore, the alternate hypothesis is supported in this case, namely that \( T\mu > LP_{based}\mu \).

- There was a significant difference in the scores for precision in filtered \( T \) (\( M = .1543043, SD = .0260313 \)) and precision in filtered \( LP - based \) (\( M = .1673389, SD = .2043398 \)); \( t(83) = -5.8990, p = 0.0000 \). Therefore, the alternate hypothesis is supported in this case, namely that filtered \( T_{\mu_{filtered}} > LP_{\mu_{based_{filtered}}} \).

Results for hypotheses 1 and 2 for user ‘BoingBoing’

- There was a significant difference in the scores for precision in \( T \) (\( M = .0541214, SD = .0205047 \)) and precision in \( LP - based \) (\( M = .1330803, SD = .0852966 \)); \( t(83) = -5.8990, p = 0.0000 \). Therefore, the null hypothesis is supported in this case, namely that \( T\mu < LP_{based}\mu \).

- There was a significant difference in the scores for precision in filtered \( T \) (\( M = .137097, SD = .1117784 \)) and precision in filtered \( LP - based \) (\( M = .274337, SD = .325813 \)); \( t(83) = -2.6097, p = 0.0054 \). Therefore, the null hypothesis is supported in this case, namely that filtered \( T_{\mu_{filtered}} < LP_{\mu_{based_{filtered}}} \).

Results for hypotheses 1 and 2 for user ‘TechCrunch’

- There was a significant difference in the scores for precision in \( T \) (\( M = .1582968, SD = .0488547 \)) and precision in \( LP - based \) (\( M = .0885927, SD = .0941136 \)); \( t(81) = 3.1916, p = 0.0013 \). Therefore, the alternate hypothesis is supported in this case, namely that \( T\mu > LP_{based}\mu \).

- There was a significant difference in the scores for precision in filtered \( T \) (\( M = .1939135, SD = .0697055 \)) and precision in filtered \( LP - based \) (\( M = .2654370, SD = .1882374 \)); \( t(80) = -3.3989, p = 0.0008 \). Therefore, the alternate hypothesis is supported in this case, namely that filtered \( T_{\mu_{filtered}} > LP_{\mu_{based_{filtered}}} \).
D.1 Results

Appendix D

.1125253, SD = .1402876); t(81) = 2.5242, p = 0.0076. Therefore, the alternate hypothesis is supported in this case, namely that filtered $T_\mu^{\text{filtered}} > LP - based_\mu^{\text{filtered}}$.

D.1.2 Results for hypotheses 3 and 4

Results for hypotheses 3 and 4 for user ‘Mashable’

- There was a significant difference in the scores for precision in $T$ ($M = .1358144, SD = .0293552$) and precision in $CF - based$ ($M = .1501967, SD = .0305297$); $t(107) = -2.5073, p = 0.0068$. Therefore, the null hypothesis is supported in this case, namely that $T_\mu < CF - based_\mu$.

- There was a significant difference in the scores in filtered $T$ ($M = .2260434, SD = .0614484$) and precision in filtered $CF - based$ ($M = .2681347, SD = .0905981$); $t(107) = -2.8523, p = 0.0026$. Therefore, the null hypothesis is supported in this case, namely that filtered $T_\mu^{\text{filtered}} < CF - based_\mu^{\text{filtered}}$.

Results for hypotheses 3 and 4 for user ‘BoingBoing’

- There was a significant difference in the scores for precision in $T$ ($M = .0541214, SD = .0205047$) and precision in $CF - based$ ($M = .1684375, SD = .066203$); $t(81) = -10.7849, p = 0.0000$. Therefore, the null hypothesis is supported in this case, namely that $T_\mu < CF - based_\mu$.

- There was a significant difference in the scores in filtered $T$ ($M = .137097, SD = .1117784$) and precision in filtered $CF - based$ ($M = .2689958, SD = .2621584$); $t(81) = -3.0185, p = 0.0017$. Therefore, the null hypothesis is supported in this case, namely that filtered $T_\mu^{\text{filtered}} < CF - based_\mu^{\text{filtered}}$.

Results for hypotheses 3 and 4 for user ‘TechCrunch’

- There was not a significant difference in the scores for precision in $T$ ($M = .1939135, SD = .0697055$) and precision in $CF - based$ ($M = .2348203, SD = .1055745$); $t(81) = -1.5633, p = 0.1252$. Therefore, the alternate hypothesis is supported in this case, namely that $T_\mu = CF - based_\mu$.

- There was a significant difference in the scores for precision in filtered $T$ ($M = .0431382, SD = .2504981$) and precision in filtered $CF - based$ ($M = .1653415, SD = .0425844$); $t(81) = -2.2554, p = 0.0147$. Therefore, the null hypothesis is supported in this case, namely that filtered $T_\mu^{\text{filtered}} < CF - based_\mu^{\text{filtered}}$.

D.1.3 Results for hypotheses 5 and 6

Results for hypotheses 5 and 6 for user ‘Mashable’

- There was a significant difference in the scores for $LP - based$ ($M = .1013269, SD = .1495961$) and precision in $CF - based$ ($M = .1501967, SD = .0305297$); $t(107) = -2.2004, p = 0.0150$. Therefore, the alternate hypothesis is supported in this case, namely that $LP - based_\mu < CF - based_\mu$.

- There was a significant difference in the scores for precision in filtered $LP - based$ ($M = .2681347, SD = .0905981$); $t(107) = -3.1191, p = 0.0012$. Therefore, the alternate hypothesis is supported in this case, namely that filtered $LP - based_\mu^{\text{filtered}} < CF - based_\mu^{\text{filtered}}$. 

94
Appendix D  

D.2 Summary Tables of Results

Results for hypotheses 5 and 6 for user ‘BoingBoing’

- There was a significant difference in the scores for precision in $LP - based$ ($M = .1330803, SD = .0852966$) and precision in $CF - based$ ($M = .1684375, SD = .0662033$); $t(80) = -2.0897, p = 0.0199$. Therefore, the alternate hypothesis is supported in this case, namely that $LP - based \mu < CF - based \mu$.

- There was not a significant difference in the scores for precision in filtered $LP - based$ ($M = .137097, SD = .325813$) and precision in filtered $CF - based$ ($M = .2689958, SD = .2621584$); $t(80) = -3.0185, p = 0.0017$. Therefore, it is inconclusive whether $LP - based_{filtered} \mu < CF - based_{filtered} \mu$ or $LP - based_{filtered} \mu > CF - based_{filtered} \mu$. Rather, the t-test supports the proposition that $LP - based_{filtered} \mu = CF - based_{filtered} \mu$.

Results for hypotheses 5 and 6 for user ‘TechCrunch’

- There was a significant difference in the scores for precision in $LP - based$ ($M = .0431382, SD = .2504981$) and precision in $CF - based$ ($M = .1653415, SD = .0428544$); $t(78) = -2.2554, p = 0.0147$. Therefore, the alternate hypothesis is supported in this case, namely that $LP - based \mu < CF - based \mu$.

- There was a significant difference in the scores for precision in filtered $LP - based$ ($M = .0670707, SD = .2754121$) and precision in filtered $CF - based$ ($M = .2348203, SD = .1055745$); $t(78) = -2.6676, p = 0.0054$. Therefore, the alternate hypothesis is supported in this case, namely that filtered $LP - based_{filtered} \mu < CF - based_{filtered} \mu$.

D.2  Summary Tables of Results

D.2.1  Summary Tables for User: Mashable
### Summary Findings and Statistical Tests Used

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Supported?</th>
<th>$\mu_1$</th>
<th>$\sigma_1$</th>
<th>$\mu_2$</th>
<th>$\sigma_2$</th>
<th>DF</th>
<th>t-value</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $T_\mu &lt; LP - based_\mu$</td>
<td>No</td>
<td>.1013269</td>
<td>.0905981</td>
<td>.1501967</td>
<td>.0305297</td>
<td>107</td>
<td>-2.004</td>
<td>0.0015</td>
</tr>
<tr>
<td>2. $T_{filtered} &lt; LP - based_{filtered}$</td>
<td>No</td>
<td>.1673389</td>
<td>.204398</td>
<td>.2681347</td>
<td>.0905981</td>
<td>107</td>
<td>-3.119</td>
<td>0.0012</td>
</tr>
</tbody>
</table>

Table D.1: Summary Findings and Statistical Tests Used. (Source User: ‘Mashable’)
D.2.2 Summary Tables for User: BoingBoing
<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Supported?</th>
<th>$\mu_1$</th>
<th>$\sigma_1$</th>
<th>$\mu_2$</th>
<th>$\sigma_2$</th>
<th>DF</th>
<th>t-value</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $T_\mu &lt; LP - based_\mu$</td>
<td>Yes</td>
<td>.0541214</td>
<td>.0205047</td>
<td>.1330803</td>
<td>.0852966</td>
<td>83</td>
<td>-5.8990</td>
<td>0.0000</td>
</tr>
<tr>
<td>2. $T_\mu^{filtered} &lt; LP - based_\mu^{filtered}$</td>
<td>Yes</td>
<td>.1358144</td>
<td>.1117784</td>
<td>.274337</td>
<td>.325813</td>
<td>83</td>
<td>-2.6097</td>
<td>0.0054</td>
</tr>
<tr>
<td>3. $T_\mu &lt; CF - based_\mu$</td>
<td>Yes</td>
<td>.0541214</td>
<td>.0205047</td>
<td>.1684375</td>
<td>.066203</td>
<td>81</td>
<td>-10.7849</td>
<td>0.0000</td>
</tr>
<tr>
<td>4. $T_\mu^{filtered} &lt; CF - based_\mu^{filtered}$</td>
<td>Yes</td>
<td>.137097</td>
<td>.1117784</td>
<td>.2689958</td>
<td>.2621584</td>
<td>81</td>
<td>-3.0185</td>
<td>0.0017</td>
</tr>
<tr>
<td>5. $LP - based_\mu &gt; CF - based_\mu$</td>
<td>No $H_0$: $LP - based_\mu^{filtered} &lt; CF - based_\mu^{filtered}$</td>
<td>.1330803</td>
<td>.0852966</td>
<td>.1684375</td>
<td>.066203</td>
<td>80</td>
<td>-2.0897</td>
<td>0.0199</td>
</tr>
<tr>
<td>6. $LP - based_\mu^{filtered} &gt; CF - based_\mu^{filtered}$</td>
<td>No $H_0$: $LP - based_\mu^{filtered} = CF - based_\mu^{filtered}$</td>
<td>.137097</td>
<td>.325813</td>
<td>.2689958</td>
<td>.2621584</td>
<td>80</td>
<td>-3.0185</td>
<td>0.0017</td>
</tr>
</tbody>
</table>

Table D.2: Summary Findings and Statistical Tests Used. (Source User: ‘BoingBoing’)
D.2.3 Summary Tables for User: TechCrunch
### Summary Findings and Statistical Tests Used

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Supported?</th>
<th>$\mu_1$</th>
<th>$\sigma_1$</th>
<th>$\mu_2$</th>
<th>$\sigma_2$</th>
<th>DF</th>
<th>t-value</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $T_\mu &lt; LP - based_\mu$</td>
<td>No</td>
<td>$H_0$: $T_\mu &gt; LP - based_\mu$</td>
<td>.1582968</td>
<td>.0488547</td>
<td>.0885927</td>
<td>.0941136</td>
<td>81</td>
<td>3.1916</td>
</tr>
<tr>
<td>2. $T_{\mu, filtered} &lt; LP - based_{filtered}$</td>
<td>No</td>
<td>$H_0$: $T_{\mu, filtered} &gt; LP - based_{filtered}$</td>
<td>.1939135</td>
<td>.0697055</td>
<td>.1125253</td>
<td>.1402876</td>
<td>81</td>
<td>2.5242</td>
</tr>
<tr>
<td>3. $T_\mu &lt; CF - based_\mu$</td>
<td>No</td>
<td>$H_0$: $T_\mu = CF - based_\mu$</td>
<td>.1582968</td>
<td>.0488547</td>
<td>.1653415</td>
<td>.0428544</td>
<td>81</td>
<td>-0.5178</td>
</tr>
<tr>
<td>4. $T_{\mu, filtered} &lt; CF - based_{filtered}$</td>
<td>Yes</td>
<td></td>
<td>.1939135</td>
<td>.0697055</td>
<td>.2348203</td>
<td>.1055745</td>
<td>81</td>
<td>-1.5633</td>
</tr>
<tr>
<td>5. $LP - based_\mu &gt; CF - based_\mu$</td>
<td>No</td>
<td>$H_0$: $LP - based_\mu &lt; CF - based_\mu$</td>
<td>.0431382</td>
<td>.2504981</td>
<td>.1653415</td>
<td>.0428544</td>
<td>78</td>
<td>-2.2554</td>
</tr>
<tr>
<td>6. $LP - based_{filtered} &gt; CF - based_{filtered}$</td>
<td>No</td>
<td>$H_0$: $LP - based_{filtered} &lt; CF - based_{filtered}$</td>
<td>.0670707</td>
<td>.2754121</td>
<td>.2348203</td>
<td>.1055745</td>
<td>78</td>
<td>-2.6676</td>
</tr>
</tbody>
</table>

Table D.3: Summary Findings and Statistical Tests Used. (Source: 'TechCrunch')
D.3 Graph Plots

D.3.1 Graph Plots for hypotheses 1 and 2

Figure D.1: $\mu^{(f)}_T > \mu^{(f)}_{LP-based}$ (Source User: ‘Mashable’)

Figure D.2: $\mu^{(f)}_T < \mu^{(f)}_{LP-based}$ (Source User: ‘BoingBoing’)
D.3 Graph Plots

D.3.2 Graph Plots for hypotheses 3 and 4

D.3.3 Graph Plots for hypotheses 5 and 6
Figure D.4: $T_{\mu}^{(f)} < CF - based_{\mu}^{(f)}$ (Source User: ‘Mashable’)

Figure D.5: $T_{\mu}^{(f)} < CF - based_{\mu}^{(f)}$ (Source User: ‘BoingBoing’)

Appendix D  
D.3 Graph Plots
D.3 Graph Plots

**Figure D.6:** $T^{(f)}_\mu \text{ CF – based}_\mu^{(f)}$ (Source User: ‘TechCrunch’)

**Figure D.7:** $LP – \text{ based}_\mu^{(f)} < CF – \text{ based}_\mu^{(f)}$ (Source User: ‘Mashable’)

104
Figure D.8: $LP - based \mu(f) < CF - based \mu(f)$ (Source User: ‘BoingBoing’)

Figure D.9: $LP - based \mu(f) < CF - based \mu(f)$ (Source User: ‘TechCrunch’)

Appendix D  
D.3 Graph Plots
Interestingness, or potential precision, is the average maximum precision over time that is measured as shown in Figure E.1 (source code lines 69 - 71 show how precision is calculated and accumulated, and source code line 85 shows how a single average precision is returned).

Inputs:

- Ground-truth = $T$ network of manually configured followees, constructed as follows:
  - Get 30 latest (manually configured) followees from root user (from user profile on Twitter)
  - Collect three most recent tweets with URL entities per information stream of each user:
    * root user
    * 30 latest followees
    * 30 latest followees-of-followees (ie. the 30 latest (manually configured) followees of the 30 latest (manually configured) followees)

- $CF$ - based or $LP$ - based network = network of $CF$ - based or $LP$ - based predictions, constructed as follows:
  - Get 30 predicted followees from root user
  - Collect three most recent tweets with URL entities per information stream of each user:
    * root user
    * 30 predicted followees
    * 30 latest manually configured followees-of-followees (ie. the 30 latest manually configured followees of the 30 predicted followees)

- Precision = potential precision as running average precision over all users (aprox. 1000) in ground truth ($T$) and $CF$ - based and $LP$ - based networks

Pseudo-code sketch of the evaluation algorithm:

Every 8 hours do:

```
GroundTruthNetwork T = buildGroundTruthNetwork(user);
IRStatistics statsGroundTruthNetwork = evaluator.evaluate(<params>);
averagePrecision_T = statsGroundTruthNetwork.getPrecision();
```
ImplicitFolloweeNetwork ifn = buildImplicitFolloweeNetwork(user);

CFBasedNetwork CFBased = buildCFBasedNetwork(ifn, user);
IRStatistics statsCFBasedNetwork = evaluator.evaluate(<params>);
averagePrecision_CFBased = statsCFBasedNetwork.getPrecision();

LPBasedNetwork LPBased = buildLPBasedNetwork(ifn, user);
IRStatistics statsLPBasedNetwork = evaluator.evaluate(<params>);
averagePrecision_LPBased = statsLPBasedNetwork.getPrecision();

return averagePrecision_T, averagePrecision_CFBased, averagePrecision_LPBased

Outputs:
- date/time
- number of users
- number of items
- number of filtered users
- number of filtered items
- precision (potential precision)
- precision filtered (potential precision)

Filtered Alternative Followee Networks. As an additional step we employed semantic enrichment to filter tweets that would likely be interesting for users based on their semantic profiles extracted using the TweetUM webservice [48]. This step resulted in filtered versions of the ground-truth \((T)\) and \(CF-based\) and \(LP-based\) networks. These filtered networks had about 30% to 40% less users and items. The construction and evaluation of such networks only differed in the additional filtering step. Thus, the above pseudo-code is also useful to understand how we constructed and evaluated filtered networks.
1 @Override
2 public IRStatistics evaluate(RecommenderBuilder recommenderBuilder,
3                                DataModelBuilder dataModelBuilder,
4                                DataModel dataModel,
5                                IDRescorer rescorer,
6                                int at,
7                                double relevanceThreshold,
8                                double evaluationPercentage) throws TasteException {
9    Preconditions.checkNotNull(recommenderBuilder != null, "recommenderBuilder is null");
10   Preconditions.checkNotNull(dataModel != null, "dataModel is null");
11   Preconditions.checkNotNull(at >= 1, "at must be at least 1");
12   Preconditions.checkNotNull(evaluationPercentage > 0.0 && evaluationPercentage <= 1.0,
13       "Invalid evaluationPercentage: %s", evaluationPercentage);
14
15   int numItems = dataModel.getNumItems();
16   RunningAverage precision = new FullRunningAverage();
17   RunningAverage recall = new FullRunningAverage();
18   RunningAverage fallOut = new FullRunningAverage();
19   LongPrimitiveIterator it = dataModel.getUserIDs();
20   while (it.hasNext()) {
21     long userID = it.nextLong();
22     if (random.nextDouble() < evaluationPercentage) {
23       long start = System.currentTimeMillis();
24       FastIDSet relevantItemIDs = new FastIDSet(at);
25       PreferenceArray prefs = dataModel.getPreferencesFromUser(userID);
26       int size = prefs.length();
27       if (size < 2 * at) {
28         // Really not enough prefs to meaningfully evaluate this user
29         continue;
30       }
31
32       // List some most-preferred items that would count as (most) "relevant" results
33       double theRelevanceThreshold =
34           Double.isNaN(relevanceThreshold) ? computeThreshold(prefs) : relevanceThreshold;
35       prefs.sortByValueReversed();
36       for (int i = 0; (i < size) && (relevantItemIDs.size() < at); i++) {
37         if (prefs.getValue(i) >= theRelevanceThreshold) {
38           relevantItemIDs.add(prefs.getItemID(i));
39         }
40       }
41       int numRelevantItems = relevantItemIDs.size();
42       if (numRelevantItems > 0) {
43         FastByIDMap<PreferenceArray> trainingUsers = new FastByIDMapPreferenceArray(dataModel
44             .getNumUsers());
45         LongPrimitiveIterator it2 = dataModel.getUserIDs();
46         while (it2.hasNext()) {
47           processOtherUser(userID, relevantItemIDs, trainingUsers, it2
48               .nextLong(), dataModel);
49         }
50
51         DataModel trainingModel = dataModelBuilder == null ? new GenericDataModel(trainingUsers)
52             : dataModelBuilder.buildDataModel(trainingUsers);
53         Recommender recommender = recommenderBuilder.buildRecommender(trainingModel);
54
55         try {
56           trainingModel.getPreferencesFromUser(userID);
57         } catch (NoSuchUserException nsee) {
58           continue; // Oops we excluded all prefs for the user -- just move on
59         }
60
61         int intersectionSize = 0;
62         List<RecommendedItem> recommendedItems = recommender.recommend(userID, at, rescorer);
63         for (RecommendedItem item : recommendedItems) {
64           boolean contains = relevantItemIDs.contains(item.getItemID());
65           if (contains) {
66             intersectionSize++;
67           }
68           int numRecommendedItems = recommendedItems.size();
69           recall.addDatum((double) intersectionSize / (double) numRecommendedItems);
70           precision.addDatum((double) intersectionSize / (double) numRecommendedItems);
71           recall.addDatum((double) intersectionSize / (double) numRecommendedItems);
72           if (numRecommendedItems < size) {
73             fallOut.addDatum((double) (numRecommendedItems - intersectionSize)
74                 / (double) (numItems - numRecommendedItems));
75           }
76
77         long end = System.currentTimeMillis();
78         GenericRecommenderIRStatsEvaluator.log.info("Evaluated with user {} in {}ms", userID, end - start);
79         log.info("Precision/recall/fall-out: {} / {} / {}",
80             new Object[] {precision.getAverage(), recall.getAverage(), fallOut.getAverage()});
81       }
82     }
83   }
84
85   return new IRStatisticsImpl(precision.getAverage(), recall.getAverage(), fallOut.getAverage());
86  }

Figure E.1: Source Code of org.apache.mahout.cf.taste.impl.eval.GenericRecommenderIRStatsEvaluator.evaluate
Bibliography


