Optimization Techniques for Data Analytics

Ming Fai Felix Wong

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Electrical Engineering
Adviser: Mung Chiang

March 2015

Abstract

As new sources of large-scale data with increasing volume and complexity are being created, finding scalable ways to gain insights from unstructured big data has become a big challenge. Furthermore, our big data challenge is exacerbated by the data often being noisy, sparse and heterogeneous. This dissertation illustrates, through three studies, the benefits of using an optimization framework to devise methods for big data analytics.

We study three problems in computational political science, finance and recommender systems, analyzing a wide range of data, including time series, text, ratings and social networks. First, we propose an inference technique to quantify the political leaning of Twitter users based on the patterns of how they get retweeted. We apply the technique to Twitter data collected during the U.S. presidential election of 2012. Second, we propose a joint latent space model for stock price movements and word usage patterns in newspapers. We apply the model and develop an algorithm to predict stock closing prices of a given day using the full text of *The Wall Street Journal* of the morning. Finally, we study the fundamental question of evaluating the quality of a social recommender network. We propose a pair of metrics to quantify (a) a network's efficiency in disseminating recommendations and (b) the quality of a user's neighbors in the network. Then we empirically study their trade-off on Yelp data, and devise an algorithm to improve a network's quality through friend recommendation and news feed curation.

Acknowledgements

About five years ago when my first semester at Princeton had almost ended, I was looking for an advisor but had no clear idea of which research field to pursue. One day, I saw on the department homepage that my to-be advisor, Prof. Mung Chiang, had recently established the Princeton EDGE Lab with a thrust in social networks research. With some research experience in online social networks, I immediately approached him, and very luckily, I was admitted to his research group. It has been quite a journey since then. My original goal was to (a) do research in online social networks (b) using real large-scale data and (c) learn a few mathematical optimization techniques along the way. After a few research dead-ends, a few awards, and a few family-related events, I am glad that the final outcome, *i.e.,* this dissertation, does not deviate too much from my original goal. Here I would like to express my gratitude to Mung for the advice and the opportunities he offered.

The research in this dissertation would not have been possible without the financial support from National Science Foundation, Army Research Office and the Research Grants Council of Hong Kong. Equally important is the support from several organizations, especially Twitter and Yelp, by making high-quality data available to the research community. I would also like to thank Prof. Paul Cuff, Prof. Sanjeev Kulkarni and Prof. Prateek Mittal for serving on my general exam and/or FPO committees.

Many past and current members of the Princeton EDGE Lab have contributed directly or indirectly to this dissertation. The diversity of the members' backgrounds enables me to maintain a healthy balance, both between theory and practice in research, and between work and leisure in life. I would particularly like to thank three persons: Prof. Sangtae Ha, Dr. Zhenming Liu and Prof. Chee Wei Tan, for the countless hours of research discussions, and the mentorship that extends far beyond academics. Chee Wei also hosted my academic visit to the City University of Hong Kong. Other lab members and visitors with whom I have interacted include:[1] Ehsan Aryafar, Bharath Balasubramanian, Chris Brinton, Swapna

[1]Titles omitted since almost everyone in this list will eventually be a Dr. or Prof.

Buccapatnam, Jiasi Chen, Aveek Dutta, Amitabha Ghosh, Felix Huang, Joe Jiang, Carlee Joe-Wong, Ioannis Kamitsos, Haris Kremo, Hongseok Kim, Tian Lan, Chris Leberknight, Ranjan Pal, Kartik Pandit, Rüdiger Rill, Soumya Sen, Michael Wang, Shirley Wang, Jie Xiang, Liang Zheng, Steve Zhuo.

Many thanks to Prof. Peter Marbach, my master's advisor at Toronto, for kickstarting my transition from an engineer to a researcher. After having written a few papers with him, no research problem looks insurmountable anymore. Also thanks to the members of the networking group at Toronto, especially Dr. Stratis Ioannidis, Larry Zhang and Lilin Zhang, for the friendship and assistance.

My graduate study is possible only with the blessing of several professors at CUHK. Thanks to Prof. John C.S. Lui and Prof. Yiu-Sang Moon for the support in all these years, which includes taking time to chat with me during my occasional visits to Hong Kong, and tolerating my occasional stubbornness and naivety. My four years in the Embedded System Laboratory (and the CSE department in general) at CUHK are fondly remembered.

Finally, I would like to thank my family. Thanks to Anna, my wife, for sharing the ups and downs of a graduate student and her sacrifices. Thanks to Ryan, my son, for being a good baby in the past year, and for many years to come. Thanks to my father, mother and sister for supporting my academic endeavors and their sacrifices. I dedicate this dissertation to my father, whose untimely passing will leave forever my greatest regrets. Being a man of ambition but deprived of opportunities of higher education, he used to talk about his unfulfilled childhood dream of going to college to learn calculus and become an aerospace engineer. I hope the fact that his son has learnt calculus (and much more) and has a PhD in electrical engineering (although not aerospace engineering) will make him feel better, and I really wish he had the chance to see the completion of this dissertation.

To my late father.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Big Data Deluge

Recent advances in technology have enabled the creation and collection of data in ever-increasing rates from sources such as social networking services, web and mobile services, e-commerce, scientific research, smart homes and healthcare. IBM estimated (in 2012) that 2.5 quintillion bytes of data are created every day, and 90% of the data in existence were created in the past two years [94].

It is a challenge to cope with the scale and volume of big data. The challenge is further complicated by the data being noisy, sparse and heterogeneous, as they often originate from individual uncoordinated sources, *e.g.,* users of a social networking service, and traffic sensors spread over a large geographical area. But with careful modeling choices and design decisions, this dissertation shows how it is possible to develop methods that can glean insights from big data, and moreover, to turn the learnt insights into actions that can improve our lives through recommendations and predictions.

1.2 A Data-driven Optimization Framework

This dissertation presents three studies of devising data analytics techniques on a variety of (unstructured) data, *e.g.,* time series, text, ratings and social networks, for applications in computational political science, finance and recommender systems. While the problem domains and the types of data appear diverse, the techniques we develop are unified under a data-driven optimization framework.

By *optimization* we focus on using *continuous* optimization techniques to develop methods that are:

- Scalable: we want algorithms to be expressable as one or multiple convex optimization problems so that they can be efficiently solved. It also allows us to apply many tools from the optimization literature.

- Robust: real data are rarely balanced, *e.g.,* individuals in a social network exhibit a heavy-tailed degree distribution, so it is important to devise methods that work well for entities with sparse data by, *e.g.,* borrowing data from entities with more data. This can be readily achieved by, *e.g.,* adding a regularization term to the objective function of an optimization problem.

- Minimal: we attempt to minimize a method's (a) number of control knobs or tunable parameters, so as to reduce the risk of overfitting and the effort needed for calibration; and (b) number of data preprocessing steps, so as to reduce the risk of discarding useful (but noisy) data in the process.

By *data-driven* we heavily involve data in modeling and algorithm development. In each study we focus on one set of data, and perform exploratory analysis to derive insights for modeling. Then, with a model in hand, we develop algorithms for inference, prediction or recommendation. The data is used again to evaluate our algorithms, with the results used to refine the model and the algorithm. This model-algorithm-evaluation cycle iterates until satisfactory results are achieved, as summarized in Figure 1.1.

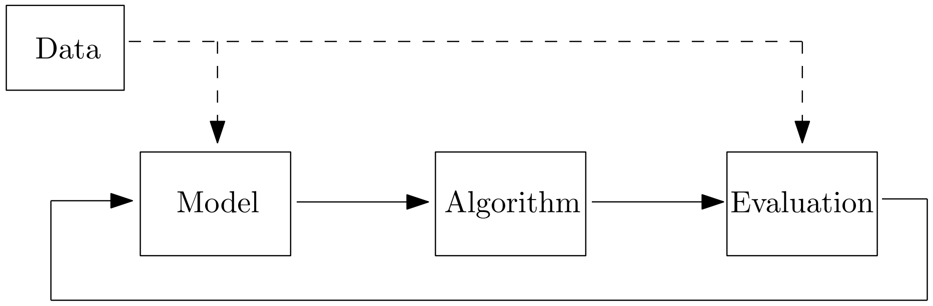


Figure 1.1: Data-driven optimization framework for data analytics.

1.3 Contributions

In this section we briefly motivate and summarize the three data analytics studies of this dissertation. They are organized into three chapters by the theme of the data analyzed.

1.3.1 Online Social Networks

Online social networks (OSNs) have fundamentally changed how people interact and exchange information. They democratize the spread of information and opinions by allowing ordinary OSN users to "share," "like" or "retweet" posts or messages of interest. Furthermore, the technological infrastructure of OSNs means all actions by all users are, with the users' permission, archivable and searchable. The availability of OSN data has enabled the field of computational social science [56] in taking a data-driven approach to study questions in economics, sociology and political science.

Twitter, the second most popular OSN after Facebook by traffic volume [5], has been the most popular source of data for research in OSNs. Through the Twitter API, researchers are able to obtain social media data which are (a) big, because of Twitter's popularity, (b) of high quality, *i.e.,* accurate with a plethora of metadata, and (c) in real-time, which is particularly important for researchers would like to track public opinions on real-time events in online social media. While Twitter offers many opportunities for big data analysis, the challenge lies in devising methods that are scalable (to cope with the volume of Twitter data) and robust to noise (tweets are characterized by short length and the frequent use of informal language).

The first contribution of this dissertation is in proposing an inference technique to quantify the political leaning of Twitter users. This technique can serve as a precursor to many applications, including studying social science questions, predicting election outcomes from social media, and personalizing news feeds. We formulate political leaning inference as a convex optimization problem that incorporates two insights from data: (a) users are consistent in their actions of tweeting and retweeting about political issues, and (b) similar users tend to be retweeted by similar audience. Then we apply our technique to 119 million election-related tweets collected in seven months during the 2012 U.S. presidential election campaign. Our technique achieves 94% accuracy and high rank correlation as compared with manually created labels. By studying the political leaning of 1,000 frequently retweeted sources, 230,000 ordinary users who retweeted them, and the hashtags used by these sources, our numerical study sheds light on the political demographics of the Twitter population, and the temporal dynamics of political polarization as events unfold.

1.3.2 Newspapers and Stocks

We next turn our attention to the intersection of text mining and time series analysis, using data from more traditional sources: newspapers and the stock market. We study the problem of predicting directional movements of stock prices based on news articles, and in particular, using *The Wall Street Journal* of one morning to predict (nowcast) the closing stock prices of the same day.

While this has been an oft-studied problem [71], we take a new approach inspired by collaborative filtering. We propose a unified latent space model to characterize co-movements between stock prices and news articles. Then we propose model learning as solving an optimization problem reminiscent of matrix factorization. To avoid overfitting, we introduce sparsity-inducing regularization so that few words are selected for the model (a feature selection problem) and each selected word is associated with few latent factors. Unlike many existing approaches, our new model is able to simultaneously leverage the correlations: (a) among stock prices, (b) among news articles, and (c) between stock prices and news articles. Thus, our model is able to make daily predictions on more than 500 stocks (most of which are not even mentioned in any news article) while having low complexity. We carry out extensive backtesting on a trading strategy based on our algorithm. The result shows that our model produces substantially better accuracy than those of many widely used prediction algorithms. The cumulative return and Sharpe ratio due to our trading strategy are also much higher than those due to baseline strategies.

We note a philosophical difference between existing approaches and ours. The usual approach is to first perform significant preprocessing on input text, *e.g.,* sentiment analysis and other natural language processing techniques, to obtain relatively "clean" signals, and then feed the signals to a relatively simple prediction algorithm. On the other hand, we take the approach where minimal preprocessing, *i.e.,* computing scaled word count time series, is performed on newspaper text, and rely on sophisticated optimization formulations to account for noise in data.

1.3.3 Social Recommender Networks

A social recommender network is an OSN built on top of a recommender system with the purpose of sharing reviews and recommendations. A crucial property of these networks is they are constructed in a decentralized manner, *i.e.,* two individuals have the freedom to decide whether to be linked, often without knowledge of the global properties of the network.

While there has been much work in studying the role of social networks in rumor spreading [53, 19, 18, 1] and exploiting social network information to improve recommender system performance [64], there has been little work in evaluating the quality of a social recommender network. Even the notion of "quality" itself requires some consideration: intuitively, a high-quality network should be efficient in disseminating recommendations, but if the network is constructed to blindly optimize efficiency, *e.g.,* randomly matching users to construct a random graph (which are known to be efficient under many metrics), user experience may suffer as dissimilar users are matched and low-quality recommendations are exchanged. Thus, a rigorous evaluation of social recommender networks requires a formal definition of quality in terms of both efficiency and user experience.

The third contribution of this dissertation attempts to close the gap in three directions. First, we introduce a stylized stochastic model for recommendation diffusion. Such a model allows us to highlight the connection between user experience at the individual level, and network efficiency at the macroscopic level. We also propose a set of metrics for quantifying both user experience and network efficiency. Second, based on these metrics, we extensively study the tradeoff between the two factors in a Yelp dataset, concluding that Yelp's social network is surprisingly efficient, though not optimal. Finally, we design a friend recommendation and news feed curation algorithm that can simultaneously address individuals' need to connect to high quality friends, and service providers' need to improve network efficiency in information propagation.

1.3.4 Publications

Parts of this dissertation have appeared in the following peer-reviewed publications:

- F. M. F. Wong, C. W. Tan, S. Sen, and M. Chiang, "Quantifying political leaning from tweets and retweets," in *Proc. International AAAI Conference on Weblogs and Social Media (ICWSM)*, Boston, MA, July 2013.

- F. M. F. Wong, Z. Liu, and M. Chiang, "Stock market prediction from WSJ: Text mining via sparse matrix factorization," in *Proc. IEEE International Conference on Data Mining (ICDM)*, Shenzhen, China, December 2014.

- F. M. F. Wong, Z. Liu, and M. Chiang, “On the efficiency of social recommender networks,” in *Proc. IEEE Conference on Computer Communications (INFOCOM)*, Hong Kong, China, April 2015.

In addition, parts of Chapter 2 are currently under peer review:

- F. M. F. Wong, C. W. Tan, S. Sen, and M. Chiang, “Quantifying political leaning from tweets, retweets, and retweeters,” in submission, 2015.

Chapter 2

Quantifying Political Leaning from Tweets, Retweets, and Retweeters

2.1 Introduction

In recent years, big online social media data have found many applications in the intersection of political and computer science. Examples include answering questions in political and social science (*e.g.,* proving/disproving the existence of media bias [6, 58] and the "echo chamber" effect [2, 8]), using online social media to predict election outcomes [90, 59], and personalizing social media feeds so as to provide a fair and balanced view of people's opinions on controversial issues [75]. A prerequisite for answering the above research questions is the ability to accurately estimate the political leaning of the population involved. If it is not met, then for the above examples, either the conclusion will be invalid, the prediction will perform poorly [70, 77] due to a skew towards highly vocal individuals [62], or user experience will suffer.

In the context of Twitter, accurate political leaning estimation poses two key challenges: (a) Quantification: Is it possible to assign meaningful numerical scores to tweeters about their position in the political spectrum? (b) Scalability: Given Twitter's large scale and server limitations, how can we devise a method that is efficient and scalable? We propose a new approach that incorporates the following two sets of information to infer the political leaning of a target set of Twitter users.

Tweets and *retweets*: the target users' temporal patterns of being retweeted, and the tweets published by their retweeters. The insight is that a user's tweet contents should be consistent with who they retweet, *e.g.,* if a user tweets a lot during a political event, she is expected to also retweet a lot at the same time. This is the "time series" aspect of the data.

Retweeters: the identities of the users who retweeted the target users. The insight is similar users get followed and retweeted by similar audience due to the homophily principle. This is the "network" aspect of the data.

Our technical contribution is to frame political leaning inference as a convex optimization problem that jointly maximizes tweet-retweet agreement with an error term, and user similarity agreement with a regularization term which is constructed to also account for heterogeneity in data. The result is an inference technique that is:

- Scalable: it does not require explicit knowledge of the network topology, and works within rate limits imposed by the Twitter API;
- Efficient: computationally efficient because it is formulated as a convex optimization problem, and data efficient because the time required to collect sufficient data to obtain good results is short; and
- Intuitive: the computed scores have a simple interpretation of "averaging," *i.e.,* a score is the average number of positive/negative tweets expressed when retweeting the target user. See Figure 2.1 for an illustration.

Using a set of 119 million tweets on the U.S. presidental election of 2012 collected over seven months, we extensively evaluate our method to show that it outperforms several standard algorithms and is robust with respect to variations to the algorithm.

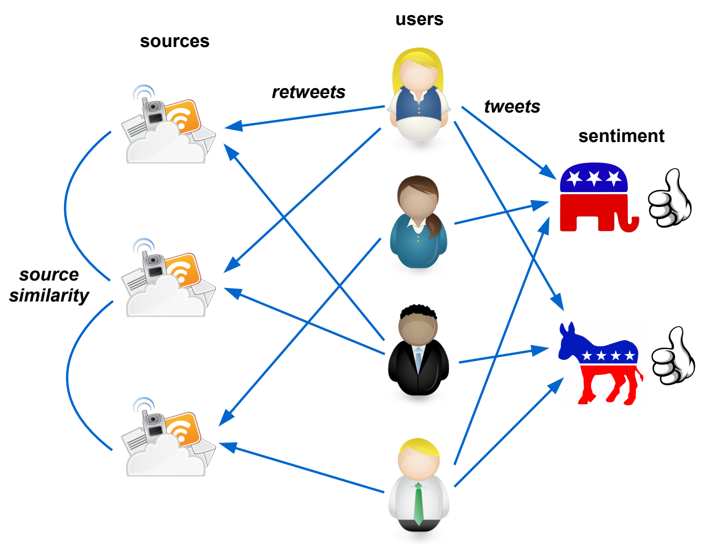


Figure 2.1: Incorporating tweets and retweets to quantify political leaning.

The second part of this chapter presents a numerical study on our collected tweets from the 2012 election, by first (a) quantifying the political leaning of 1,000 frequently-retweeted Twitter users, and then (b) using their political leaning, infer the leaning of 232,000 ordinary Twitter users. We make a number of findings:

- Parody Twitter accounts have a higher tendency to be liberal as compared to other account types. They also tend to be temporally less stable.
- Liberals dominate the population of less vocal Twitter users with less retweet activity, but for highly vocal populations, the liberal-convservative split is balanced. Partisanship also increases with vocalness of the population.
- Hashtag usage patterns change significantly as political events unfold.
- As an event is happening, the influx of Twitter users participating in the discussion makes the active population more liberal and less polarized.

The organization of the rest of this chapter is as follows. Section 2.2 reviews related work in studies of Twitter and quantifying political orientation in traditional and online social media. Section 2.3 details our inference technique by formulating political leaning inference as an optimization problem. Section 2.4 describes our dataset collected during the U.S. presidential election of 2012, which we use to derive ground truth for evaluation in Section 2.5. Then in Section 2.6 we perform a numerical study on the same dataset, studying the political leaning of Twitter users and hashtags, and how it changes with time. Section 2.7 concludes the chapter with future work.

2.2 Related Work

In political science, the ideal point estimation problem [80, 20] and its Bayesian extensions [38, 37] aim to estimate the political leaning of legislators from roll call data. This line of work assumes legislators to vote probabilistically according to their positions ("ideal points") in a latent space, and the latent positions are statistically inferred from observed data, *i.e.,* how they vote. The main difference between our work and this line of work is in the data: while legislators are characterized by their voting history, which can be considered as their explicit stances on various issues, we do not have access to comparably detailed data for most Twitter users.

A variety of methods have been proposed to quantify the extent of bias in traditional news media. Indirect methods involve linking media outlets to reference points with known political positions. For example, Lott and Hassett [61] linked the sentiment of newspaper headlines to economic indicators. Groseclose and Milyo [44] linked media outlets to Congress members by co-citation of think tanks, and then assigned political bias scores to media outlets based on the Americans for Democratic Action (ADA) scores of Congress members. Gentzkow and Shapiro [36] performed an automated analysis of text content in newspaper articles, and quantified media slant as the tendency of a newspaper to use phrases more commonly used by Republican or Democrat members of the Congress. In contrast, direct methods quantify media bias by analyzing news content for explicit (dis)approval of political parties and issues. Ho and Quinn [47] analyzed newspaper editorials on Supreme

Court cases to infer the political positions of major newspapers. Ansolabehere et al. [7] used 60 years of editorial election endorsements to identify a gradual shift in newspapers' political preferences with time.

Except for [36], the above studies require some form of manual coding and analysis, which is expensive and time-consuming. A more fundamental problem is data scarcity. Because the amount of data available for analysis is limited by how fast the media sources publish, researchers may need to aggregate data created over long periods of time, often years, to perform reliable analysis. Analyzing media sources through their OSN outlets offers many unprecedented opportunities with high volume data from interaction with their audience.

Political polarization has been studied in different types of online social media. Outside of Twitter, Adamic and Glance [2] analyzed link structure to uncover polarization of the political blogosphere. Zhou et al. [105] incorporated user voting data into random walk-based algorithms to classify users and news articles in a social news aggregator. Park et al. [78] inferred the political orientation of news stories by the sentiment of user comments in an online news portal. Weber et al. [98] assigned political leanings to search engine queries by linking them with political blogs. Regarding Twitter, political polarization was studied in [23]. Machine learning techniques have been proposed to classify Twitter users using *e.g.,* linguistic content, mention/retweet behavior and social network structure [11, 4, 79, 21]. Conover et al. [22] applied label propagation to a retweet graph for user classification, and found the approach to outperform tweet content-based machine learning methods.

Our problem of assigning meaningful political leaning scores to Twitter users is arguably more challenging than the above classification problem. There have already been several works on quantifying political leaning using the Twitter follower network. An et al. [6] and King et al. [50] applied multidimensional scaling on media sources with their pairwise distances computed from their mutual follower sets. Barberá [8] applied Bayesian ideal point estimation using following actions as observations. Golbeck and Hansen [41] proposed a graph-based method to propagate ADA scores of Congress members on Twitter to media sources through their followers. Weber et al. [99] quantified the political leaning of Twitter hashtags.

We argue that using retweet, rather than follower, data has its advantages. First, the huge sizes of most OSNs mean it is difficult for an average researcher to obtain an up-to-date snapshot of a network. The Twitter API prevents crawling the network beyond the one-hop neighborhood of a few thousand nodes.[1] On the other hand, our method requires only one connection to the real-time Twitter stream to collect retweets. Second, retweet data is more robust than follower data. Retweeting is an explicit act of approval, but following is not, *e.g.,* a user can follow two users with opposing stances to get a balanced view. Also, the follower graph is static in the sense that it does not capture real-time information flow and lacks fine-grained edge creation times.

Besides [22], retweet data have been applied in several recent works. Our retweet-based regularization is related to [30], which built a co-retweeted network for studying political polarization, and [97], which proposed a regularization framework using co-retweeting and co-retweeted information. Volkova et al. [93] built a series of Twitter social graphs to augment neighbors features (also studied in [4] but not on retweet graphs) to improve performance. Compared to the above, our work (a) does not directly use a co-retweeted network but adds a matrix scaling preprocessing step to account for heterogeneity in Twitter users' popularity, (b) introduces the tweet-retweet consistency condition, and (c) performs a longitudinal study on our dataset collected over seven months.

[1]As of the time of writing, each authenticated client can make 350 requests/hour, with each request returning at most 5,000 followers of a queried user. This is not to say scraping the complete network is impossible, but many tricks are needed [34].

2.3 Formulation

2.3.1 Motivation and Summary

To motivate our approach in using retweets for political leaning inference, we present two examples to highlight the existence of useful signals from retweet information.

From our dataset on the 2012 presidential election (see details in Section 2.4), we identify the Twitter accounts of two major media sources, one with liberal- and the other with conservative-leaning. In Figure 2.2 we plot their retweet popularity (their columns in matrix $\mathbf{A}$, see Section 2.3.2) during the 12 events in the dataset (see Table 2.1). We observe negative correlation between two sources' patterns of being retweeted, especially during events 6 and 7.[2] This can be explained by Democrat/Republican supporters enthusiastically retweeting Romney/Obama-bashing tweets published by the media outlets during the corresponding events.

This example provides two hints: (a) The number of retweets received by a retweet source during an event can be a signal of her political leaning. In particular, one would expect a politically inclined tweeter to receive more retweets during a favorable event. (b) The action of retweeting carries implicit sentiment of the retweeter. This is true even if the original tweet does not carry any sentiment itself. The intuition is that users tend to follow and retweet those who share similar political views, *e.g.,* a user is more likely to retweet a newspaper to which it subscribes than any random newspaper, a manifestation of the homophily principle.

Our second example shows the identities of retweeters are a signal of one's political leaning. In Figure 2.3, we plot a portion of matrix $\mathbf{S}$, which stores the cosine similarity of any two Twitter accounts based on the overlap of their sets of retweeters (see Section 2.3.4

[2]Event 6: a video was leaked with Romney criticizing "47% of Americans [who pay no income tax]" at a private fundraiser. Event 7: the first presidential debate, where Obama was criticized for his poor performance.

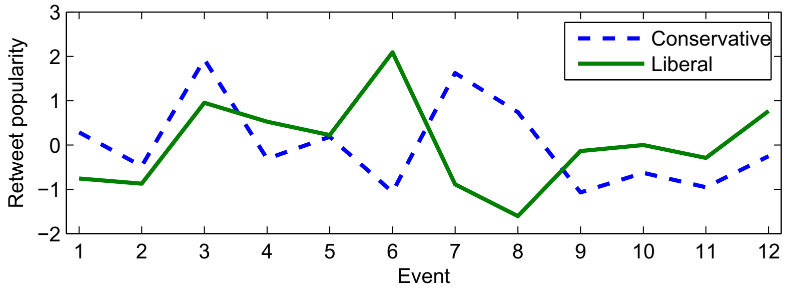


Figure 2.2: Negatively correlated retweet patterns of two media Twitter accounts with opposite political leaning.

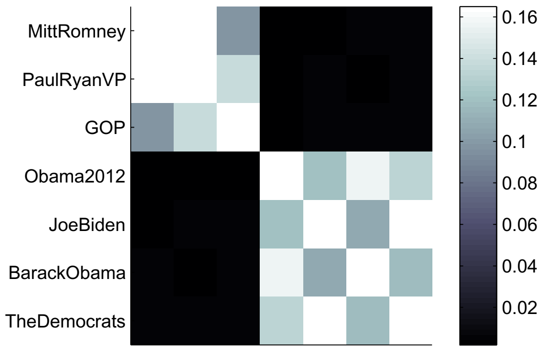


Figure 2.3: Similarity of U.S. presidential election candidates by co-retweeter information. Clear separation is observed, with low similarity (dark areas) between two accounts in different parties.

for details). By focusing on the election candidates[3] and official political party accounts, we see a clear separation of the two camps: two same-camp accounts have similarity that is at least 14 times of that between two different-camp accounts.

Given retweet and retweeter information are useful for inferring a Twitter account's political leaning, we formulate inference as a graph Laplacian-regularized least squares problem (Section 2.3.5) which consists of two steps. First, we assume that there is a large Twitter population that tweet and retweet at the same time, and the two forms of expressing political opinions are *consistent*. Then we frame political leaning estimation as a least squares (or linear inverse) problem in Section 2.3.3. Second, we add a regularization term

[3]Obama2012 is Barack Obama's official campaign account, and BarackObama is his personal account. There is no such distinction for Mitt Romney's Twitter account(s).

to the least squares problem to ensure similar Twitter users, *i.e.,* those having similar sets of audience who have retweeted them, have similar political leaning. We remark that naively building the regularization matrix results in poor performance. See Section 2.3.4 for how we carefully construct the matrix.

2.3.2 Definitions

Consider two political parties running for an election. During the election campaign there have been E *events* which attracted considerable attention in Twitter. We are interested in quantifying the *liberal-conservative*[4] political leaning of N prominent retweet *sources*, *e.g.,* media outlets' Twitter accounts and celebrities.

For event i, let U_i be the set of users who tweeted about the event, and T_{iu} be the set of tweets sent by user $u \in U_i$ about the event. Also define each tweet t to carry a score $s_t \in [-1, 1]$, such that it is 1 if the tweet shows full support for one candidate, or -1 if full support is for the other. Then for user u her approval score is

$$\sum_{t \in T_{iu}} \frac{s_t}{|T_{iu}|}.$$

Averaging over all users in U_i, the *average tweet leaning* y_i of event i is[5]

$$y_i = \frac{1}{|U_i|} \sum_{u \in U_i} \sum_{t \in T_{iu}} \frac{s_t}{|T_{iu}|}. \tag{2.1}$$

[4]In our analysis of the 2012 U.S. presidential election, it is the Republican and Democratic Parties competing, and we assume liberals to support the Democrats/Obama, and conservatives to support the Republicans/Romney.

[5]The specific forms of Eqs. (2.1) and (2.2) imply a user's contribution is limited in $[-1, 1]$ regardless of the number of tweets/retweets it sends. If we treat all tweets the same, *i.e.,* defining $y_i = \sum_{u \in U_i} \sum_{t \in T_{ui}} s_t / \sum_{u \in U_i} |T_{iu}|$, the performance degrades probably due to the skew from a few highly vocal users.

For source j, we quantify her political leaning as[6] $x_j \in \mathbb{R}$, interpreted as the *average* approval shown when someone retweets a tweet originating from j.

Now let V_i be the set of users who retweeted any one of the N sources during event i,[7] and $R_{uj}^{(i)}$ be the number of retweets sent by user u with the tweet originating from source j. Then the retweet approval score of user $u \in V_i$ is the average over all sources it has retweeted:

$$\sum_{j=1}^{N} \frac{R_{uj}^{(i)}}{\sum_{k=1}^{N} R_{uk}^{(i)}} x_j \tag{2.2}$$

and the *average retweet leaning* is the average over all u:

$$\begin{aligned} &\frac{1}{|V_i|} \sum_{u \in V_i} \sum_{j=1}^{N} \frac{R_{uj}^{(i)}}{\sum_{k=1}^{N} R_{uk}^{(i)}} x_j \\ =& \sum_{j=1}^{N} \left(\frac{1}{|V_i|} \sum_{u \in V_i} \frac{R_{uj}^{(i)}}{\sum_{k=1}^{N} R_{uk}^{(i)}} \right) x_j \\ =& \sum_{j=1}^{N} A_{ij} x_j, \end{aligned} \tag{2.3}$$

where A_{ij} is used to denote the inner summation term. The matrix $\mathbf{A}$ with elements A_{ij} can be interpreted as a Retweet matrix that captures the tweet-and-retweet response feature in Twitter.

2.3.3 An Ill-posed Linear Inverse Problem

The main premise of this chapter is that the behavior of tweeting and retweeting is consistent. Mathematically, we require the average tweet and retweet leanings per event to be similar:

$$y_i \approx \sum_{j=1}^{N} A_{ij} x_j, \qquad i = 1, \ldots, E. \tag{2.4}$$

[6]We do not constrain x_j to be bounded in $[-1, 1]$, although x_j and y_i should be on the same scale, and a properly designed algorithm should be able to recover it.

[7]In practice, we further restrict U_i and V_i to be the same user population by setting $U_i, V_i \leftarrow U_i \cap V_i$.

Our goal is to choose x_j's that minimize the error from the consistency equations Eq. (2.4), where the error measure is chosen to be the sum of squared differences $\sum_i(\sum_j A_{ij}x_j - y_i)^2$. Writing in matrix form, we are solving the unconstrained least squares problem

$$\underset{\mathbf{x}\in\mathbb{R}^N}{\text{minimize}} \qquad \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2. \tag{2.5}$$

We often have many more tweeters than events ($N = 1000$, $E = 12$ in Sections 2.5 and 2.6), then $N > E$ and the system of linear equations $\mathbf{A}\mathbf{x} = \mathbf{y}$ is underdetermined, which means there are infinitely many solutions $\mathbf{x}$ that can achieve the minimum possible error of 0 in Problem (2.5). Then the problem becomes an *ill-posed linear inverse problem* [12]. The challenge of solving ill-posed problems is in selecting a reasonable solution out of the infinite set of feasible solutions. For example, in our initial studies, the least-norm solution to (2.5) yielded unsatisfactory results.

2.3.4 Regularization

In statistical inference, solving ill-posed problems requires us to incorporate prior knowledge of the problem to rule out undesirable solutions. One such common approach is regularization, and we can change the objective function in Problem (2.5), $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$, to $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda f(\mathbf{x})$, where $\lambda > 0$ is a regularization parameter, and $f(\mathbf{x})$ quantifies the "fitness" of a solution such that undesirable solutions have higher $f(\mathbf{x})$ values. For example, Tikhonov regularization for least-squares uses $f(\mathbf{x}) = \|\mathbf{x}\|_2^2$ [12]. In this chapter, we propose a regularization term that favors political leaning assignments $\mathbf{x}$ with x_i being close to x_j if sources i and j have similar retweet responses.

Let W_{ij} be a regularization weight between sources i and j such that $W_{ij} \geq 0$ and $W_{ij} = W_{ji}$. Furthermore, let $\mathbf{W}$ be the weight matrix whose elements are W_{ij}. Then we set

$$f(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}(x_i - x_j)^2, \tag{2.6}$$

so that if W_{ij} is large (sources i and j are similar), then x_i should be close to x_j to minimize $W_{ij}(x_i - x_j)^2$.

Note that $f(\mathbf{x})$ can be rewritten in terms of a graph Laplacian. Let $\mathbf{D} = [D_{ij}]$ be defined as

$$D_{ij} = \begin{cases} \sum_{k=1}^{N} W_{ik} & i = j \\ 0 & \text{otherwise,} \end{cases}$$

and $\mathbf{L}$ be the graph Laplacian defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. Then it can be shown that

$$\sum_{i=1}^{N} \sum_{j=1}^{N} W_{ij}(x_i - x_j)^2 = 2\mathbf{x}^T\mathbf{L}\mathbf{x}. \tag{2.7}$$

Our $\mathbf{W}$ is constructed to account for the following.

Similarity based on co-retweeter sets. The first step in constructing $\mathbf{W}$ is to construct a similarity matrix $\mathbf{S} = [S_{ij}]$ that captures the similarity between two sources i and j by who retweeted them. Let $\mathcal{U}_i$ and $\mathcal{U}_j$ be the sets of users who have retweeted sources i and j respectively. We consider two standard similarity measures:

- Cosine similarity: $S_{ij} = \dfrac{|\mathcal{U}_i \cap \mathcal{U}_j|}{\sqrt{|\mathcal{U}_i| \cdot |\mathcal{U}_j|}}$, and
- Jaccard coefficient: $S_{ij} = \dfrac{|\mathcal{U}_i \cap \mathcal{U}_j|}{|\mathcal{U}_i \cup \mathcal{U}_j|}$.

Individualizing regularization strength. Regularization is more important for sources with insufficient information available from $\mathbf{A}$: if source i does not get retweeted often, her corresponding column sum, $\sum_j A_{ji}$, is small, and inferring her score x_i based on the error term $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2$ suffers from numerical stability issues.

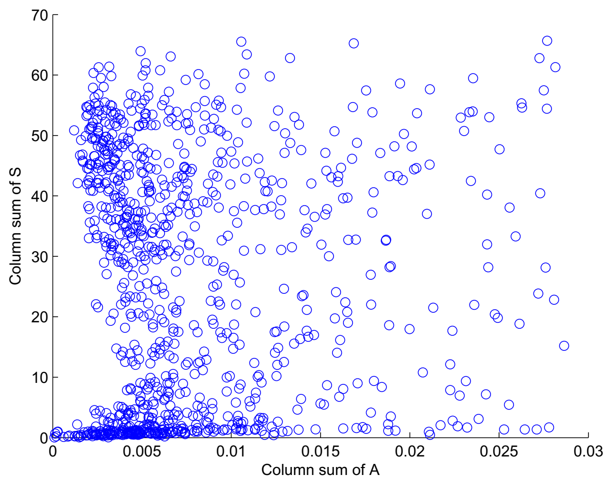


Figure 2.4: Relationship between retweet popularity (column sum on $\mathbf{A}$) and regularization strength (column sum on $\mathbf{S}$) of top 1,000 retweet sources in Section 2.4. A large number of sources are on the bottom left corner, meaning they are both unpopular and insufficiently regularized.

In practice, a source can simultaneously be scarcely retweeted and have low similarity with other sources, *i.e.,* source i has S_{ij} small for all other sources j. If $\mathbf{S}$ is directly used for regularization (*i.e.,* by setting $\mathbf{W} \leftarrow \mathbf{S}$), the source is insufficiently regularized and inference becomes unreliable. See Figure 2.4 for confirmation from real data.

Our solution to this problem is to apply matrix scaling [83] to $\mathbf{S}$. We compute $\mathbf{W}$ as the matrix closest to $\mathbf{S}$ (under an Kullback-Leibler divergence-like dissimilarity function) such that the row and column sums of $\mathbf{W}$ satisfy equality constraints:

$$\begin{array}{lll} \underset{\mathbf{W}\geq 0}{\text{minimize}} & \displaystyle\sum_{i,j=1}^{N} W_{ij} \log \frac{W_{ij}}{S_{ij}} & \\ \text{subject to} & W_{ij} = 0 & \text{for } (i,j) \text{ s.t. } S_{ij} = 0 \\ & \sum_{j=1}^{N} W_{ij} = u_i & i = 1, \ldots, N \\ & \sum_{j=1}^{N} W_{ji} = u_i & i = 1, \ldots, N, \end{array} \tag{2.8}$$

where $\mathbf{u} = \{u_i\}$ is a “regularization strength” parameter vector.

Problem (2.8) is solved by iterated rescaling of the rows and columns of $\mathbf{S}$ until convegence [83]. If the resultant $\mathbf{W}$ is asymmetric, we take the transformation $\mathbf{W} \leftarrow (\mathbf{W} + \mathbf{W}^T)/2$. It can be shown that the transformed $\mathbf{W}$ also satisfies the constraints in (2.8).

Incorporating prior knowledge. Prior knowledge can readily be incorporated into our inference technique as constraints of an optimization problem. In this chapter we consider two types of prior knowledge:

- Anchors: sources carrying an extreme leaning, *e.g.,* the election candidates themselves, can serve as anchors with fixed political leaning x_i. In the literature this idea has been used frequently [47, 6, 41].

- Score distribution: u_i can be interpreted as the strength of influence that source i exerts on sources similar to herself. Intuitively, anchors should exert higher influence, because we are more confident in their political leaning. Therefore, we set $\mathbf{u}$ as:

$$u_i = \begin{cases} \alpha_L & i \in \mathcal{A}_L \\ \alpha_C & i \in \mathcal{A}_C \\ 1 & \text{otherwise,} \end{cases} \tag{2.9}$$

  where α_L and α_C are tuning parameters, and $\mathcal{A}_L$ and $\mathcal{A}_C$ are the sets of liberal and conservative anchors respectively. Given there should exist some non-anchor sources with extreme leaning, we tune α_L and α_C as follows: (a) initialize $\alpha_L = \alpha_C = 1$, and then (b) iteratively increase α_L and α_C until the computed most extreme political leaning of a non-anchor source is within 90% of that of an anchor.

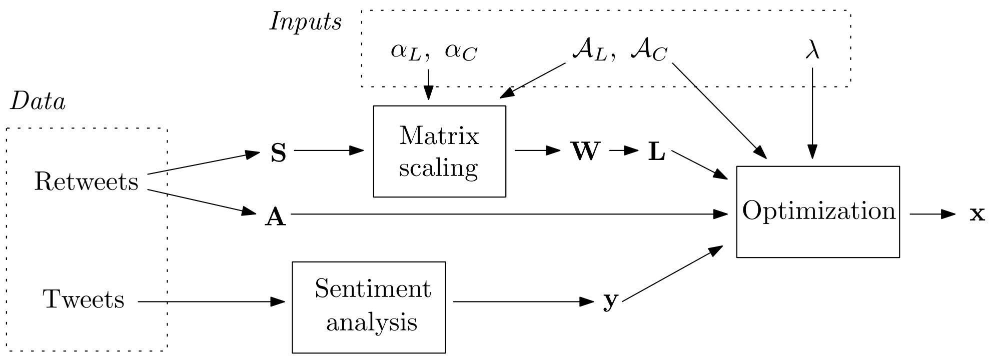


Figure 2.5: Flowchart of data processing and inference steps.

2.3.5 Optimization Problem

Combining Eqs. (2.5), (2.6) and our knowledge of anchors,[8] our final optimization formulation is:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^N}{\text{minimize}} \quad & \|\mathbf{A}\mathbf{x}-\mathbf{y}\|_2^2 + \lambda \mathbf{x}^T\mathbf{L}\mathbf{x} && \\ \text{subject to} \quad & x_i = -1 && i \in \mathcal{A}_L \\ & x_i = 1 && i \in \mathcal{A}_C, \end{aligned} \tag{2.10}$$

where λ is a tuning parameter. Figure 2.5 summarizes the data processing and inference steps in Section 2.3.

2.3.6 Extension: Sparsifying Graph Laplacian

Since most of the sources we consider are popular, most pairs of sources have at least one user who has retweeted both of them, and S is likely to be dense (in our dataset, 83% of its entries are nonzero). From a computational standpoint it is advantageous to sparsify the matrix, so we also evaluate our algorithm with an extra k-nearest-neighbor step, such that S_{ij} is kept only if j is a nearest neighbor of i, or vice versa. We are able to obtain good performance even when S is less than 10% sparse. See Section 2.6.4 for details.

[8]We define a liberal (conservative) anchor to have a negative (positive) score to be consistent with the convention that liberals (conservatives) are on the "left" ("right").

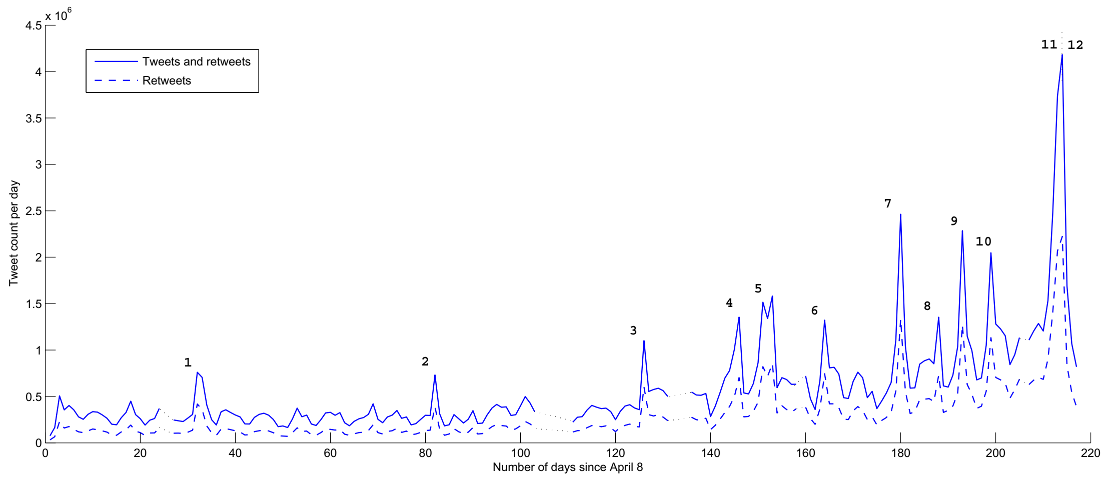


Figure 2.6: Number of tweets per day. Numbers on plot indicate events (see Table 2.1), and dotted lines indicate time periods when significant data were lost due to network outage (five instances).

2.4 Dataset

In this section we describe the collection and processing of our Twitter dataset of the U.S. presidential election of 2012. Our dataset was collected over a timespan of seven months, covering from the initial phases to the climax of the campaign.

Data Collection. From April 8 to November 10 2012, we used the Twitter streaming API[9] to collect 119 million tweets which contain any one of the following keyword phrases: "obama", "romney", "barack", "mitt", "paul ryan", "joe biden", "presidential", "gop", "dems", "republican" and "democrat" (string matching is case-insensitive).

Event Identification. By inspecting the time series of tweet counts in Figure 2.6, we manually identified 12 events as listed in Table 2.1. We defined the dates of an event as follows: the start date was identified based on our knowledge of the event, *e.g.,* the start time of a presidential debate, and the end date was defined as the day when the number of tweets reached a local minimum or dropped below that of the start date. After the events were identified, we extracted all tweets in the specified time interval[10] without additional filtering, assuming all tweets are relevant to the event and those outside are irrelevant.

[9]https://dev.twitter.com/streaming/overview

[10]For retweets, we only include those with the original tweet being created within the time interval.

Table 2.1: Summary of events identified in the dataset.

ID	Dates[11]	Description	# tweets (m)	# non-RT tweets (m)
1	May 9 - 12	Obama supports same-sex marriage	2.10	1.35
2	Jun 28 - 30	Supreme court upholds health care law	1.21	0.78
3	Aug 11 - 12	Paul Ryan selected as Republican VP candidate	1.62	0.96
4	Aug 28 - Sep 1	Republican National Convention	4.32	2.80
5	Sep 4 - 8	Democratic National Convention	5.81	3.61
6	Sep 18 - 22	Romney's 47 percent comment	4.10	2.55
7	Oct 4 - 5	First presidential debate	3.49	2.19
8	Oct 12 - 13	Vice presidential debate	1.92	1.19
9	Oct 17 - 19	Second presidential debate	4.38	2.67
10	Oct 23 - 26	Third presidential debate	5.62	3.35
11	Nov 4 - 6	Elections (before Obama projected to win)	7.50	4.40
12	Nov 7 - 9	Elections (after Obama projected to win)	6.86	4.43
Total			48.90	30.28

Extracting Tweet Sentiment. We applied SentiStrength [87], a lexicon-based sentiment analysis package, to extract the sentiment of tweets. We adjusted the provided lexicon by compiling a high-frequency tweet-word list per event, and then removing words[12] that we consider to not carry sentiment in the context of elections. Sentiment analysis was done as a ternary (positive, negative, neutral) classification.

For each tweet t, we set its score $s_t = -1$ if either (a) it mentions solely the Democrat camp (has "obama", "biden" etc. in text) and is classified to have positive sentiment, or (b) it mentions solely the Republican camp ("romney", "ryan" etc.) and has negative sentiment. We set $s_t = 1$ if the opposite criterion is satisfied. If both criteria are not satisfied, we set $s_t = 0$.

2.5 Evaluation

2.5.1 Ground Truth Construction

We compare the political leaning scores learnt by our technique with "ground truth" constructed by human evaluation. First, 100 sources are randomly selected from the 1,000 most popular retweet sources.[13] Then we ask 12 human judges with sufficient knowledge of American politics to classify each of the 100 sources as "L" (Liberal, if she is expected to vote Obama), "C" (Conservative, if expected to vote Romney), or "N" (Neutral, if she cannot decide), supposing each source is one voter who would vote in the presidential election. For each source, a judge is presented with (a) the source's user profile, including screen name, full name, self description, location and etc., and (b) ten random tweets published by the source. Given the set of labels, we compute our ground truth political leaning

[11] A time interval starts at 00:00:00 of start date, and ends at 23:59:59 of end date. Timezone used is UTC.

[12] They are "gay" (as in "gay marriage"), "foreign" ("foreign policy"), "repeal" ("repeal obamacare") and "battle*" ("battleship").

[13] Those who were retweeted the largest cumulative number of times during the 12 events.

scores $\{\tilde{x}_i\}$ as follows: for each label of L/N/C, assign a score of -1/0/$+1$, then the score of source i, call it $\tilde{x}_i$, as the average of her labels.

While there are many alternatives to defining and constructing ground truth, our choice is motivated by our implicit assumption of Twitter political leaning being the perceived leaning by a source's retweeters. If source i has $\tilde{x}_i$ with extreme values (-1 or $+1$), then it is unambiguously liberal/conservative, but if $\tilde{x}_i$ takes intermediate values, then some human judges may be confused with the source's leaning, and the general Twitter population is likely to have similar confusion, which suggests that the "correct" x_i should also take intermediate values. Defining $\{\tilde{x}_i\}$ this way also allows us to understand the usefulness of quantifying political leaning with a continuous score. Obviously, if all $\tilde{x}_i$ are either -1 or $+1$, a simple binary classification of the sources is enough, but as we see in Figure 2.7, $\{\tilde{x}_i\}$ is evenly spread across the range of allowed values $[-1, 1]$, so characterizing sources with simple binary, or even ternary, classification appears too coarse. To further support our claim, we also compute the inter-rater agreement of our manual labels as Fleiss' $\kappa = 0.430$ [31], a moderate level of agreement [54]. This suggests that while the labels are reliable, classifying sources is not trivial and a continuous political leaning score is useful.

Finally, we also manually classify each of the 1,000 most popular sources into four classes:

- Parody: role-playing and joke accounts created for entertainment purposes (example joke tweet: "I cooked Romney noodles Obama self," a pun on "I cooked ramen noodles all by myself")
- Political: candidates of the current election and accounts of political organizations
- Media: outlets for distributing information in an objective manner, setting aside media bias issues
- Others: personal accounts, including those of celebrities, pundits, reporters, bloggers and politicians (excluding election candidates).

2.5.2 Performance Metrics

The quality of political leaning scores is measured under two criteria.

Classification. One should be able to directly infer the liberal/conservative stance of a source i from her sign of x_i, *i.e.,* it is liberal if $x_i < 0$, or conservative if $x_i > 0$. Taking $\{\tilde{x}_i\}$ as ground truth, we say source i is correctly classified if the signs of x_i and $\tilde{x}_i$ agree.[14] Classification performance is measured using the standard metrics of accuracy, precision, recall and F1 score.

Rank correlation. The set of scores $\{x_i\}$ induce a ranking of the sources by their political leaning. This ranking should be close to that induced by the ground truth scores $\{\tilde{x}_i\}$. We measure this aspect of performance using Kendall's τ, which varies from -1 (perfect disagreement) to 1 (perfect agreement).

2.5.3 Results

We solve Problem (2.10) with $\mathcal{A}_L = \{\text{Obama2012}\}$ and $\mathcal{A}_C = \{\text{MittRomney}\}$ and compare the results with those from several baselines:

- PCA: we run Principal Components Analysis on $\mathbf{A}$ with each column being the feature vector of a source, with or without the columns being standardized, and take the first component as $\{x_i\}$. This is the baseline when we use only the information from $\mathbf{A}$ (retweet counts).

- Eigenvector: we compute the second largest eigenvector of $\mathbf{L}$, with $\mathbf{L}$ becoming computed from $\mathbf{S}$ being either the cosine or Jaccard matrix. This is a technique commonly seen in spectral graph partitioning [29], and is the standard approach when only the information from $\mathbf{S}$ (retweeters) is available. Note that the $\mathbf{x}$ computed this way is equivalent to solving the optimization problem: minimize $\mathbf{x}^T\mathbf{L}\mathbf{x},$ subject to $\|\mathbf{x}\|_2 = 1,\ \mathbf{x}^T\mathbf{1} = 0.$

[14]In the unlikely case of $\tilde{x}_i = 0$ (2 out of 100 test cases), we require $x_i = 0$ for correct classification.

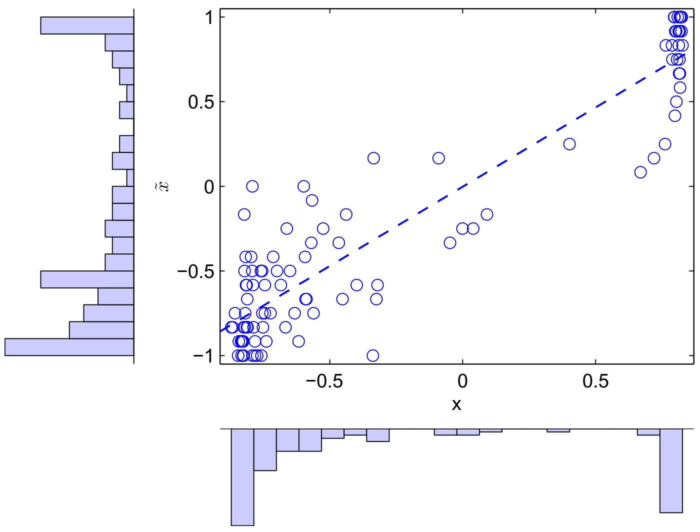


Figure 2.7: Relationship of ground truth $\{\tilde{x}_i\}$ and our computed scores $\{x_i\}$ on the 100 sources with manual labels, together with their marginal distributions. Our method is able to recover both the correct classifications (datapoints in bottom-left and upper-right quadrants) and rankings for most sources.

- Sentiment analysis: we take x_i as the average sentiment of the tweets published by source i, using the same methdology in computing $\mathbf{y}$ [87]. This is the baseline when only tweets are used.

Table 2.2 reports the evaluation results. Our algorithm, in combining information from $\mathbf{A}$, $\mathbf{S}$ and $\mathbf{y}$, performs significantly better than all baselines in terms of Kendall's τ, F1 score and accuracy. We also observe that if no matrix scaling is applied in constructing $\mathbf{W}$, the algorithm tends to assign all $\{x_i\}$ (except those of anchors) to the same sign, resulting in poor classification performance. Somewhat surprisingly, sentiment analysis performs the best among all the baselines, in contrast to what one would expect based on related work [69, 22, 79]. In the remaining of this chapter, we focus on the political leaning scores computed using cosine similarity.

Figure 2.7 shows the correspondence between $\{x_i\}$ and $\{\tilde{x}_i\}$. The two sets of scores exhibit similar rankings of sources and a bimodal score distribution. The scores due to our algorithm are slightly more polarized.

2.6 Numerical Study

2.6.1 Quantifying Prominent Retweet Sources

We study the properties of the political leaning of the 1,000 most popular retweet sources. Similar to that in Figure 2.7, the score histogram on the full set (Figure 2.8) has a bimodal distribution. We note that by incorporating retweeter information, our algorithm is able to correctly position "difficult" sources that were highly retweeted during events unfavorable to the candidate they support, *e.g.,* JoeBiden, CBSNews and all accounts related to Big Bird. We also find that WSJ is assigned a slightly liberal score. This is consistent with findings reported in prior studies [44, 61] explained by the separation between *WSJ*'s news and editorial sections.

We also study the score distributions of sources grouped by account type. Figure 2.9 shows noticeable differences among the different groups. Parody sources are skewed towards the liberal side. Political sources are strongly polarized with no sources having neutral (close to zero) scores. Media sources are less polarized with a more even spread of scores. Sources in the "Others" class have a score distribution close to that of political sources, but also includes a few neutral scores, which can be attributed to celebrities with no clear political stance.

2.6.2 Quantifying Ordinary Twitter Users

Given the political leaning of 1,000 retweet sources, we can use them to infer the political leaning of ordinary Twitter users who have retweeted the sources. We consider the set of users seen in our dataset who have retweeted the sources at least ten times, including retweets made during non-event time periods. In total there are 232,000 such users. We caution this set of users is not necessarily representative of the general Twitter population, or even the full population of our dataset (9.92 million users in total), but we believe it is

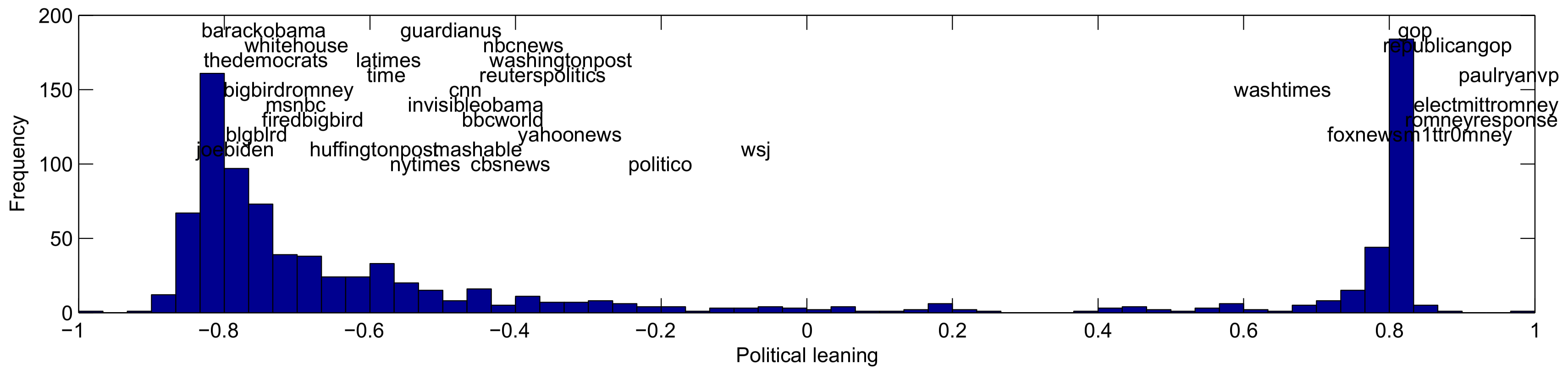


Figure 2.8: Distribution of political leaning scores of top 1,000 retweet sources with positions of some example sources.

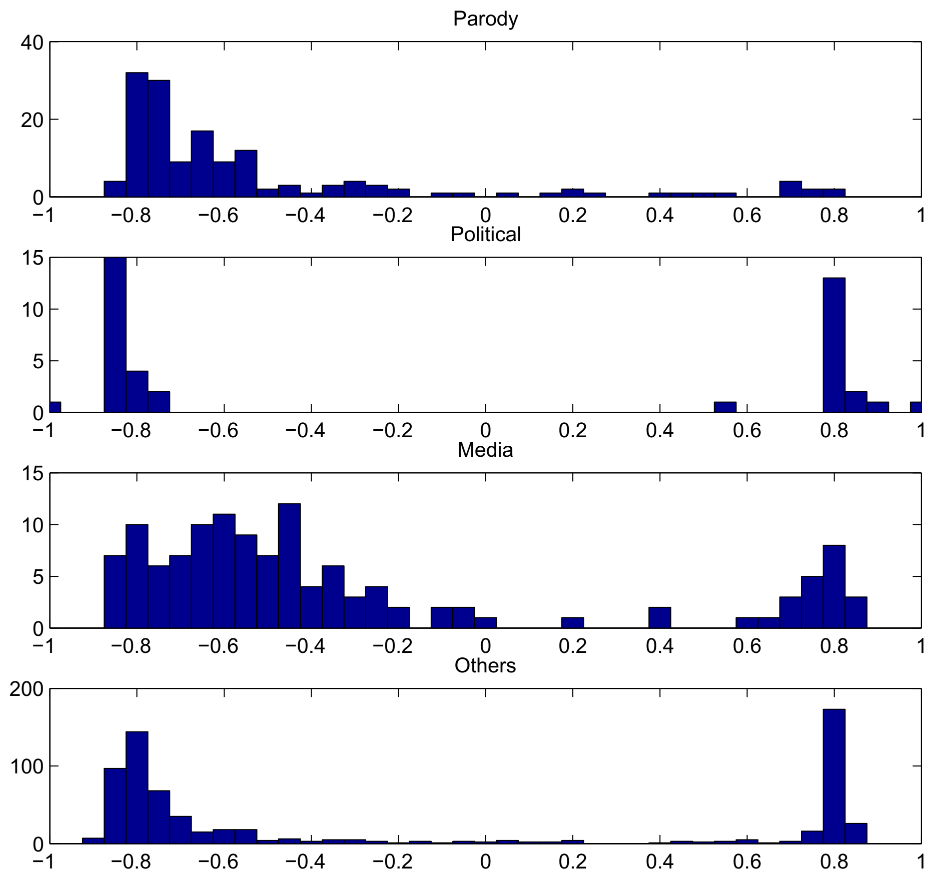


Figure 2.9: Distribution of political leaning scores grouped by Twitter account type. Parody/comedy accounts are skewed towards the liberal side. Political accounts are more polarized (no scores close to zero) than other accounts.

possible to "propagate" score estimates from these 232,000 users to everyone else, which remains as future work.

For user u, we infer her political leaning x_u as

$$x_u = \frac{\sum_{i=1}^{N} R_{ui} x_i}{\sum_{i=1}^{N} R_{ui}}, \tag{2.11}$$

where R_{ui} is the number of times user u retweeted source i and x_i is source i's political leaning.

Figure 2.10 shows the kernel density estimate of the political leaning scores of the 232,000 ordinary users, compared with that of the set of 1,000 sources. These users have a

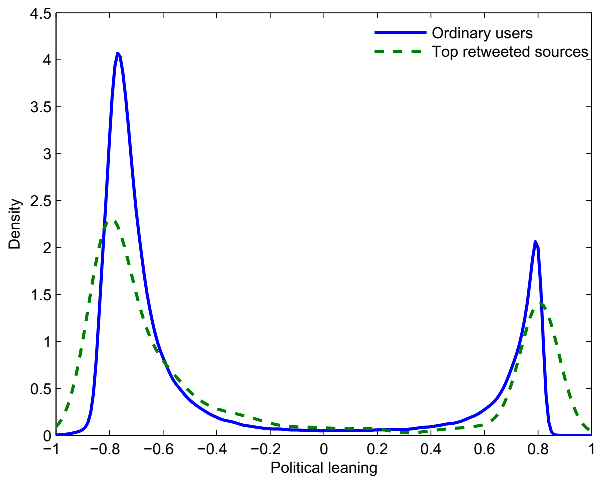


Figure 2.10: Kernel density estimates of political leaning of top 1,000 retweet sources and 232,000 ordinary Twitter users.

slightly less polarized distribution (density function is closer to zero), but are more skewed towards the liberal side (72.5% have $x_u < 0$, compared to 69.5% for retweet sources).

Measuring polarization. Using the learnt political leaning scores, we aim to quantify the political polarization of a population, but for this to be possible, we first need a polarization measure. Let us consider one user u. A natural measure P_u for her polarization can be defined as how far her political leaning is away from neutral: $P_u = |x_u - 0| = |x_u|$. Then for a population $\mathcal{U}$, its polarization measure can be taken as the average of all P_u for $u \in \mathcal{U}$. However, such a definition does not account for class imbalance (liberals outnumbering conservatives in our case), so we take a class-balanced definition instead:

$$P_{\mathcal{U}} = \frac{1}{|\mathcal{U}_+|} \sum_{u \in \mathcal{U}_+} x_u + \frac{1}{|\mathcal{U}_-|} \sum_{u \in \mathcal{U}_-} |x_u|, \tag{2.12}$$

where $\mathcal{U}_+ = \{u \mid u \in \mathcal{U},\ x_u > 0\}$ and $\mathcal{U}_- = \{u \mid u \in \mathcal{U},\ x_u < 0\}$.

Now we put the 232,000 ordinary users into ten percentile bins, such that the first bin contains the lowest 10% of users according to their retweet activity (number of retweets made, including retweets of non-top 1,000 sources), then the next bin contains the next lowest 10%, and so on. Figure 2.11 shows the plot of two skew measures: the polarization

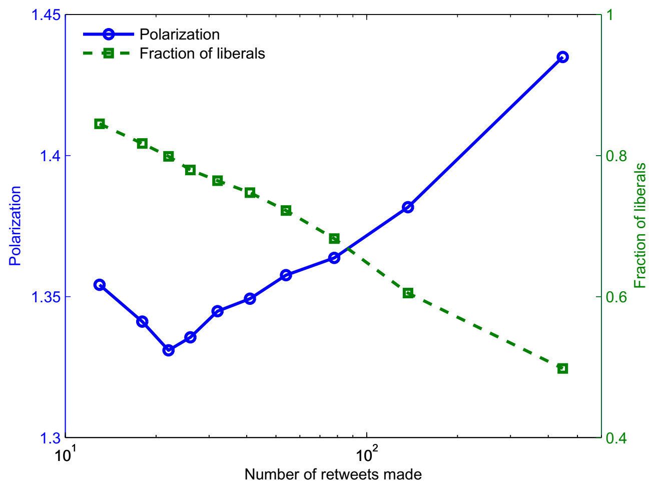


Figure 2.11: Skew measures of ordinary users binned by 10%-tiles of rewteet activity. Polarization and liberal-conservative balance increase for more active populations.

measure as defined in Eq. (2.12), and the fraction of liberals $|\mathcal{U}_+|/(|\mathcal{U}_+ + \mathcal{U}_-|)$. We observe that liberals dominate (>80%) the population of low activity users, but as retweet activity increases, the liberal-conservative split becomes more balanced (roughly 50% in the last bin). This suggests that most Twitter users (80% of them make less than 100 retweets on the election) tend to be liberals, but the most vocal population (those making 100 to 33,000 retweets) consists of users who are more politically opiniated such that they spend more effort to promote their causes in social media. This is supported by the plot of the polarization measure, which increases with retweet activity.

2.6.3 Temporal Dynamics

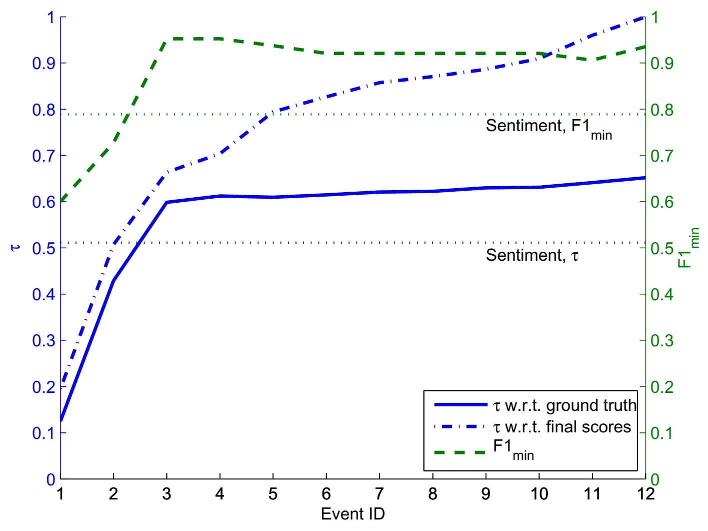
With the large amount of data available from Twitter one can perform fine-grained temporal analysis using data from a relatively short timespan. In this section, we study how many events are necessary to obtain sufficiently good performance, and then present two examples in applying our methodology to social media monitoring.

Stability. Here we quantify the political leaning of the 1,000 sources studied in Section 2.6.1 but with varying amounts of information. We start by running our inference technique using data from only the first event, then we use events 1 to 2, and so on. Then each source has a sequence of 12 political leaning scores, and we evaluate the quality of these scores compared to ground truth. Figure 2.12(a) shows that three events, or 10% of the data, are enough for achieving performance close to that from using the full dataset. From the description of the events in Table 2.1, the first two events are skewed towards the liberal side, and it is not clear how conservative users react to them (either object strongly or remain silent). With the third event added, we have a more balanced set of data for reliable inference.

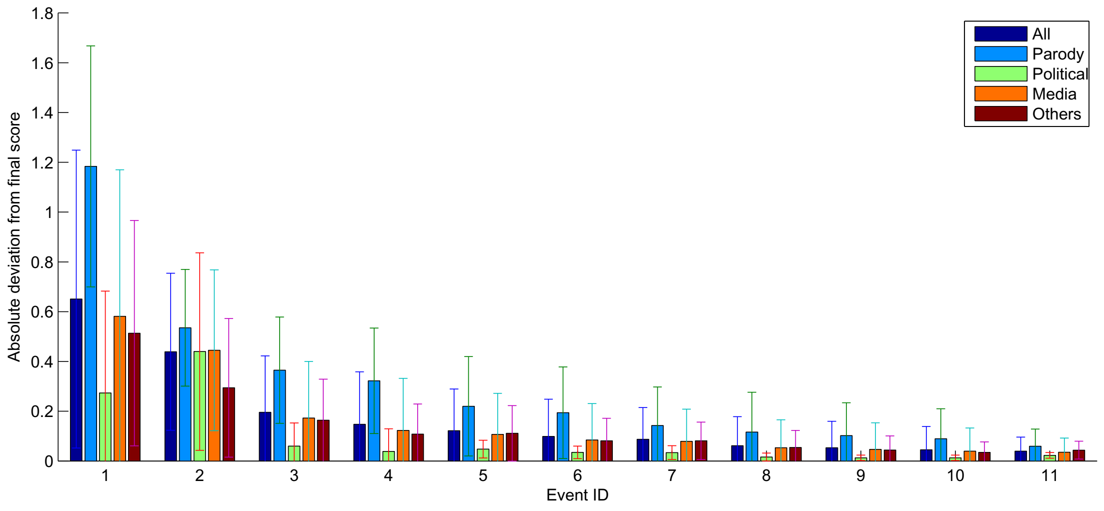
Next we look at the stability of scores per source across the 12 events. Figure 2.12(b) plots the mean deviation of a source's score per event from her final score. Using the classification from Section 2.5, we see stability varies with the type of a source. Political sources are the most stable with their score deviations having decreased quickly after three events, consistent with our intuition that they are the easy cases with the least ambiguity in political leaning. On the other hand, parody accounts are the least stable. Their patterns of being retweeted are the least stable with large fluctuations in retweet counts across different events, resulting in less reliable inference. Also, users do not retweet parody sources by how agreeable, but rather by how funny they are, so even retweeter information on these sources is less stable across different events.

Trending hashtags. The political leaning of hashtags [99] can be quantified by how they are being used by Twitter users. Here we consider a simple way to estimate it using the political leaning scores of the top retweet sources. For each hashtag, we compute its political leaning as the average of all sources (within the top 1000 retweet sources) that have used it at least once in their published tweets. Moreover, we perform this computation per event (excluding previous events) to obtain a sequence of hashtag political leaning scores. This allows us to track trending hashtags as events unfold.

Tables 2.3 and 2.4 are the lists of the most liberal or conservative hashtags that have been used by at least ten sources in each of the events. Besides the static hashtags indi-



(a) Performance measures. $\mathrm{F1}_{\mathrm{min}}$ is the minimum of F1 scores of the C and L classes.



(b) Deviation from final result, with grouping by account type. Error bars indicate one standard deviation from the group average.

Figure 2.12: Stability of results with varying number of events: good performance is achieved with only the first three events, and parody accounts are the least stable.

cating users' political affiliation such as #TCOT (top conservatives on twitter) and #p2b (progressive 2.0 bloggers), we discover hashtags being created in response to events (#ssm (same-sex marriage) in event 1, #MyFirstTime in event 10 for a suggestive Obama ad). Moreover, there are hashtags showing opposite opinions on the same issue (#aca (affordable care act) vs #ObamaTax in event 2, #WrongAgainRyan vs #BidenUnhinged in event 8), and many instances of sarcasm (#StopIt and #YouPeople in event 6, #notoptimal in event 9 for Obama saying the government's response was "not optimal" during the Benghazi attack) and accusations (#MSM (main stream media) for liberal media bias).

In the latter half of the dataset, liberals tend to focus on accusing the other side of lying during the debate (#LyinRyan, #SketchyDeal, #MittLies, #AdmitItMitt), while conservatives tend to focus on the Benghazi attack (#BenghaziGate, #notoptimal, #7HoursOfHell, #Benghazi). Finally, we note a change in hashtag usage before and after the election outcome came out (from #WhyImNotVotingForRomney to #FourMoreYears).

Tracking temporal variation in polarization. We begin with this question: is the Twitter population more (or less) polarized during an event?

Without analyzing the data, the answer is not clear because it is influenced by two factors with opposite effects:

- An event draws attention from less vocal users who are likely to have weak political leaning. These users join the discussion because everyone talks about it.
- On the other hand, the fact that a usually silent user joining the discussion may indicate he/she is strongly opinionated about the topic of discussion.

To answer the question, we compute the polarization measure in Eq. (2.12) for each day in our dataset, with population $\mathcal{U}$ taken as the set of users (out of the 232,000 users from Section 2.6.2) who have tweeted or retweeted on that day. For comparison purposes we also compute the fraction of liberals per day.

The results are shown in Figure 2.13. First, liberals outnumber conservatives in every single day, regardless of whether it is an event day or not. Second, there is a slight increasing trend in polarization over the course of events, which corresponds to discussions become more heated as the election campaign progresses. Third, the fraction of liberals is significantly higher at the onset (first day) of an event. This is observed in 10 out of the 12 events. In contrast, polarization drops and reaches a local minimum during an event (10/12 events), and the level is lower than nearby non-event days. It appears the influx of users during an event drives polarization of the Twitter population down, because these extra users tend to have weaker political leaning.

We also contrast the changes in liberal skew and polarization right before and after the election outcome came out: at the climax of the election, the liberal-conservative share in active users is relatively balanced because both sides want to promote their candidate of support; at the same time the population is more polarized. After Obama is projected to win, there is a jump in liberal fraction with conservatives leaving the discussion, and polarization plummets, probably with the departure of strong-leaning conservatives.

2.6.4 Sensitivity to Parameter Variation

Our algorithm is robust to variation in input parameters λ, α_C and α_L. Starting from the chosen $(\lambda, \alpha_C, \alpha_L)$ tuple from the previous section, we fix two parameters and vary the remaining one. Figures 2.14(a) and 2.14(c) show the resultant performance does not vary significantly over a wide range of parameter values.

We also consider a simple approach to sparsify the graph Laplacian. Given $\mathbf{S}$, we preprocess it with a k-nearest-neighbor step before passing it to matrix scaling: for each source i we find the k other sources $\{j\}$ with highest S_{ij}, then we keep only the S_{ij} entries (not set to zero) for j being a neighbor of i or i being a neighbor of j. We vary the value of k from 50 to 1000 to vary the sparsity of $\mathbf{S}$ (and $\mathbf{L}$). Figure 2.14(b) shows even when k is small ($k = 50$ results in a sparsity of 8.7%), the performance is still very good.[15]

2.7 Conclusions

Scoring individuals by their political leaning is a fundamental research question in computational political science. From roll calls to newspapers, and then to blogs and microblogs, researchers have been exploring ways to use bigger and bigger data for political leaning

[15]Reducing k further introduces convergence problems in the matrix scaling step.

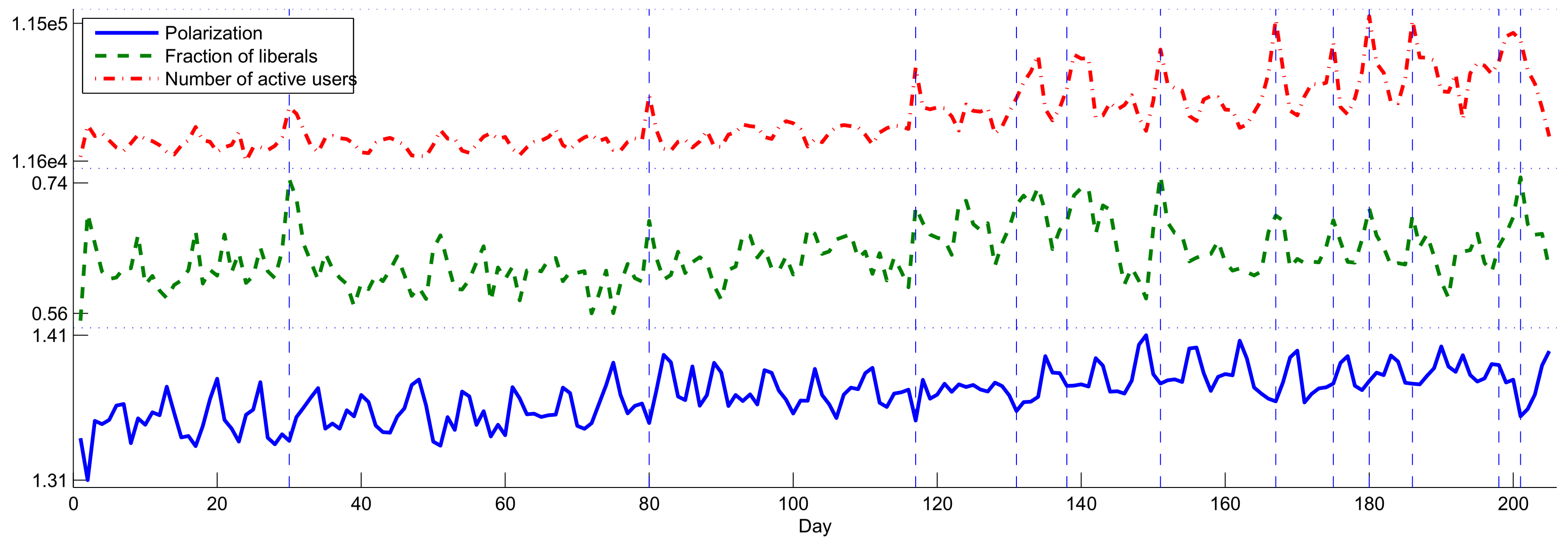


Figure 2.13: Skew measures of ordinary users across time. Days with significant data loss during data collection are omitted. Liberal fraction and polarization are negatively correlated during events.

inference. But new challenges arise in how one can exploit the structure of the data, especially because bigger data are often are noisier and sparser.

In this chapter, we propose to leverage two properties of Twitter data: (a) Twitter users tend to tweet and retweet consistently, and (b) similar Twitter users tend to be retweeted by similar sets of audience, to develop a convex optimization-based political leaning inference technique that is simple, efficient and intuitive. Our method is evaluated on a large dataset of 119 million U.S. election-related tweets collected over seven months, and using manually constructed ground truth labels, we found it to outperform many baseline algorithms. With its reliability validated, we applied it to quantify a set of prominent retweet sources, and then propagated their political leaning to a larger set of ordinary Twitter users and hashtags. The temporal dynamics of political leaning and polarization were also studied.

We believe this is the first systematic step in this type of approaches in quantifying Twitter users' behavior. The Retweet matrix and retweet average scores can be used to develop new models and algorithms to analyze more complex tweet-and-retweet features. Our optimization framework can readily be adapted to account to incorporate other types of information. For example:

- The $\mathbf{y}$ vector does not need to be computed from sentiment analysis of tweets. It can be built from exogenous information that can be used to match the opinions of the Twitter retweet population, such as poll results.

- The $\mathbf{A}$ matrix is currently built with each row corresponding to one event, but the correspondence can be made with respect to other groupings of tweets and retweets, such as by economic/diplomatic/religious issues.

- The $\mathbf{W}$ matrix can be constructed from other types of network data (*e.g.,* the network of mentions) or similarity measures.

- Our methodology is also applicable to other OSNs with retweet-like endorsement mechanisms, such as Facebook and YouTube with "like" functionality.

Table 2.2: Performance of our algorithm compared to several baselines. Best two results (almost always due to our method) are highlighted in bold.

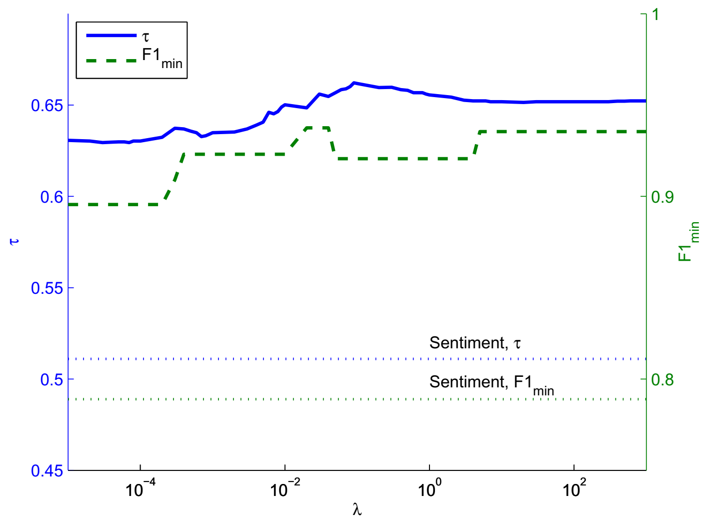
Algorithm	Kendall's τ	Precision, L	Recall, L	Precision, C	Recall, C	F1 score, L	F1 score, C	Accuracy
Ours, cosine matrix	**0.652**	**0.942**	**0.970**	**0.935**	**0.935**	**0.955**	**0.935**	**0.94**
Ours, Jaccard matrix	**0.654**	**0.940**	0.940	**0.879**	**0.935**	**0.940**	**0.906**	**0.92**
Ours, cosine w/o scaling	0.649	0.670	**1**	0	0	0.802	0	0.67
Ours, Jaccard w/o scaling	0.641	0	0	0.31	**1**	0	0.473	0.31
PCA	0.002	0.663	0.791	0.300	0.194	0.721	0.235	0.59
PCA, standardized columns	0.011	0.750	0.224	0.325	0.839	0.345	0.468	0.41
Eigenvector, cosine matrix	0.370	0.838	0.851	0.688	0.710	0.844	0.698	0.79
Eigenvector, Jaccard matrix	0.292	0.864	0.761	0.610	0.806	0.810	0.694	0.76
Sentiment analysis	0.511	0.926	0.746	0.700	0.903	0.826	0.789	0.78

Table 2.3: Hashtags with highest liberal political leaning per event.

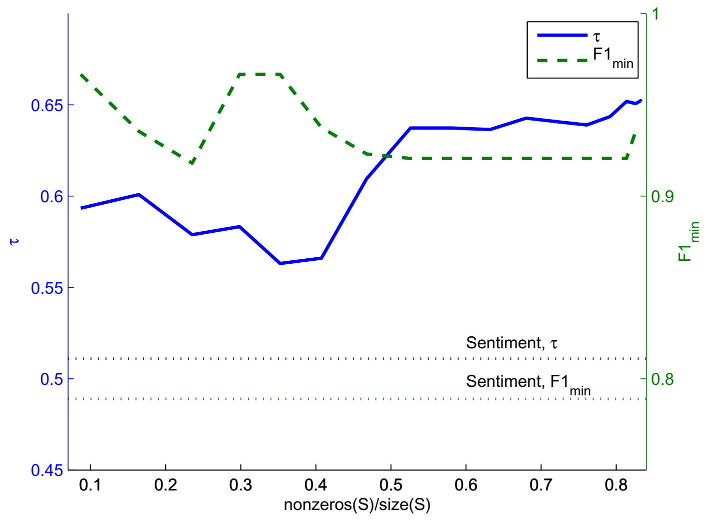
Event	Top five liberal hashtags				
1	#MarriageEquality	#equality	#marriageequality	#edshow	#ssm
2	#aca	#p2b	#ACA	#Obama2012	#edshow
3	#p2b	#Medicare	#topprog	#edshow	#Obama2012
4	#p2b	#YouPeople	#Current2012	#msnbc2012	#uppers
5	#ACA	#PaulRyan	#LyinRyan	#nerdland	#DavidGregorysToughQuestions
6	#LyinRyan	#ObamaBiden2012	#StopIt	#p2b	#YouPeople
7	#47Percent	#47percent	#Forward	#TeamObama	#topprog
8	#WrongAgainRyan	#p2b	#Bain	#Sensata	#LyinRyan
9	#SketchyDeal	#Bain	#MittLies	#p2b	#BinderFullofWomen
10	#StrongerWithObama	#RomneyWrong	#ObamaBiden2012	#RomneyNotReady	#AdmitItMitt
11	#p2b	#uppers	#msnbc2012	#ObamaBiden2012	#WhyImNotVotingForRomney
12	#OBAMA	#obama2012	#msnbc2012	#Boehner	#FourMoreYears

Table 2.4: Hashtags with highest conservative political leaning per event.

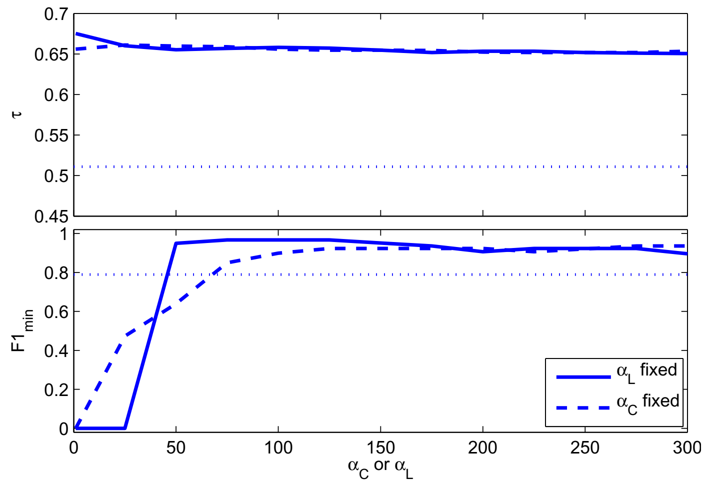
Event	Top five conservative hashtags				
1	#LNYHBT	#lnyhbt	#twisters	#sgp	#ocra
2	#LNYHBT	#ObamaTax	#Obamatax	#ocra	#lnyhbt
3	#LNYHBT	#sgp	#TCOT	#Mitt2012	#ocra
4	#LNYHBT	#twisters	#ocra	#sgp	#TCOT
5	#Mitt2012	#LNYHBT	#ocra	#twisters	#military
6	#ObamaIsntWorking	#WAR	#Resist44	#2016	#MSM
7	#EmptyChair	#ForwardNotBarack	#sgp	#resist44	#TCOT
8	#BenghaziGate	#sgp	#CNBC2012	#BidenUnhinged	#lnyhbt
9	#CantAfford4more	#BenghaziGate	#notoptimal	#sgp	#Missouri
10	#BenghaziGate	#Benghazigate	#sgp	#caring	#MyFirstTime
11	#military	#7HoursOfHell	#Twibbon	#LNYHBT	#BENGHAZI
12	#sgp	#ocra	#Benghazi	#WAR	#lnyhbt



(a) Performance w.r.t. overall regularization strength.



(b) Performance w.r.t. sparsity of graph Laplacian.



(c) Performance w.r.t. anchors' regularization strength.

Figure 2.14: Our inference technique is robust with respect to parameter variations.

Chapter 3

Sparse Optimization for Stock Market Nowcasting from Newspapers

3.1 Introduction

A main goal in algorithmic trading in financial markets is to predict if a stock's price will go up or down at the end of the current trading day as the algorithms continuously receive new market information. One variant of the question is to construct effective prediction algorithms based on news articles. Understanding this question is important for two reasons: (1) A better solution helps us gain more insights on how financial markets react to news, which is a long-lasting question in finance [28, 15, 86]. (2) It presents a unique challenge in machine learning, where time series analysis meets text information retrieval. While there have been quite extensive studies on stock price prediction based on news, much less work can be found on *simultaneously* leveraging the correlations (1) among stock prices, (2) among news articles, and (3) between stock prices and news articles [71].

In this chapter, we revisit the stock price prediction problem based on news articles. On each trading day, we feed a prediction algorithm all the articles that appeared on that day's *Wall Street Journal* (*WSJ*) (which becomes available before the market opens), then we ask the algorithm to predict whether each stock in S&P 500, DJIA and Nasdaq will move up or down. Our algorithm's accuracy is approximately 55% (based on $\geq 100,000$ test cases). This shall be contrasted with "textbook models" for time series that have less than 51.5% prediction accuracy (see Section 3.5). We also remark that we require the algorithm to predict *all* the stocks of interest while most of the stocks are *not mentioned at all* in a typical *WSJ* newspaper. On the other hand, most of the existing news-based prediction algorithms can predict only stocks that are explicitly mentioned in the news. Finally, when we use this algorithm to construct a portfolio, we find our portfolio yields substantially better return and Sharpe ratio compared to a number of standard indices (see Figures 3.5(a) and 3.5(b)).

Performance surprises. We were quite surprised by the performance of our algorithm for the following reasons.

(1) Our algorithm runs on minimal data. Here, we use only daily open and close prices and *WSJ* news articles. It is clear that all serious traders on Wall Street have access to both pieces of information, and much more. By the efficient market hypothesis, it should be difficult to find arbitrage based on our dataset (in fact, the efficient market hypothesis explains why the accuracy rates of "textbook models" are below 51.5%). Thus, we were intrigued by the performance of our algorithm. It also appears that the market might not be as efficient as one would imagine.

(2) Our model is quite natural but it appears to have never been studied before. As we shall see in the forthcoming sections, our model is rather natural for capturing the correlation between stock price movements and news articles. While the news-based stock price prediction problem has been extensively studied [71], we have not seen a model similar to ours in existing literature. Section 3.7 also compares our model with a number of important existing approaches.

(3) Our algorithm is robust. Many articles in *WSJ* are on events that happened a day before (instead of reporting new stories developed overnight). Intuitively, the market shall

be able to absorb information immediately and thus "old news" should be excluded from a prediction algorithm. Our algorithm does not attempt to filter out any news since deciding the freshness of a news article appears to be remarkably difficult, and yet even when a large portion of the input is not news, our algorithm can still make profitable predictions.

Our approach. We now outline our solution. We build a unified latent factor model to explain stock price movements and news. Our model originates from straightforward ideas in time series analysis and information retrieval: when we study co-movements of multiple stock prices, we notice that the price movements can be embedded into a low dimensional space. The low dimensional space can be "extracted" using standard techniques such as Singular Value Decomposition. On the other hand, when we analyze texts in news articles, it is also standard to embed each article into latent spaces using techniques such as probabilistic latent semantic analysis or latent Dirichlet allocation [76].

Our crucial observation here is that stock prices and financial news should "share" the same latent space. For example, the coordinates of the space can represent stocks' and news articles' weights on different industry sectors (*e.g.,* technology, energy) and/or topics (*e.g.,* social, political). Then if a fresh news article is about "crude oil," we should see a larger fluctuation in the prices of stocks with higher weight in the "energy sector" direction.

Thus, our approach results in a much simpler and more interpretable model. But even in this simplified model, we face a severe overfitting problem: we use daily trading data over six years. Thus, there are only in total approximately 1500 trading days. On the other hand, we need to predict about 500 stocks. When the dimension of our latent space is only ten, we already have 5000 parameters. In this setting, appropriate regularization is needed.

Finally, our inference problem involves non-convex optimization. We use Alternating Direction Method of Multipliers (ADMM) [14] to solve the problem. Here the variables in the ADMM solution are matrices and thus we need a more general version of ADMM. While the generalized analysis is quite straightforward, it does not seem to have appeared in the literature. This analysis for generalized ADMM could be of independent interest.

In summary,

1. We propose a unified and natural model to leverage the correlation between stock price movements and news articles. This model allows us to predict the prices of all the stocks of interest even when most of them are not mentioned in the news.

2. We design appropriate regularization mechanisms to address the overfitting problem and develop a generalized ADMM algorithm for inference.

3. We carry out extensive backtesting experiments to validate the efficacy of our algorithm. We also compare our algorithm with a number of widely used models and observe substantially improved performance.

3.2 Notation and Preliminaries

Let there be n stocks, m words, and $s+1$ days (indexed as $t = 0, 1, \ldots, s$). We then define the following variables:

- x_{it}: closing price of stock i on day t,
- y_{jt}: intensity of word j on day t,
- $r_{it} = \log\Big(\dfrac{x_{it}}{x_{i,t-1}}\Big)$: log return of stock i on day $t \geq 1$.

The stock market prediction problem using newspaper text is formulated as follows:

> *For given day t, use both historical data $[r_{it'}]$, $[y_{jt'}]$ (for $t' < t$) and this morning's newspaper $[y_{jt}]$ to predict $[r_{it}]$, for all i and j.*[1]

In this chapter we compute y_{jt} as the z-score on the number of newspaper articles that contain word j relative to the article counts in previous days. To reduce noise, an

[1] $[x_{it}]$ is recoverable from $[r_{it}]$ given $[x_{i,t-1}]$ is known.

extra thresholding step is included to remove values that are negative or below 3 standard deviations.

Dataset. We use stock data in a period of almost six years and newspaper text from *WSJ*. We identified 553 stocks that were traded from 1/1/2008 to 9/30/2013 and listed in at least one of the S&P 500, DJIA, or Nasdaq stock indices during that period. We then downloaded opening and closing prices[2] of the stocks from CRSP.[3] Additional stock information was downloaded from Compustat. For text data, we downloaded the full text of all articles published in the print version of *WSJ* in the same period. We computed the document counts per day that mention the top 1000 words of highest frequency and the company names of the 553 stocks. After applying a stoplist and removing company names with too few mentions, we obtained a list of 1354 words.

3.3 Sparse Matrix Factorization Model

Equipped by recent advances in matrix factorization techniques for collaborative filtering [52], we propose a unified framework that incorporates (a) historical stock prices, (b) correlation among different stocks and (c) newspaper content to predict stock price movement. Underlying our technique is a latent factor model that characterizes a stock (*e.g.,* it is an energy stock) and the *average* investor mood of a day (*e.g.,* economic growth in America becomes more robust and thus the demand for energy is projected to increase), and that the price of a stock on a certain day is a function of the latent features of the stock and the investor mood of that day.

More specifically, we let stocks and trading days share a d-dimensional latent factor space, so that stock i is described by a nonnegative feature vector $u_i \in \mathbb{R}_+^d$ and trading day t is described by another feature vector $v_t \in \mathbb{R}^d$. Now if we assume u_i and v_t are known,

[2]We adjust prices for stock splits, but do not account for dividends in our evaluation.

[3]CRSP, Center for Research in Security Prices. Graduate School of Business, The University of Chicago 2014. Used with permission. All rights reserved. www.crsp.uchicago.edu

we model day t's log return, $\hat{r}_{it}$, as the inner product of the feature vectors $\hat{r}_{it} = u_i^T v_t + \epsilon$, where ϵ is a noise term. In the current setting we can only infer v_t by that morning's newspaper articles as described by $y_t = [y_{jt}] \in \mathbb{R}_+^m$, so naturally we may assume a linear transformation $W \in \mathbb{R}^{d\times m}$ to map y_t to v_t, *i.e.,* we have $v_t = Wy_t$. Then log return prediction can be expressed as

$$\hat{r}_{it} = u_i^T W y_t. \tag{3.1}$$

Our goal is to learn the feature vectors u_i and mapping W using historical data from s days. Writing in matrix form: let $R = [r_{it}] \in \mathbb{R}^{n\times s}$, $U = [u_1 \cdots u_n]^T \in \mathbb{R}^{n\times d}$, $Y = [y_1 \cdots y_s] \in \mathbb{R}^{m\times s}$, we aim to solve

$$\underset{U\geq 0,\ W}{\text{minimize}} \qquad \frac{1}{2}\|R - UWY\|_F^2. \tag{3.2}$$

Remark Here, the rows of U are the latent variables for the stocks while the columns of WY are latent variables for the news. We allow one of U and WY to be negative to reflect the fact that news can carry negative sentiment while we force the other one to be non-negative to control the complexity of the model. Also, the model becomes less interpretable when both U and WY can be negative.

Note our formulation is similar to the standard matrix factorization problem except we add the matrix Y. Once we have solved for U and W we can predict price $\hat{x}_{it}$ for day t by $\hat{x}_{it} = x_{i,t-1}\exp(\hat{r}_{it}) = x_{i,t-1}\exp\big(u_i^T W y_t\big)$ given previous day's price $x_{i,t-1}$ and the corresponding morning's newspaper word vector y_t.

Overfitting. We now address the overfitting problem. Here, we introduce the following two additonal requirements to our model:

1. We require the model to be able to produce a predicted log returns matrix $\hat{R} = [\hat{r}_{it}]$ that is close to R and be of low rank at the same time, and

2. be sparse because we expect many words to be irrelevant to stock market prediction (a feature selection problem) and each selected word to be associated with few factors.

The first requirement is satisfied if we set $d \ll s$. The second requirement motivates us to introduce a sparse group lasso [32] regularization term in our optimization formulation. More specifically, feature selection means we want only a small number of columns of W (each column corresponds to one word) to be nonzero, and this can be induced by introducing the regularization term $\lambda \sum_{j=1}^{m} \|W_j\|_2$, where W_j denotes the j-th column of W and λ is a regularization parameter. On the other hand, each word being associated with few factors means that for each relevant word, we want its columns to be sparse itself. This can be induced by introducing the regularization term $\mu \sum_{j=1}^{n} \|W_j\|_1 = \mu\|W\|_1$, where μ is another regularization parameter, and $\|W\|_1$ is taken elementwise.

Thus our optimization problem becomes

$$\begin{aligned} \underset{U,\, W}{\text{minimize}} \quad & \frac{1}{2}\|R - UWY\|_F^2 + \lambda \sum_{j=1}^{m} \|W_j\|_2 + \mu\|W\|_1 \\ \text{subject to} \quad & U \geq 0. \end{aligned} \tag{3.3}$$

We remark we also have examined other regularization approaches, *e.g.,* ℓ_2 regularization and plain group lasso, but they do not outperform baseline algorithms.

3.4 Optimization Algorithm

Our problem is biconvex, *i.e.,* convex in either U or W but not jointly. It has been observed such problems can be effectively solved by ADMM [104]. Here, we study how such techniques can be applied in our setting. We rewrite the optimization problem by replacing the nonnegative constraint with an indicator function and introducing auxiliary variables A and

B:

$$\begin{aligned} &\underset{A,\, B,\, U,\, W}{\text{minimize}} \quad \frac{1}{2}\|R - ABY\|_F^2 + \lambda \sum_{j=1}^{m} \|W_j\|_2 + \mu\|W\|_1 + I_+(U) \\ &\text{subject to} \quad A = U,\ B = W, \end{aligned} \tag{3.4}$$

where $I_+(U) = 0$ if $U \geq 0$, and $I_+(U) = \infty$ otherwise.

We introduce Lagrange multipliers C and D and formulate the augmented Lagrangian of the problem:

$$\begin{aligned} L_\rho(A, B, U, W, C, D) = &\frac{1}{2}\|R - ABY\|_F^2 + \lambda \sum_{j=1}^{m} \|W_j\|_2 + \mu\|W\|_1 + I_+(U) \\ &+ \text{tr}\big(C^T(A - U)\big) + \text{tr}\big(D^T(B - W)\big) \\ &+ \frac{\rho}{2}\|A - U\|_F^2 + \frac{\rho}{2}\|B - W\|_F^2. \end{aligned} \tag{3.5}$$

Using ADMM, we iteratively update the variables A, B, U, W, C, D, such that in each iteration (denote G_+ as the updated value of some variable G):

$$\begin{aligned} A_+ &= \text{argmin}_A\ L_\rho(A, B, U, W, C, D) \\ B_+ &= \text{argmin}_B\ L_\rho(A_+, B, U, W, C, D) \\ U_+ &= \text{argmin}_U\ L_\rho(A_+, B_+, U, W, C, D) \\ W_+ &= \text{argmin}_W\ L_\rho(A_+, B_+, U_+, W, C, D) \\ C_+ &= C + \rho(A_+ - U_+) \\ D_+ &= D + \rho(B_+ - W_+). \end{aligned}$$

Algorithm 1 lists the steps involved in ADMM optimization. In the remainder of this section we derive the update equations.

Algorithm 1 ADMM optimization for (3.3).

```
Input: $R, Y, \lambda, \mu, \rho$
Output: $U, W$
  Initialize $A, B, C, D$
  repeat
    $A \leftarrow (RY^TB^T - C + \rho U)(BYY^TB^T + \rho I)^{-1}$
    $B \leftarrow$ solution to
        $\left(\frac{1}{\rho}A^TA\right)B(YY^T) + B = \frac{1}{\rho}(A^TRY^T - D) + W$
    $U \leftarrow \left(A + \frac{1}{\rho}C\right)^+$
    for $j = 1$ to $m$ do
      $W_j \leftarrow \left(\frac{\|w\|_2 - \lambda}{\rho\|w\|_2}\right)^+ w$, where
            $w = \rho\,\mathrm{sgn}(v)(|v| - \mu/\rho)^+$, $v = B_j + D_j/\rho$
    end for
    $C \leftarrow C + \rho(A - U)$
    $D \leftarrow D + \rho(B - W)$
  until convergence or max iterations reached
```

We first make use of the fact $\|G\|_F^2 = \mathrm{tr}(G^TG)$ and express the augmented Lagrangian in terms of matrix traces:

$$
\begin{aligned}
L_\rho = {} & \frac{1}{2}\mathrm{tr}\big((R - ABY)^T(R - ABY)\big) + \lambda\sum_{j=1}^{m}\|W_j\|_2 + \mu\|W\|_1 \\
& + I_+(U) + \mathrm{tr}\big(C^T(A - U)\big) + \mathrm{tr}\big(D^T(B - W)\big) \\
& + \frac{\rho}{2}\mathrm{tr}\big((A - U)^T(A - U)\big) + \frac{\rho}{2}\mathrm{tr}\big((B - W)^T(B - W)\big),
\end{aligned}
$$

then we expand and take derivatives as follows.

Updating A. We have

$$
\begin{aligned}
\frac{\partial L_\rho}{\partial A} &= \frac{1}{2}\frac{\partial \mathrm{tr}(Y^TB^TA^TABY)}{\partial A} - \frac{1}{2}\cdot 2\frac{\partial \mathrm{tr}(R^TABY)}{\partial A} \\
&\quad + \frac{\partial \mathrm{tr}(C^TA)}{\partial A} + \frac{\rho}{2}\frac{\partial \mathrm{tr}(A^TA)}{\partial A} - \frac{\rho}{2}\cdot 2\frac{\partial \mathrm{tr}(U^TA)}{\partial A} \\
&= ABYY^TB^T - RY^TB^T + C + \rho A - \rho U.
\end{aligned}
$$

By setting the derivative to 0, the optimal A^* satisfies

$$A^* = (RY^TB^T - C + \rho U)(BYY^TB^T + \rho I)^{-1}.$$

Updating B. Similarly,

$$\begin{aligned}\frac{\partial L_\rho}{\partial B} &= \frac{1}{2}\frac{\partial \text{tr}(Y^TB^TA^TABY)}{\partial B} - \frac{1}{2}\cdot 2\frac{\partial \text{tr}(R^TABY)}{\partial B} \\ &\quad + \frac{\partial \text{tr}(D^TB)}{\partial B} + \frac{\rho}{2}\frac{\partial \text{tr}(B^TB)}{\partial B} - \frac{\rho}{2}\cdot 2\frac{\partial \text{tr}(W^TB)}{\partial B},\end{aligned}$$

then setting 0 and rearranging, we have

$$\Big(\frac{1}{\rho}A^TA\Big)B^*(YY^T) + B^* = \frac{1}{\rho}(A^TRY^T - D) + W.$$

Hence B^* can be computed by solving the above Sylvester matrix equation of the form $AXB + X = C$. See Appendix 3.A for details.

Updating U. Note that

$$\begin{aligned}U_+ &= \text{argmin}_U \; I_+(U) - \text{tr}(C^TU) + \frac{\rho}{2}\|A - U\|_F^2 \\ &= \text{argmin}_U \; I_+(U) + \frac{\rho}{2}\left\|\Big(A + \frac{1}{\rho}C\Big) - U\right\|_F^2 \\ &= \Big(A + \frac{1}{\rho}C\Big)^+,\end{aligned}$$

with the minimization in step 2 being equivalent to taking the Euclidean projection onto the convex set of nonnegative matrices [14].

Updating W. W is chosen to minimize

$$\lambda\sum_{j=1}^{m}\|W_j\|_2 + \mu\|W\|_1 - \text{tr}(D^TW) + \frac{\rho}{2}\|B - W\|_F^2.$$

Note that this optimization problem can be solved for each of the m columns of W separately:

$$\begin{aligned} W_j^* &= \text{argmin}_u \ \lambda\|u\|_2 + \mu\|u\|_1 - D_j^T u + \frac{\rho}{2}\|B_j - u\|_2^2 \\ &= \text{argmin}_u \ \lambda\|u\|_2 + \mu\|u\|_1 + \frac{\rho}{2}\left\|u - \left(B_j + \frac{D_j}{\rho}\right)\right\|_2^2, \end{aligned} \tag{3.6}$$

We can obtain a closed-form solution by studying the subdifferential of the above expression.

Lemma 3.4.1. *Let $F(u) = \lambda\|u\|_2 + \mu\|u\|_1 + \rho/2\|u - v\|_2^2$. Then the minimizer u^* of $F(u)$ is*

$$u^* = \left(\frac{\|w\|_2 - \lambda}{\rho\|w\|_2}\right)^+ w,$$

where $w = [w_i]$ is defined as $w_i = \rho\, sgn(v_i)(|v_i| - \mu/\rho)^+$.

This result was given in a slightly different form in [84]. A more detailed proof is given in Appendix 3.B for completeness.

Then applying Lemma 3.4.1 to (3.6), we obtain

$$W_j^* = \left(\frac{\|w\|_2 - \lambda}{\rho\|w\|_2}\right)^+ w,$$

where $w = \rho \ \text{sgn}(v)\Big(|v| - \frac{\mu}{\rho}\Big)^+$ and $v = B_j + \frac{D_j}{\rho}$.

3.5 Evaluation

We split our dataset into a training set using years 2008 to 2011 (1008 trading days), a validation set using 2012 (250 trading days), and a test set using the first three quarters of 2013 (188 trading days). In the following, we report on the results of both 2012 (validation set) and 2013 (test set), because a comparison between the two years reveals interesting insights. We fix $d = 10$, *i.e.,* ten latent factors, in our evaluation.

3.5.1 Price Direction Prediction

First we focus on the task of using one morning's newspaper text to predict the closing price of a stock on the same day. Because our ultimate goal is to devise a profitable stock trading strategy, our performance metric is the accuracy in predicting the up/down direction of price movement,[4] averaged across all stocks and all days in the evaluation period.

We compare our method with baseline models outlined below. The first two baselines are trivial models but in practice it is observed that they yield small least square prediction errors.

- **Previous** X: we assume stock prices are flat, *i.e.,* we always predict today's closing prices being the same as yesterday's closing prices.
- **Previous** R: we assume returns R are flat, *i.e.,* today's returns are the same as the previous day's returns. Note we can readily convert between predicted prices $\hat{X}$ and predicted returns $\hat{R}$.
- **Autoregressive (AR) models** on historical prices ("AR on X") and returns ("AR on R"): we varied the order of the AR models and found them to give best performance at order 10, *i.e.,* a prediction depends on previous ten day's prices/returns.
- **Regress on** X/R: we also regress on previous day's prices/returns on *all* stocks to predict a stock's price/return to capture the correlation between different stocks.

Table 3.1 summarizes our evaluation results in this section. Our method performs better than all baselines in terms of directional accuracy. Although the improvements look modest

[4]An up/down prediction $\hat{R}_{it}$ for stock i on day t is correct if either $\hat{R}_{it} > 0$ and $R_{it} > 0$, or $\hat{R}_{it} \leq 0$ and $R_{it} \leq 0$. This definition is motivated by considering $\hat{R}_{it} > 0$ as a "buy" signal (which is correct if following it results in a gain, *i.e.,* $R_{it} > 0$), and $\hat{R}_{it} \leq 0$ as a "don't buy" signal (note that $R_{it} = 0$ still implies a loss due to transaction costs).

Table 3.1: Results of price prediction.

Model	Accuracy, 2012 (%)	Accuracy, 2013 (%)
Ours	**53.9**	**55.7**
Previous X	49.9	46.9
Previous R	49.9	49.1
AR(10) on X	50.4	49.5
AR(10) on R	50.6	50.9
Regress on X	50.2	51.4
Regress on R	48.9	50.8

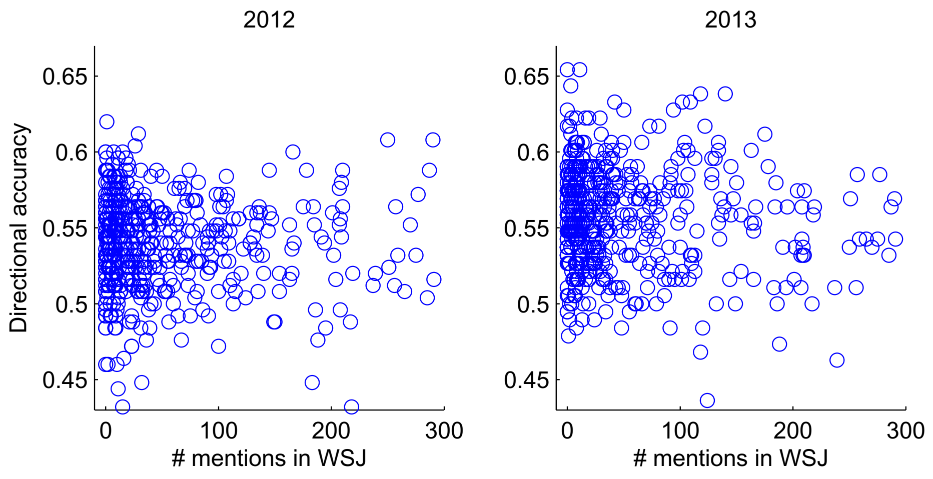


Figure 3.1: Scatterplot of directional accuracy of individual stocks.

by only a few percent, we will see in the next section that they result in significant financial gains. Note that our accuracy results should not be directly compared to other results in existing work because the evaluation environments are different. Factors that affect evaluation results include timespan of evaluation (years vs weeks), size of data (*WSJ* vs multiple sources), frequency of prediction (daily vs intraday) and target to predict (all stocks in a fixed set vs news-covered stocks or stock indices).

Stocks not mentioned in *WSJ*. The performance of our algorithm does not degrade over stocks that are *rarely mentioned* in *WSJ*: Figure 1 presents a scatter plot on stocks' directional accuracy against their number of mentions in *WSJ*. One can see that positive correlations between accuracy and frequencies of mention do not exist. To our knowledge, none of the existing prediction algorithms have this property.

3.5.2 Backtesting of Trading Strategies

We next evaluate trading strategies based on our prediction algorithm. We consider the following simplistic trading strategy: at the morning of each day we predict the closing prices of all stocks, and use our current capital to buy all stocks with an "up" prediction, such that all bought stocks have the same amount of investment. Stocks are bought *at the opening prices* of the day. At the end of the day we sell all we have to obtain the capital for the next morning.[5]

We compare our method with four sets of baselines:

- Three major stock indices (S&P 500, DJIA and Nasdaq),
- Trading based on the baseline price prediction models,[6]
- Uniform portfolios, *i.e.,* spend an equal amount of capital on each stock, and
- Minimum variance portfolios (MVPs) [67] with expected returns at 95th percentile of historical stock returns.

For the latter two we consider the strategies of buy and hold (BAH), *i.e.,* buy stocks on the first day of the evaluation period and sell them only on the last day, and constant rebalancing (CBAL), *i.e.,* for a given portfolio (weighting) of stocks we maintain the stock weights by selling and rebuying on each day. Following [35] (and see the discussion therein for the choices of the metrics), we use five performance metrics: cumulative return, worst day return $= \min_t (X_{it} - X_{i,t-1})/X_{i,t-1}$, maximum drawdown, Conditional Value at Risk (CVaR) at 5% level, and daily Sharpe ratio with S&P 500 returns as reference.

Tables 3.2 and 3.3 summarizes our evaluation. In both years our strategy generates significantly higher returns than all baselines. As for the other performance metrics, our strategy dominates all baselines in 2013, and in 2012, our strategy's metrics are either the best or close to the best results.

[5]Incorporating shorting and transaction costs is future work.

[6]We omit the results from the "previous X" baseline because it produces "don't buy" signals for all stocks on all days.

Table 3.2: Results of simulated trading in 2012.

Model	Return	Worst day	Max drawdown	CVaR	Sharpe ratio
Ours	**1.21**	-0.0291	**0.0606**	**-0.0126**	0.0313
S&P 500	1.13	-0.0246	0.0993	-0.0171	–
DJIA	1.07	**-0.0236**	0.0887	-0.0159	-0.109
Nasdaq	1.16	-0.0282	0.120	-0.0197	**0.0320**
U-BAH	1.13	-0.0307	0.134	-0.0204	0.00290
U-CBAL	1.13	-0.0278	0.0869	-0.0178	-0.00360
MVP-BAH	1.06	-0.0607	0.148	-0.0227	-0.0322
MVP-CBAL	1.09	-0.0275	0.115	-0.0172	-0.0182
Previous R	1.15	-0.0299	0.0826	-0.0176	0.00693
AR(10) on X	1.12	-0.0317	0.103	-0.0198	-0.00721
AR(10) on R	1.13	-0.0297	0.0910	-0.0181	-0.00214
Regress on X	1.13	-0.0342	0.124	-0.0209	0.000472
Regress on R	1.12	-0.0293	0.0842	-0.0181	-0.00803

Table 3.3: Results of simulated trading in 2013.

Model	Return	Worst day	Max drawdown	CVaR	Sharpe ratio
Ours	**1.56**	**-0.0170**	**0.0243**	**-0.0108**	**0.148**
S&P 500	1.18	-0.0250	0.0576	-0.0170	–
DJIA	1.15	-0.0234	0.0563	-0.0151	-0.0561
Nasdaq	1.25	-0.0238	0.0518	-0.0179	0.117
U-BAH	1.22	-0.0296	0.0647	-0.0196	0.0784
U-CBAL	1.14	-0.0254	0.0480	-0.0169	-0.0453
MVP-BAH	1.24	-0.0329	0.0691	-0.0207	0.0447
MVP-CBAL	1.10	-0.0193	0.0683	-0.0154	-0.0531
Previous R	1.21	-0.0256	0.0457	-0.0160	0.0263
AR(10) on X	1.12	-0.0288	0.0505	-0.0180	-0.0560
AR(10) on R	1.13	-0.0257	0.0485	-0.0174	-0.0527
Regress on X	1.15	-0.0253	0.0506	-0.0170	-0.0302
Regress on R	1.18	-0.0264	0.0554	-0.0176	-0.00351

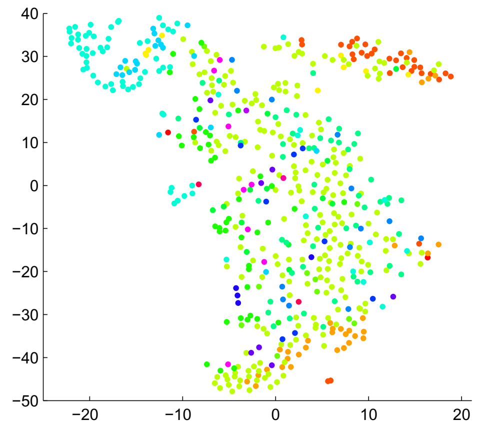
Table 3.4: Closest stocks. Stocks are represented by ticker symbols.

Target	10 closest stocks
BAC	XL STT KEY C WFC FII CME BK STI CMA
HD	BBBY LOW TJX BMS VMC ROST TGT AN NKE JCP
GOOG	CELG QCOM ORCL ALXN CHKP DTV CA FLIR ATVI ECL

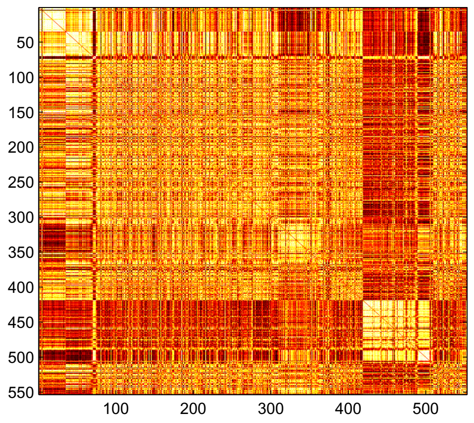
3.6 Interpretation of the Models and Results.

Block structure of U. Given we have learnt U with each row being the feature vector of a stock, we study whether these vectors give meaningful interpretations by applying t-SNE [91] to map our high-dimensional (10D) stock feature vectors on a low-dimensional (2D) space. Intuitively, similar stocks should be close together in the 2D space, and by “similar” we mean stocks being in the same (or similar) sectors according to North American Industry Classification System (NAICS). Figure 3.2(a) confirms our supposition by having stocks of the same color, *i.e.,* in the same sector, being close to each other. Another way to test U is to compute the stock adjacency matrix. Figure 3.2(b) shows the result with a noticeable block diagonal structure, which independently confirms our claim that the learnt U is meaningful.

Furthermore, we show the learnt U also captures connections between stocks that are not captured by NAICS. Table 3.4 shows the 10 closest stocks to Bank of America (BAC), Home Depot (HD) and Google (GOOG) according to U. For BAC, all close stocks are in finance or insurance, *e.g.,* Citigroup (C) and Wells Fargo (WFC), and can readily be deduced from NAICS. However, the stocks closest to HD include both retailers, *e.g.,* Lowe’s (LOW) and Target (TGT), and related non-retailers, including Bemis Company (BMS, specializes in flexible packaging) and Vulcan Materials (VMC, specializes in construction materials). Similarly, the case of GOOG reveals its connections to biotechnology stocks including Celgene Corporation (CELG) and Alexion Pharmaceuticals (ALXN). Similar results have also been reported by [26].



(a) t-SNE on rows of U. Each stock is a datapoint and each color represents an NAICS industry sector.



(b) Adjacency matrix of rows of U by correlation distance. Stock IDs are sorted by sectors.

Figure 3.2: Visualizing stocks.

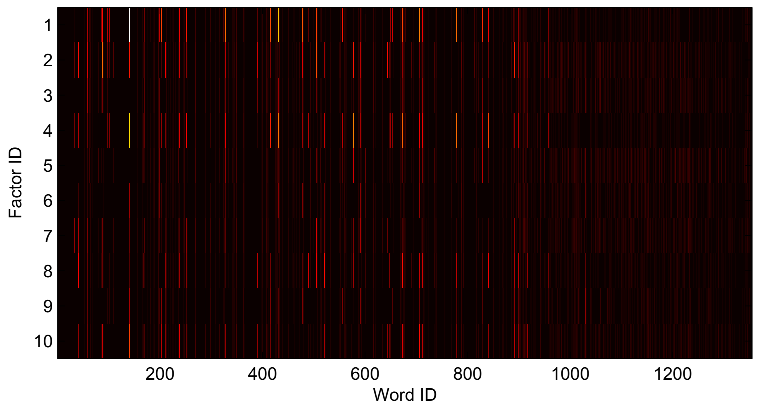


Figure 3.3: Heatmap of W. It is inter and intra-column sparse.

Sparsity of W. Figure 3.3 shows the heat map of our learnt W. It shows that we are indeed able to learn the desired sparsity structure: (a) few words are chosen (feature selection) as seen from few columns being bright, and (b) each chosen word corresponds to few factors.

Studying W reveals further insights on the stocks. We consider the ten most positive and negative words of two latent factors as listed in Table 3.5. We note that the positive word list of one factor has significant overlap with the negative word list of the other factor. This leads us to hypothesize that the two factors are anticorrelated.

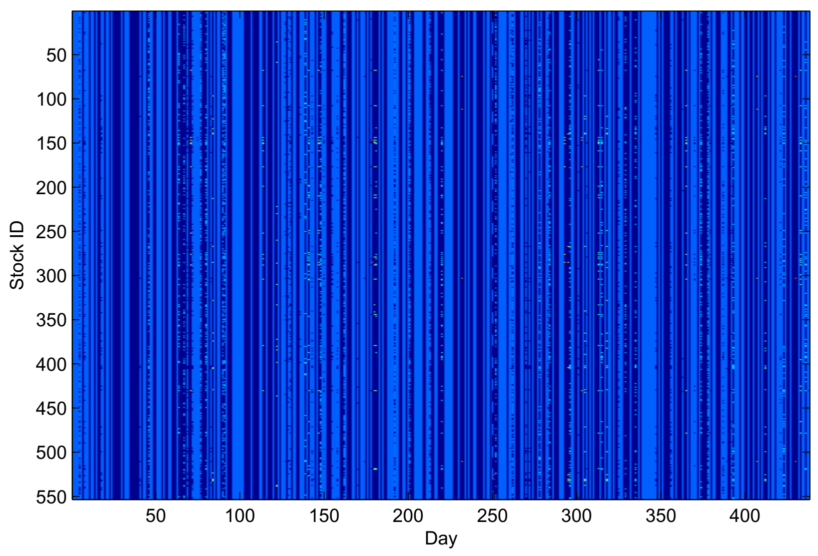
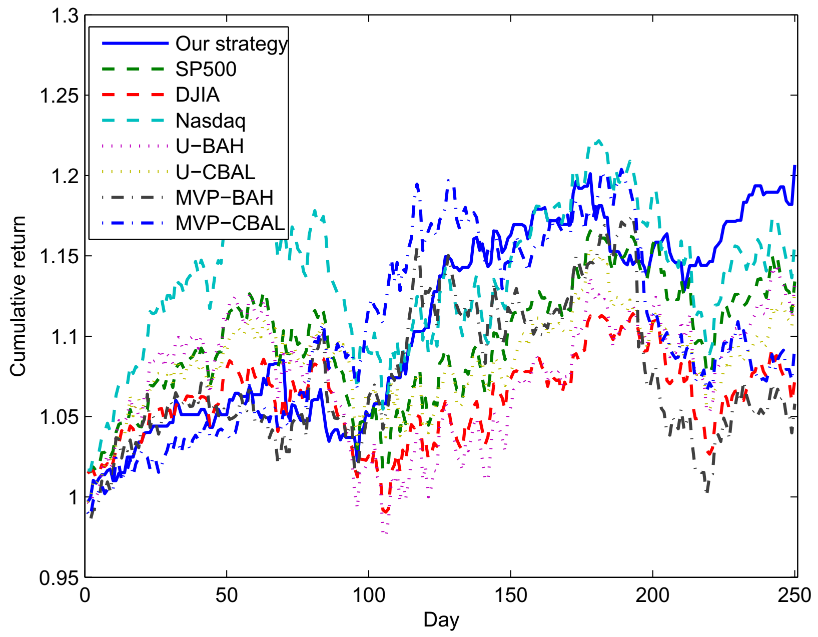


Figure 3.4: Stock weights from portfolio due to our strategy. Region left (right) of dashed line corresponds to 2012 (2013).

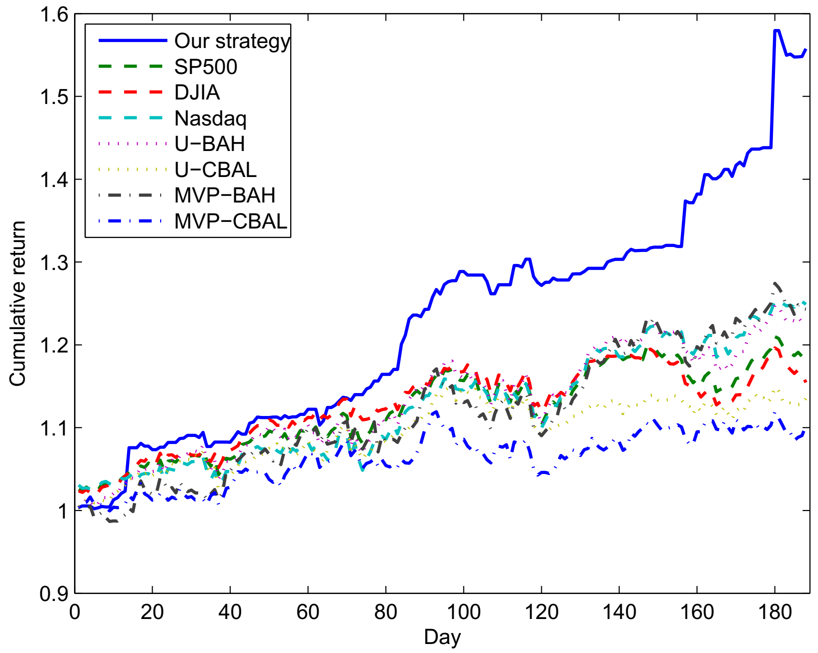
To test this hypothesis, we find the two sets of stocks that are dominant in one factor:[7] {IRM, YHOO, RYAAY} are dominant in factor 1, and {HAL, FFIV, MOS} are dominant in factor 2. Then we pair up one stock from each set by the stock exchange from which they are traded: YHOO and FFIV from NASDAQ, and IRM and HAL from NYSE. We compare the two stocks in a pair by their performance (in cumulative returns) relative to the stock index that best summarizes the stocks in the exchange (*e.g.,* S&P 500 for NYSE), so that a return below that of the reference index can be considered losing to the market, and a return above the reference means beating the market. Figure 3.6 shows that two stocks with different dominant factors are in opposite beating/losing positions (relative to the reference index) for most of the time, and for the (IRM, HAL) pair the two stocks interchange beating/losing positions multiple times.

Visualizing learnt portfolio and returns. We try to gain a better understanding of our trading strategy by visualizing the learnt stock portfolio. Figure 3.4 shows (bright means higher weight to the corresponding stock) that our trading strategy alternates between three options on each day: (a) buy all stocks when an optimistic market is expected, (b) buy no

[7]That is, the stock's strength in that factor is in the top 40% of all stocks *and* its strength in the other factor is in the bottom 40%.



(a) In 2012.



(b) In 2013.

Figure 3.5: Cumulative returns.

Table 3.5: Top ten positive and negative word lists of two factors.

List	Words
Factor 1, positive	street billion goal designed corporate ceo agreement position buyers institute
Factor 1, negative	wall worlds minutes race free short programs university chairman opposition
Factor 2, positive	wall start opposition lines asset university built short race risks
Factor 2, negative	agreement designed billion tough bond set street goal find bush

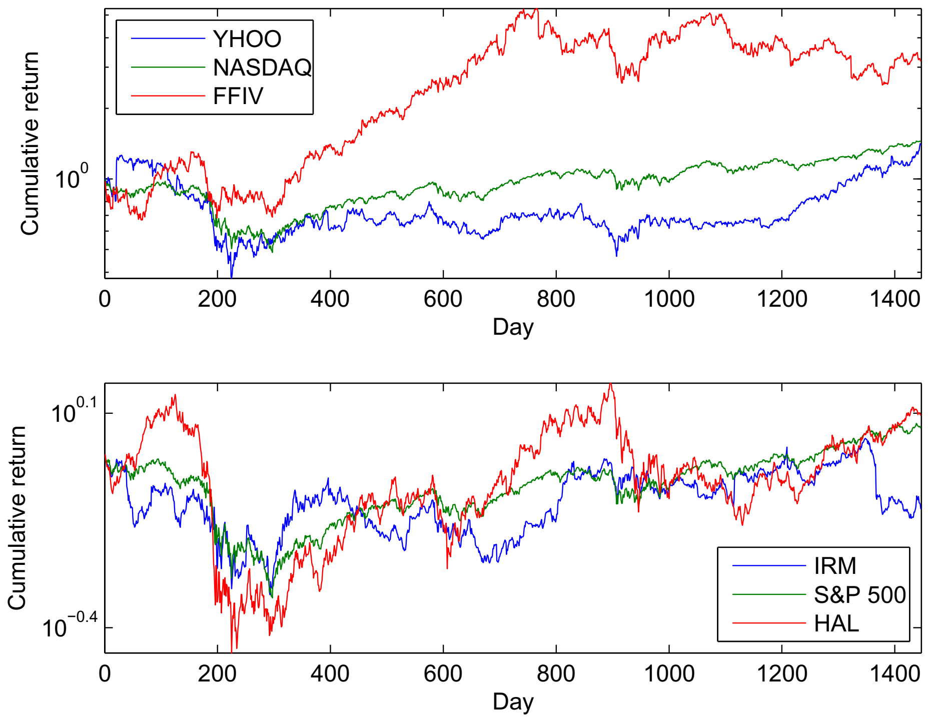


Figure 3.6: Returns of stocks with a different dominating factor. Green line is the reference index.

stocks when market pessimism is detected, and (c) buy a select set of stocks. The numbers of days with (a) or (b) chosen are roughly the same, while that of (c) are fewer but still significant. This shows our strategy is able to intelligently select stocks to buy/avoid in response to market conditions.

Reaction to important market events. To understand why our strategy results in better returns than the baselines, we also plot the cumulative returns of the different trading strategies. Figure 3.5(a) reveals that our strategy is more stable in growth in 2012, in that it avoids several sharp drops in value experienced by other strategies (this can also be seen from the fact that our strategy has the lowest maximum drawdown and CVaR). Although it initially performs worse than the other baselines (Nasdaq in particular), it is able to catch up and eventually beat all other strategies in the second half of 2012. It appears the ability to predict market drawdown is key for a good trading strategy using newspaper text (also see [86]).

Looking deeper we find *WSJ* to contain cues of market drawdown for two of the five days in 2012 and 2013 that have S&P 500 drop by more than 2%. On 6/1/2012, although a poor US employment report is cited as the main reason for the drawdown, the looming European debt crisis may have also contributed to a negative investor sentiment, as seen by "euro" being used in many *WSJ* articles on that day. On 11/7/2012, the US presidential election results cast fears on a fiscal cliff and more stringent controls on the finance and energy sectors. Many politics-related words, *e.g.,* democrats, election, won, voters, were prominent in *WSJ* on that day.

In 2013, our strategy is also able to identify and invest in rapidly rising stocks on several days, which resulted in superior performance. We note the performance of our algorithm in the two years are not the same, with 2013 being a significantly better year. To understand why, we look into the markets, and notice 2013 is an "easier" year because (a) other baseline algorithms also have better performance in 2013, and (b) the volatility of stocks prices in 2012 is higher, which suggests the prices are "harder" to predict. In terms of S&P

500 returns, 2012 ranks 10th out of 16 years since 1997, while 2013 is the best year among them.

3.7 Related Work

Our discussion here focuses on works that study the connection between news texts (including those generated from social media) and stock prices. Portfolio optimization (*e.g.,* [67, 24, 10, 3, 35] and references therein) is an important area in financial econometrics, but it is not directly relevant to our work because it does not incorporate news data.

The predictive power of news articles to the financial market has been extensively studied. Tetlock [86] applied sentiment analysis to a *Wall Street Journal* column and showed negative sentiment signals precede a decline in DJIA. Chan [15] studied newspaper headlines and showed investors tend to underreact to negative news. Dougal et al. [25] showed that the reporting style of a columnist is causally related to market performance. Wüthrich et al. [101], Lavrenko et al. [55], Fung et al. [33], Schumaker and Chen [82], Zhang and Skiena [103] used news media to predict stock movement with machine learning and/or data mining techniques. On top of using news, other text sources are also examined, such as corporate announcements [46, 73], online forums [88], blogs [103], and online social media [103, 66]. See [71] for a comprehensive survey.

Comparison to existing approaches. Roughly speaking, most prediction algorithms discussed above follow the same framework: first an algorithm constructs a feature vector based on the news articles. Next the algorithm will focus on prediction on the subset of stocks or companies mentioned in the news. Different feature vectors are considered, *e.g.,* Lavrenko et al. [55] used vanilla bag-of-word models while Zhang and Skiena [103] extracted sentiment from text. Also, most "off-the-shelf" machine learning solutions, such as generalized linear models [86], Naive Bayes classifiers [55], and Support Vector Machines

[82] have been examined in the literature. Our approach differ from the existing ones in the following two ways:

(1) No NLP. Unlike [86, 82, 46], we do not attempt to interpret or understand news articles with techniques like sentiment analysis and named entity recognition. In this way, the architecture of our prediction algorithm becomes simpler (and thus has lower variance).

(2) Leveraging correlation between stocks. Lavrenko et al. [55], Fung et al. [33] also made predictions without using NLP, but all these algorithms do not leverage the correlations that can exist between different stocks. It is not clear how these algorithms can be used to predict a large number of stocks without increasing model complexity substantially.

3.8 Conclusions

In this chapter we revisit the problem of mining text data to predict the stock market. We propose a unified latent factor model to model the joint correlation between stock prices and newspaper content, which allows us to make predictions on individual stocks, even those that do not appear in the news. Then we formulate model learning as a sparse matrix factorization problem solved using ADMM. Extensive backtesting using almost six years of *WSJ* and stock price data shows our method performs substantially better than the market and a number of portfolio building strategies. We note our methodology is generally applicable to all sources of text data, and we plan to extend it higher frequency data sources such as Twitter.

3.A Solving matrix equation $AXB + X = C$.

To solve for X, we apply the Hessenberg-Schur method [42] as follows:

1. Compute $H = U^T AU$, where $U^T U = I$ and H is upper Hessenberg, *i.e.,* $H_{ij} = 0$ for all $i > j + 1$.

2. Compute $S = V^T BV$, where $V^T V = I$ and S is quasi-upper triangular, *i.e.,* S is triangular except with possible 2×2 blocks along the diagonal.

3. Compute $F = U^T CV$.

4. Solve for Y in $HYS^T + Y = F$ by back substitution.

5. Solve for X by computing $X = UYV^T$.

To avoid repeating the computationally expensive Schur decomposition step (step 2), we precompute and store the results for use across multiple iterations of ADMM. This prevents us from using a one-line call to numerical packages (*e.g.,* `dlyap`$()$ in Matlab) to solve the equation.

Here we detail the back substitution step (step 4), which was omitted in [42]. Following [42], we use m_k and m_{ij} to denote the k-th column and (i, j)-th element of matrix M respectively. Since S is quasi-upper triangular, we can solve for Y from the last column, and then back substitute to solve for the second last column, and so on. The only complication is when a 2×2 nonzero block exists; in that case we solve for two columns simultaneously. More specifically:

(a) If $s_{k,k-1} = 0$, we have

$$H\left(\sum_{j=k}^{n} s_{kj} y_j\right) + y_k = f_k$$

$$(s_{kk} H + I) y_k = f_k - H \sum_{j=k+1}^{n} s_{kj} y_j,$$

then we can solve for y_k by Gaussian elimination.

(b) If $s_{k,k-1} \neq 0$, we have

$$H \begin{bmatrix} y_{k-1} & y_k \end{bmatrix} \begin{bmatrix} s_{k-1,k-1} & s_{k,k-1} \\ s_{k-1,k} & s_{kk} \end{bmatrix} + \begin{bmatrix} y_{k-1} & y_k \end{bmatrix}$$
$$= \begin{bmatrix} f_{k-1} & f_k \end{bmatrix} - \sum_{j=k+1}^{n} H \begin{bmatrix} s_{k-1,j} y_j & s_{kj} y_j \end{bmatrix}.$$

The left hand side can be rewritten as

$$\begin{aligned} & H \begin{bmatrix} s_{k-1,k-1} y_{k-1} + s_{k-1,k} y_k & s_{k,k-1} y_{k-1} + s_{kk} y_k \end{bmatrix} + \begin{bmatrix} y_{k-1} & y_k \end{bmatrix} \\ &= \begin{bmatrix} (s_{k-1,k-1} H + I) y_{k-1} + s_{k-1,k} H y_k & s_{k,k-1} H y_{k-1} + (s_{kk} H + I) y_k \end{bmatrix} \\ &= \begin{bmatrix} s_{k-1,k-1} H + I & s_{k-1,k} H \\ s_{k,k-1} H & s_{kk} H + I \end{bmatrix} \begin{bmatrix} y_{k-1} \\ y_k \end{bmatrix} \end{aligned}$$

by writing $\begin{bmatrix} y_{k-1} & y_k \end{bmatrix}$ as $\begin{bmatrix} y_{k-1} \\ y_k \end{bmatrix}$. The right hand side can also be rewritten as

$$\begin{bmatrix} f_{k-1} \\ f_k \end{bmatrix} - \sum_{j=k+1}^{n} \begin{bmatrix} s_{k-1,j} H y_j \\ s_{kj} H y_j \end{bmatrix}.$$

Thus we can solve for columns y_k and y_{k-1} at the same time through Gaussian elimination on

$$\begin{bmatrix} s_{k-1,k-1} H + I & s_{k-1,k} H \\ s_{k,k-1} H & s_{kk} H + I \end{bmatrix} \begin{bmatrix} y_{k-1} \\ y_k \end{bmatrix} = \begin{bmatrix} f_{k-1} \\ f_k \end{bmatrix} - \sum_{j=k+1}^{n} \begin{bmatrix} s_{k-1,j} H y_j \\ s_{kj} H y_j \end{bmatrix}.$$

3.B Proof of Lemma 3.4.1

Proof. u^* is a minimizer iff $0 \in \partial F(u^*)$, where

$$\partial F(u) = \lambda \partial \|u\|_2 + \mu \partial \|u\|_1 + \nabla \frac{\rho}{2}\|u - v\|_2^2, \text{ with}$$

$$\partial \|u\|_2 = \begin{cases} \left\{ \dfrac{u}{\|u\|_2} \right\} & u \neq 0 \\ \{s \mid \|s\|_2 \leq 1\} & u = 0 \end{cases}$$

$$\partial \|u\|_1 = [\partial |u_i|]$$

$$\partial |u_i| = \begin{cases} \{\mathrm{sgn}(u_i)\} & u_i \neq 0 \\ [-1, 1] & u_i = 0. \end{cases}$$

In the following, $\|\cdot\|$ denotes $\|\cdot\|_2$, and $\mathrm{sgn}(\cdot)$, $|\cdot|$, $(\cdot)^+$ are understood to be done elementwise if operated on a vector. There are two cases to consider:

Case 1: $\|w\| \leq \lambda$

This implies $u^* = 0$, $\partial \|u^*\|_2 = \{s \mid \|s\| \leq 1\}$, $\partial \|u^*\|_1 = \{t \mid t \in [-1, 1]^n\}$, and $\nabla \|u^* - v\|_2^2 = -\rho v$. Then

$$\begin{aligned} 0 \in \partial F(u^*) &\iff 0 \in \{\lambda s + \mu t - \rho v \mid \|s\| \leq 1, t \in [-1, 1]^n\} \\ &\iff \exists s : \ \|s\| \leq 1, \ t \in [-1, 1]^n \\ &\qquad \text{s.t.} \left(\lambda s + \mu t = \rho v \iff v - \frac{\mu}{\rho} t = \frac{\lambda}{\rho} s \right). \end{aligned}$$

Now we show an (s, t) pair satisfying the above indeed exists. Define $t = [t_i]$ such that

$$t_i = \begin{cases} \dfrac{\rho}{\mu} v_i & |v_i| \leq \dfrac{\mu}{\rho} \\ \mathrm{sgn}(v_i) & |v_i| > \dfrac{\mu}{\rho}. \end{cases}$$

If $|v_i| \leq \mu/\rho$, then $\rho/\mu(-\mu/\rho) \leq t_i \leq \rho/\mu(\mu/\rho) \Rightarrow t_i \in [-1,1]$. If $|v_i| > \mu/\rho$, then obviously $t_i \in [-1,1]$. Therefore we have $t \in [-1,1]^n$.

Now define $s = (\rho v - \mu t)/\lambda$. We first write

$$\begin{aligned}
\rho \operatorname{sgn}(v_i)|v_i| - \mu t_i &= \begin{cases} \rho v_i - \mu\Big(\dfrac{\rho}{\mu} v_i\Big) & |v_i| \leq \dfrac{\mu}{\rho} \\ \rho \operatorname{sgn}(v_i)|v_i| - \mu \operatorname{sgn}(v_i) & |v_i| > \dfrac{\mu}{\rho} \end{cases} \\
&= \begin{cases} 0 & |v_i| \leq \dfrac{\mu}{\rho} \\ \rho \operatorname{sgn}(v_i)\Big(|v_i| - \dfrac{\mu}{\rho}\Big) & |v_i| > \dfrac{\mu}{\rho} \end{cases} \\
&= \rho \operatorname{sgn}(v_i)\Big(|v_i| - \frac{\mu}{\rho}\Big)^+.
\end{aligned}$$

Then we show $\|s\| \leq 1$:

$$\begin{aligned}
\|s\| &= \frac{1}{\lambda}\|\rho v - \mu t\| \\
&= \frac{1}{\lambda}\|\rho \operatorname{sgn}(v)|v| - \mu t\| \\
&= \frac{1}{\lambda}\Big\|\rho \operatorname{sgn}(v)\Big(|v| - \frac{\mu}{\rho}\Big)^+\Big\| \\
&= \frac{1}{\lambda}\|w\| \leq 1.
\end{aligned}$$

Thus we have shown $0 \in \partial F(u^*)$ for $\|w\| \leq \lambda$.

Case 2: $\|w\| > \lambda$

Here $\|w\| - \lambda > 0$ and we have $u^* = (\|w\| - \lambda)/(\rho\|w\|) \cdot w$. Since $\|w\| \neq 0$ means $w \neq 0$, we also have $u^* \neq 0$.

Then $\partial\|u^*\|_2 = \{u/\|u\|\}$ and

$$\begin{aligned}
\partial F(u^*) &= \Big\{\frac{\lambda}{\|u^*\|} u^* + \rho(u^* - v)\Big\} + \mu\partial\|u^*\|_1 \\
&= \Big\{\Big(\frac{\rho\lambda}{\|w\| - \lambda} + \rho\Big) u^* - \rho v\Big\} + \mu\partial\|u^*\|_1,
\end{aligned}$$

where the last step makes use of $\|u^*\| = (\|w\| - \lambda)/(\rho\|w\|) \cdot \|w\| = (\|w\| - \lambda)/\rho$.

Our goal is to show $0 \in \partial F(u^*)$, which is true iff it is valid elementwise, *i.e.,*

$$0 \in \partial F_i(u^*) = \left\{ \left(\frac{\rho\lambda}{\|w\| - \lambda} + \rho \right) u_i^* - \rho v_i \right\} + \mu \partial |u_i^*|.$$

We consider two subcases of each element u_i^*.

(a) The case $u_i^* = 0$ results from $w_i = 0$, which in turn results from $|v_i| \leq \mu/\rho$. Then

$$\begin{aligned}\partial F_i(u^*) &= \left\{ \left(\frac{\rho\lambda}{\|w\| - \lambda} + \rho \right) \cdot 0 - \rho v_i \right\} + \mu \partial |0| \\ &= \{\mu s - \rho v_i \mid s \in [-1, 1]\} \\ &= [-\mu - \rho v_i, \mu - \rho v_i].\end{aligned}$$

Note that for all v_i with $|v_i| \leq \mu/\rho$ the above interval includes 0, since

$$\begin{aligned}-\mu - \rho v_i \leq -\mu - \rho\Big(-\frac{\mu}{\rho}\Big) = 0 \\ \mu - \rho v_i \geq \mu - \rho\Big(\frac{\mu}{\rho}\Big) = 0.\end{aligned}$$

Thus $0 \in \partial F_i(u^*)$.

(b) The case $u_i^* \neq 0$ corresponds to $|v_i| > \mu/\rho$. Then

$$\begin{aligned}\partial F_i(u^*) &= \left\{ \left(\frac{\rho\lambda}{\|w\| - \lambda} + \rho \right) u_i^* - \rho v_i \right\} + \{\mu \operatorname{sgn}(u_i^*)\} \\ &= \left\{ \frac{\rho\|w\|}{\|w\| - \lambda} u_i^* - \rho v_i + \mu \operatorname{sgn}(v_i) \right\} \\ &= \left\{ \frac{\rho\|w\|}{\|w\| - \lambda} \frac{\|w\| - \lambda}{\rho\|w\|} \rho \operatorname{sgn}(v_i) \Big(|v_i| - \frac{\mu}{\rho} \Big) - \rho v_i + \mu \operatorname{sgn}(v_i) \right\} \\ &= \{\rho v_i - \mu \operatorname{sgn}(v_i) - \rho v_i + \mu \operatorname{sgn}(v_i)\} = \{0\},\end{aligned}$$

where the second step comes from $\operatorname{sgn}(u_i^*) = \operatorname{sgn}(v_i)$ by definition of u_i^*. Thus $0 \in \partial F_i(u^*)$ for $\|w\| > \lambda$. $\square$

Chapter 4

On the Efficiency of Social Recommender Networks

4.1 Introduction

High-quality recommender systems are the hallmark of many online services, such as Netflix, Yelp and Amazon. They are also critical to many organizations in traditional sectors with a strong online presence, including the New York Times, CNN and the PlayStation Plus gaming network. When a recommender system is effective in showing users relevant and useful items (*e.g.,* a book from Amazon, a news article from CNN, or a game from PS Plus), users are more likely to make a purchase and/or re-visit the recommender system's website, which can result in a substantial increase in site traffic and revenue.

A recent trend in the design of recommender systems is the introduction of "social" features. For example, Yelp personalizes restaurant rankings by taking into account the opinions of a user's friends. Netflix shows a user what her friends have recently watched. Amazon allows users to share their purchases on Facebook and/or Twitter. These features have benefits that are apparent: it is usually more fun to interact with friends, and hopefully, friends will be able to influence each other to make purchases, or visit restaurants, which will increase a website's revenue. In addition, social features can impact the entire recommender system in ways that are less visible, but perhaps even more important. Specifically, (a) at the individual level, friends are more trusted and tend to share common interests. Thus, individuals are more likely to adopt their friends' recommendations [49, 16]. (b) At the macroscopic level, recommendations are more likely to propagate through social ties. This word-of-mouth effect plays an important role in promoting less-known but otherwise excellent businesses or products [43, 95].

In other words, the social network in a recommender system plays at least two important roles: (a) it helps individuals improve their experience through interaction with their friends. Let us refer to the resultant benefit as the *local utility* of the social network. (b) It also helps the recommender system in disseminating information or recommendations more efficiently. Let us refer to the resultant benefit as the *global utility* of the social network. Here, an important question arises:

> *For the social network in a recommender system, are its local and global utilities compatible?*

More specifically, if the social network is constructed in such a way that user experience is optimized, does it come at the cost of being less effective in propagating information? And vice versa: if we optimize the social network for information propagation, will user experience be sacrificed?

Such a question becomes more interesting as social networks are usually formed in a decentralized manner [100]: two people can establish a link so long as both parties agree to do so. They often do not have knowledge of the global properties of the network. They make a connection usually not for the sole purpose of sharing opinions and recommendations. It is also not always clear to an individual whether a potential new friend will have high-quality recommendations. Thus, it would be quite surprising to find this crude network formation mechanism to be able to optimize local and/or global utilities.

Our work and contributions. This chapter presents an extensive study of the local-global utility tradeoff in a recommender system's social network, or a *social recommender network*, focusing on a recently-released dataset from Yelp. Our contributions can be summarized as follows.

1. Stochastic model for recommendation diffusion. We propose a simple but natural stochastic model for the diffusion of recommendations in social networks. Based on this model, we propose a set of statistically-grounded yet intuitive metrics to quantify the local and global utilities of a social recommender network. Intuitively, one's local utility measures the discrepancy between her ratings and her friends' ratings. When the discrepancy is small, friendship-based recommendation will be more reliable, which results in higher local utility (see Eq. (4.1)). Meanwhile, a network's global utility measures how fast a recommendation travels. Here, average hitting time is used to measure global utility (see Eq. (4.2)).

The set of metrics we propose are fairly robust, and work well even when the actual recommendation diffusion model is misspecified.

2. Empirical study of a Yelp dataset. We next study, based on the proposed metrics, the tradeoff between local and global utilities with data from Yelp, a major rating and review website for local businesses. Our discoveries are as follows.

- When a user has only one friend, it is unclear how much the user benefits from her social ties, but when the user has more than one friend, we see improvements in the quality of recommendations from her friends.
- The global utility of Yelp's social network is also quite good. It is comparable to that of a random graph, which is known to be efficient at propagating information. On the other hand, the tradeoff between global and local utilities is not optimized, so we need the third part of our study, as discussed below.

3. Algorithm for social recommender network optimization. We also design an algorithm for finding a social network topology that simultaneously addresses individuals' need to establish links to high-quality friends, and service providers' need to maximize network efficiency in information propagation. Recommender systems and social networks are extensively studied areas with numerous works in improving a recommender system's user experience (see [52, 85] and the references therein) and maximizing network efficiency through topology design (*e.g.,* [13, 39, 18, 1]), but to the best of our knowledge, there exists no prior work that simultaneously addresses both objectives. We believe this is a promising direction for recommender systems research.

Why study Yelp. We have two reasons to focus on the Yelp dataset. First, Yelp is the only large online service, of which we are aware, to make both recent[1] rating and social network data publicly accessible.[2] Second, unlike other providers such as Amazon and Netflix, Yelp does not directly use Facebook's API or social network, but instead builds its own in-house social networking service. As a result, its users know they will interact with their Yelp friends only on the Yelp platform, and there are much fewer connections established for reasons other than sharing information in Yelp.

4.2 Model and Metrics

We shall start with a fairly natural but stylized model of the diffusion of recommendations and reviews in a social network. Based on this model, we define both an individual's utility and a network's efficiency in an intuitive way. In the next section we will analyze this model and present a number of key results in computing the proposed local and global utilities. Along the way, we will also discuss the scenarios where the model is misspecified, or when a portion of the data are missing.

[1]Including data from 2014.
[2]Without the need of scraping and crawling.

Let us denote the social network as $G = \{V, E\}$, where $V = \{v_1, v_2, \ldots, v_n\}$. Let $B = \{b_1, b_2, \ldots, b_\ell\}$ be the set of businesses (*e.g.,* restaurants) of interest. We use the following discrete-time stochastic process to model how users visit the businesses and write reviews. Here, our process is parametrized by a "strength-of-tie" variable $W = [W_{i,j}] \in \mathbb{R}_+^{n \times n}$. Intuitively, the number $W_{i,j}$ represents the strength of the social tie between v_i and v_j, *i.e.,* if v_i and v_j are close friends, then $W_{i,j}$ should be large, and vice versa. We also require $W_{i,j} = W_{j,i}$, and $W_{i,j} = 0$ if $\{v_i, v_j\} \notin E$, *i.e.,* there should not be any tie between two users if they do not know each other. As the social network is formed in a decentralized manner, we imagine the value of $W_{i,j}$ to be chosen by v_i and v_j, but not anyone else.

Our discrete-time process works as follows: at time $t = 0$, each business $b \in B$ is visited by a user, call it $p_0(b) \in V$. After user $p_0(b)$ visits b, she forms an opinion regarding b (*i.e.,* writes a review), call it $R(p_0(b))$. Here, $R(\cdot)$ is a real number and is large when the user has high opinion of b. In each of the subsequent times $t \geq 1$ and for each b, one neighbor of $p_{t-1}(b)$ visits the business, and the probability that some v is this visiting neighbor, *i.e.,* $p_t(b) = v$, is proportional to $W_{p_{t-1}(b),v}$. After the user visits the business, she also generates a review $R(p_t(b))$. This assumption captures the fact that close friends are more likely to influence each other. Notice that in this stochastic model, we allow the user to revisit the same business multiple times, but for simplicity, we assume a user's review to remain unchanged after a revisit.

Based on this process, we now are able to define both the local and global utilities of the social network.

Local utility. We make use of the following intuition to define an individual's local utility. When a user decides to visit a business because of her friend's influence, she should consider her friend's review useful only if it is consistent with her own opinion. Thus, we use the discrepancy between $R(p_{t-1}(b))$ and $R(p_t(b))$ to measure the social network's local utility.

We now formalize this intuition. Let v be an arbitrary user, and $e_t(v)$ be the set of businesses that v visits at time t. Note that $e_t(v)$ could be an empty set, in the case of v not visiting any businesses at time t. For arbitrary $b \in e_t(v)$, denote v's review on b as R_b^t, and v's neighbor's review on b at time $t-1$ as R_b^{t-1}. We define user v's local cost as

$$\lim_{t\to\infty} \mathrm{E}\left[\frac{1}{|e_t(v)|}\sum_{b\in e_t(v)} |R_b^t - R_b^{t-1}| \;\middle|\; e_t(v) \neq \emptyset\right]. \tag{4.1}$$

With slight abuse in word usage, we also refer to (4.1) as v's local utility, the lower the better.

We make a few remarks here. First, as we are interested in only the times at which v visits a business, we focus on the conditional expectation $\mathrm{E}[\cdot \mid e_t(v) \neq \emptyset]$. Second, we use $|R_b^t - R_b^{t-1}|$ to quantify the discrepancy between v's opinion and her friend's opinion. There are other choices but we use the ℓ_1-norm because it is more robust against outliers. Also, we need to average out the discrepancies in the case of $|e_t(v)| > 1$, as the volume of a user's reviews should be orthogonal to the quality of her friends' reviews. Finally, we take $t \to \infty$ as we are more interested in the system's equilibrium behavior.

Global utility. Intutively, the global utility function should be able to measure how fast information propagates. Under our recommendation diffusion model, we take it as the average waiting time for each user to review a specific business. More specifically, let $v_i = p_0(b)$ be the first person that reviews business b, and let $T(v_j, b) = \min\{t : p_t(b) = v_j\}$. The average waiting time can be defined as the mean of $T(v_j, b)$, over all v_j, *i.e.,* $\frac{1}{n}\mathrm{E}[\sum_j T(v_j, b)]$. Furthermore, we also require the average waiting time to be uniformly good, regardless the starting point $p_0(b)$. Thus, our final objective is

$$\mathrm{E}_{p_0(b)\sim_U V}\left\{\frac{1}{n}\mathrm{E}\Big[\sum_j T(v_j, b)\Big]\right\}, \tag{4.2}$$

where $p_0(b) \sim_U V$ means that the first reviewer of b is uniformly sampled from V.

4.3 Analysis of Utilities and Model Misspecification

4.3.1 Local Utility

Efficient computation. We now describe the procedure for estimating an individual's utility, starting with the assumption that there are no missing data. As all users will rate all businesses as $t \to \infty$, we refer to $R_{i,k}$ as v_i's rating on business b_k. Then we have the following proposition, which is proved in the Appendix.

Proposition 4.3.1. *Consider the aforementioned diffusion model. For a specific user v_i, we have*

$$\lim_{t\to\infty} \mathrm{E}\left[\frac{1}{|e_t(v_i)|}\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\middle|\, e_t(v_i) \neq \emptyset\right] = \frac{1}{\ell}\sum_{k\leq \ell}\sum_{j} \frac{W_{i,j}|R_{i,k} - R_{j,k}|}{\sum_j W_{i,j}}.$$

Then by Proposition 4.3.1, the local utility metric D_i of user v_i can be calculated as

$$D_i = \frac{1}{\ell}\sum_{k} \frac{\sum_j W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}}. \tag{4.3}$$

To summarize the local utility of the whole network, we take average over all users:

$$\overline{D} = \frac{1}{n}\sum_{i=1}^{n} D_i. \tag{4.4}$$

Missing data. We consider an approximation to Eq. (4.3) to account for the missing data problem in real datasets, *i.e.,* there are many (j, k) pairs such that $R_{j,k}$ is not available in the dataset. When $R_{j,k}$ is missing, we use the population mean of k as our estimate of $R_{j,k}$. This is justified by the fact that the population mean is the maximum-likelihood estimate of $R_{j,k}$, assuming the ratings of k are i.i.d. Thus, we use the formula below to compute a user's local utility:

$$D_i = \frac{1}{|\mathcal{R}_i|} \sum_{k \in \mathcal{R}_i} \frac{\sum_{j=1}^{n} W_{i,j} |\tilde{R}_{j,k} - R_{i,k}|}{\sum_{j=1}^{n} W_{i,j}} \tag{4.5}$$

$$\tilde{R}_{j,k} = \begin{cases} R_{j,k} & R_{j,k} \text{ is available} \\ r_k & \text{otherwise,} \end{cases} \tag{4.6}$$

where $\mathcal{R}_i = \{k \mid R_{i,k} \text{ is available}\}$, and r_k is the population mean of k. While the above equation is motivated by our stylized model, it also has a fairly natural interpretation, and thus even when our model is misspecified, Eq. (4.5) still offers a reasonable approach to estimate local utilities. More specifically, the above formula possesses two interesting properties:

1. Having no friends does not harm you. If you need a recommendation on a specific business but none of your friends have ever rated it, the social network should not harm you, *i.e.,* the recommender system platform should return the average rating of that business.

2. Having no or few ratings from friends is informative. If some of your friends have rated a business, then we expect the recommender system to take into account the following two factors: (a) the values of the friends' ratings, and (b) the proportion of friends that have rated the business. Consider two scenarios, where in the first one, only 1% of your friends have rated the business, while in the second one, more than 40% of your friends have rated it. Our formula gives a metric that assigns less weight on friends' opinions for the first case, and more weight on friends' opinions for the second case.

4.3.2 Global Utility

Computation. The hitting time $H_{i,j}$ is the time taken for user i to hear about a business that was originally discovered by user j. Averaging over all pairs of users, we define $\overline{H}$ as

$$\overline{H} = \frac{1}{n(n-1)} \sum_{i,j=1}^{n} H_{i,j}, \tag{4.7}$$

which is related to the eigenvalues of the unnormalized graph Laplacian: $L = \text{diag}(W\mathbf{1}) - W$. Let the sorted eigenvalues of L be $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. It can be proved [39] that $\overline{H}$ and $\{\lambda_i\}$ are related as

$$\overline{H} = \frac{2\sum_{i<j} W_{i,j}}{n(n-1)} R_{\text{tot}} \tag{4.8}$$

$$R_{\text{tot}} = n \sum_{i=2}^{n} \frac{1}{\lambda_i}, \tag{4.9}$$

where R_{tot} is known as the effective resistance of W. Note that smaller $\overline{H}$ means better global utility.

Robustness. Let us start by considering the plausible candidates for measuring global utility. Recall a network is "high-dimensional" as there can be as many as $\binom{n}{2}$ edges in a network with n nodes (so we need $\binom{n}{2}$ independent variables to represent it). Thus, we shall look for a concise statistic that can well summarize its efficiency in information propagation.

This is an extensively studied area and numerous stochastic models have been proposed (see [18, 1, 53, 19] for examples). We observe that the efficiency in information propagation from each of the models is invariably characterzied by one of these closely related statistics: (a) the second largest eigenvalue of the Laplacian or transition matrix [57], (b) the conductance of the network [18], (c) mixing time, and (d) hitting time, which is used for our model. All of these statistics are closely related, *e.g.,* if we know the average hitting time of a Markov chain, we can compute (tight) upper and lower bounds of its mixing time, conductance and second largest eigenvalue. Thus, they are all roughly equivalent, *i.e.,* when a social network is optimized for average hitting time, it is also asymptotically optimal in almost all other information diffusion models (see [57] for more discussions), and average hitting time is a fairly robust statistic.

We also remark that the second largest eigenvalue and mixing time are less appropriate statistics in our setting. Even in our simple model, mixing time or the second largest eigenvalue alone is insufficient to bound average hitting time. Knowledge of the stationary distribution is also needed (see Fact 4.3.1), but by Occam's razor principle, we prefer a simpler model with fewer parameters (*i.e.,* not specifying the stationary distribution), and there is also no natural way to interpret the stationary distribution of a social network.

Fact 4.3.1. *There exists a growing family of networks* $\{M(\tau)\}_{\tau>0}$ *such that (a) the mixing time of* M *is* 1*, (b) the second largest eigenvalue of the associated transition matrix is* 0*, and (c) the average hitting time tends to* ∞ *as* $\tau \to \infty$*.*

4.4 Dataset and Associated Graphs

The social recommender network we study is the Yelp social network from Phoenix, Arizona.[3] The dataset contains 335,022 reviews made by 70,817 users on 15,585 businesses, and the users' social network with 151,516 edges.

Our analysis focuses on two graphs. The first is the provided social graph (network), which is constructed by users in a decentralized but strategic manner. It is the focus of our empirical study for understanding the tradeoff between local and global utilities. The second graph is constructed from users' ratings. It represents how users "organically" interact with each other, either directly or indirectly in Yelp, when no intervention is presented. For instance, if two users have co-rated a large number of businesses, we consider them to have

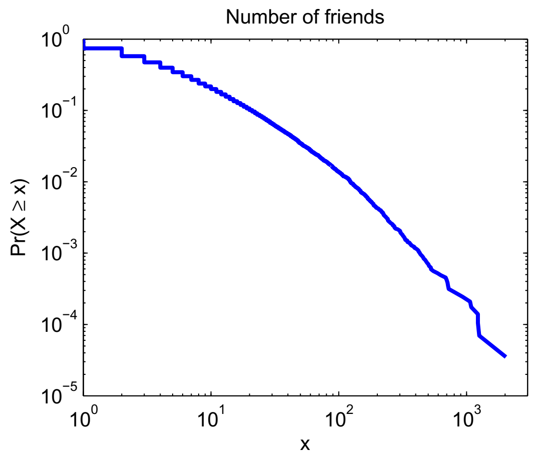
[3]`http : //www.yelp.com/dataset_challenge`. We use the data released for the third round of the Yelp Dataset Challenge.

strong interaction. This graph serves as a baseline for comparison with the social graph. Below, we describe the details of these two graphs.

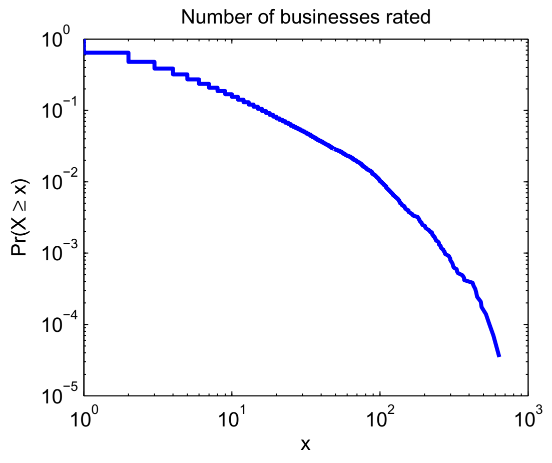
Social graph. The Yelp social network is built in-house (*i.e.,* it does not solely rely on Facebook or other third party API to manage the network), although users may import their friends from email or Facebook accounts. Following related work in OSN analysis [74], we focus on the largest connected component of the social graph. While the largest connected component has only 28,977 users, the remaining users appear not interested in any social activity, as seen from the second largest component having only four users. We note that our analysis is meaningful only for the users who care about their social network. To allow for a direct comparison with the co-rating graph (see details below), we further remove 330 users that do not have sufficient reviewing activity and are isolated in the co-rating graph. The result is a connected social graph with 28,647 users and 149,348 edges.

Co-rating graph. The co-rating graph is constructed such that two users share an edge if and only if they have co-rated at least one business. This graph can be interpreted as the largest network that can be discovered by users reading the reviews of other users. Since our metrics are based on reviews and ratings, we also expect a "good" network (*i.e.,* a network that has good local and global utilities) to be embedded in this graph with a suitable choice of edge weights and topology (*i.e.,* pruning useless edges). Focusing on only the 28,647 users seen in the social graph, this graph has 8.13 million edges, 54 times more edges than the social graph.

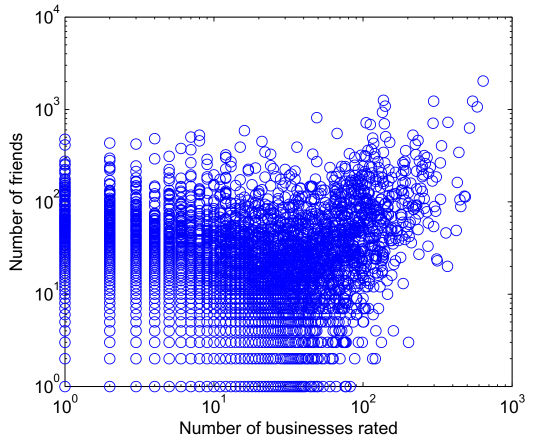
Figure 4.1 shows the basic properties of the Yelp data in terms of making friends and reviewing businesses, the two major forms of user activity. The degree distribution exhibits higher skew than review activity distribution, which is consistent with the intuition that reviewing a business (writing and typing in text) requires more effort than adding a friend (clicking a "friend request" button). Nevertheless, the two forms of user activity are positively correlated, especially towards the tail (see Figure 4.1(c)).



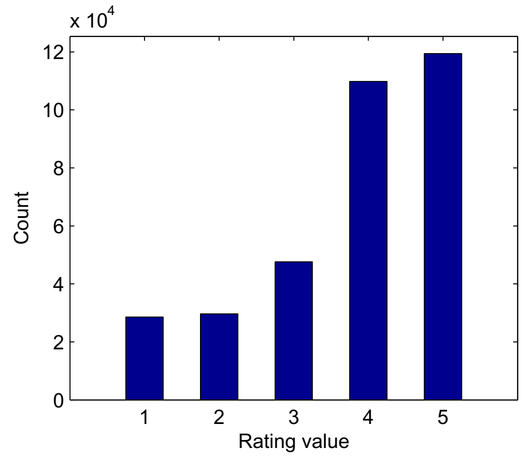
(a) Degree distribution.



(b) Review activity distribution.



(c) Degree correlates with reviews.



(d) Rating distribution.

Figure 4.1: Basic data statistics.

Given the two graphs, we consider variations of them through altering their topologies and edge weights.

Perturbing network topology. We also compare the social graph with a random graph baseline. There are two reasons for doing so. (a) A random graph represents the "average behavior" of all graphs sharing the same degree distribution. Thus, such a comparison allows us to understand how the social graph differs from an "average graph." (b) Random graphs are known to have small mixing time, small conductance, and small second largest eigenvalues [9], *i.e.,* they are efficient in information propagation. We consider a network efficient if its global utility is comparable to that of random graphs.

Starting from the social graph, we shuffle its edges to obtain a edge-perturbed graph. Algorithm 2, similar to that presented in [92], ensures the resultant graph is connected (by checking for connectedness after every `window` steps, and if not, reverting to the previous snapshot G') and the degree of each node is preserved (since each shuffle is degree-preserving). By running Algorithm 2 with increasing `num_iter`, we obtain a family of graphs with increasing distance (measured by the number of differing edges) from the original social graph, and in the limit `num_iter` $\to \infty$, we obtain a random graph such that each node's degree is the same as its degree in the social graph.

Algorithm 2 Degree and connectedness-preserving shuffling.

```
Input: $G = \{V, E\}$, num_iter, window
  for $i = 1$ to num_iter do
    $G' \leftarrow G$
    repeat
      $G \leftarrow G'$
      for $j = 1$ to window do
        Sample two edges $\{a, b\}$ and $\{c, d\}$ from $E$
         s.t. $a \neq c$, $a \neq d$, $b \neq c$, $b \neq d$, and $\{a, c\}, \{b, d\} \notin E$
        Remove $\{a, b\}$ and $\{c, d\}$ from $E$
        Add $\{a, c\}$ and $\{b, d\}$ to $E$
      end for
    until $G$ is connected
  end for
```

Scaling edge weights. In reality, we expect each user to prioritize over different friends' recommendations, so we consider the following edge weighting strategies based on rating similarity.

- Jaccard coefficient on rating sets:

$$W_{i,j} = \frac{|\mathcal{R}_i \cap \mathcal{R}_j|}{|\mathcal{R}_i \cup \mathcal{R}_j|}.$$

  This captures the likelihood of a pair of users interacting through reading and writing reviews, but is agnostic to the values of the ratings.

- Pearson correlation coefficient on rating vectors:

$$W_{i,j} = 1 + \frac{\sum_{k \in \mathcal{R}_{i,j}} (R_{i,k} - r_i)(R_{j,k} - r_j)}{\sqrt{\sum_{k \in \mathcal{R}_{i,j}} (R_{i,k} - r_i)^2} \sqrt{\sum_{k \in \mathcal{R}_{i,j}} (R_{j,k} - r_j)^2}},$$

  where $\mathcal{R}_{i,j} = \mathcal{R}_i \cap \mathcal{R}_j$, $r_i = \sum_{k \in \mathcal{R}_i} R_{i,k} / |\mathcal{R}_i|$, and the 1 is added to make $W_{i,j} \in [0, 2]$.

- Cosine similarity on rating vectors:

$$W_{i,j} = \frac{\sum_{k \in \mathcal{R}_{i,j}} R_{i,k} R_{j,k}}{\sqrt{\sum_{k \in \mathcal{R}_{i,j}} R_{i,k}^2} \sqrt{\sum_{k \in \mathcal{R}_{i,j}} R_{j,k}^2}},$$

  defined such that $W_{i,j} \in [0, 1]$. Note that $W_{i,j} \geq 0$ because rating values are positive.

4.5 Empirical Data Analysis

4.5.1 Comparing Graphs

We compare the utility metrics of the graphs with different weight scaling strategies. Table 4.1 summarizes the results:

Comparing global utility. The Yelp social graph has better global utility $\overline{H}$ than all other graphs, except for the random graph baseline.[4] This is expected because random graphs have better efficiency properties (as discussed in Section 4.4), but the difference is only moderate (only 12% degradation), and we believe most of the edges in random graphs are not discoverable by users. Also, the slight loss in global utility is compensated by an improvement in local utility $\overline{D}$.[5]

[4]Generated by applying Algorithm 2 on the social graph with num_iter $=$ window $= 1000$, so as to perform 1 million edge shuffles.

[5]A statistically principled way to compare two graphs' local utilities is to apply a Wilcoxon signed rank test on their D_i vectors. We find the results to be consistent with our simple $\overline{D}$ comparison, *i.e.,* when one graph's $\overline{D}$ is better than that of the other, we also have the test rejecting the null hypothesis of two populations (D_i vectors) being equal at 5% level. The only exception is discussed in Section 4.5.4.

It is not surprising to see that the social graph has better global utility than the uniformly weighted co-rating graph, even though the latter has many more edges. We note that the $\overline{H}$ metric does not monotonically improve with the addition of edges. In fact, when the co-rating graph is used, the spread of a specific message could "get lost" in the network (due to having too many edges), reminiscent of findings from social navigation studies [51]. Perhaps more surprisingly, even when we use the other three rating similarity measures to weigh edges, $\overline{H}$ does not improve. This implies finding good edge weights is not straightforward.

Comparing local utility. Again, the social graph performs better than all other graphs under the $\overline{D}$ metric. This confirms that users are conscious of local utility when forming the social network. Comparing the three rating similarity measures, Pearson coefficient performs significantly better than the other two. To weigh confidence level into user similarity, *i.e.,* assign higher similarity to user pairs with more co-rated businesses, we also take the product of Jaccard and Pearson/Cosine coefficients as edge weights, but this strategy does not help in improving $\overline{D}$.

We also build a "hybrid" graph, where the graph is induced by the social graph but the edge weights are derived from the three rating similarity measures. Surprisingly, only about 29% of the edges in the social graph also appear in the co-rating graph, and taking the edge intersection of the two graphs results in a disconnected graph. This suggests that many users tend to import friends from other OSNs.

Even for a disconnected graph, we can still compute its $\overline{D}$ (note $\overline{H} = \infty$ in this case), and Table 4.2 shows the results for different graph-weight combinations. Our baseline $\overline{D}$ is that of "Yelp average," which corresponds to users relying only on Yelp-provided average ratings and not consulting any friends. Somewhat surprisingly, Yelp average achieves better $\overline{D}$ than all $\overline{D}$'s reported in Table 4.1, except for that of the social graph. This supports our claim that users form a social network to improve their local utility. Weighting edges of the

Table 4.1: Network metrics on connected networks. Results of "random" are the average over 10 random graph instances.

Network	$\overline{H}$	$\overline{D}$
Social, uniform	1.68×10^5	0.882
Random, uniform	1.47×10^5	0.890
Co-rating, uniform	7.28×10^5	0.994
Co-rating, Jaccard	1.03×10^6	0.993
Co-rating, Pearson	4.23×10^6	0.955
Co-rating, Cosine	7.30×10^5	0.994
Co-rating, Jaccard $\circ$ Pearson	6.15×10^6	0.961
Co-rating, Jaccard $\circ$ Cosine	1.03×10^6	0.992

Table 4.2: Local utility of disconnected networks.

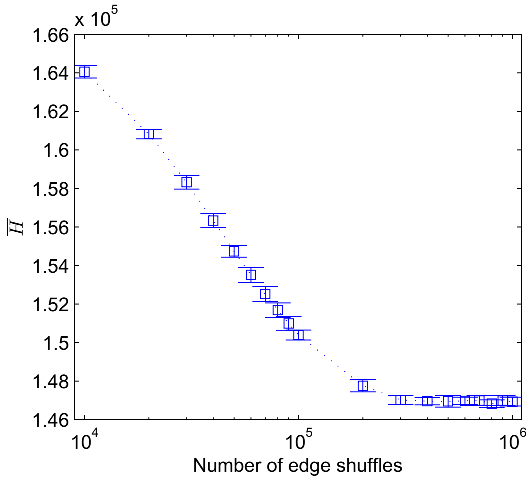
Network	$\overline{D}$	Network	$\overline{D}$
Yelp average (no network)	0.887	Social, Jaccard	0.875
Social, Jaccard $\circ$ Pearson	0.872	Social, Pearson	0.872
Social, Jaccard $\circ$ Cosine	0.875	Social, Cosine	0.876

social graph by Jaccard/Cosine/Pearson coefficients further improve $\overline{D}$. In Section 4.5.3 we consider a modification to weight social graph edges that preserve graph connectedness.

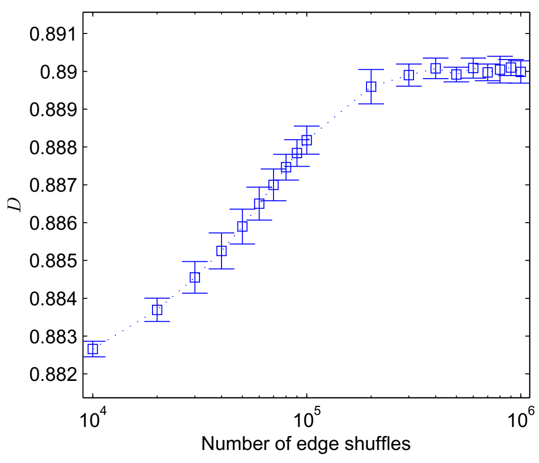
4.5.2 Varying Graph Randomness

The purpose of this and the next section is to study the tradeoff between global and local utilities. Consider interpolating the social graph and its random graph counterpart, with randomness quantified by the number of edge shuffles. Figures 4.2(a) and 4.2(b) show that as randomness increases, global utility improves but local utility degrades. This tradeoff is made evident in Figure 4.2(d), a scatterplot showing strong negative correlation between the two metrics.

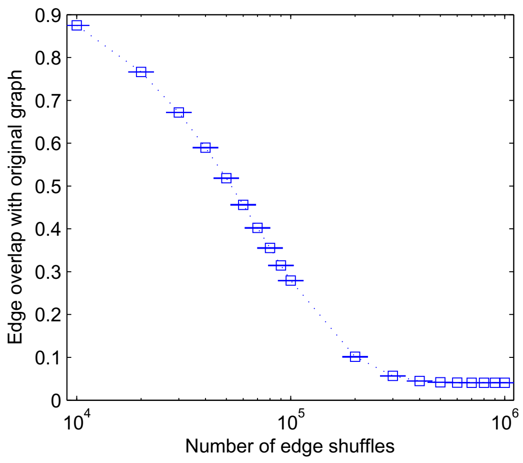
Empirical results [92] show that similar edge shuffling strategies produce a random graph with $O(m)$ shuffles, where m is the number of edges in the original graph. Since in Figure 4.2 the results stabilize at around 200,000 shuffles, we are convinced that any graph generated with more than 200,000 shuffles are close to random. Figure 4.2(c) further



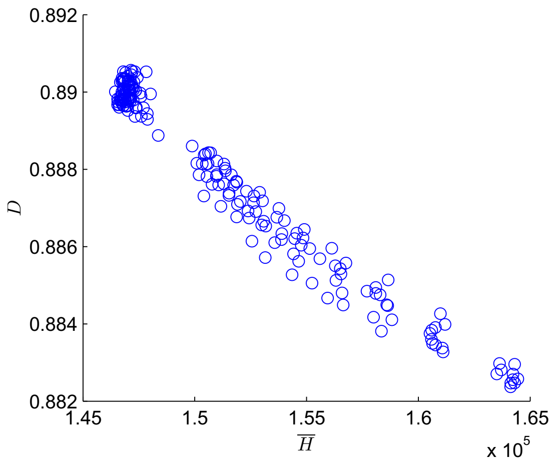
(a) More randomness improves $\overline{H}$.



(b) More randomness degrades $\overline{D}$.



(c) Similarity with original graph.



(d) Tradeoff between $\overline{H}$ and $\overline{D}$.

Figure 4.2: Results from edge-perturbing social graph. For each number of shuffles, 10 graphs are generated with 1 s.d. reported.

supports our claim by showing the proportion of edges that overlap with those of the social graph to be small (about 4%).

4.5.3 Varying Weight Skew

Here we consider an edge weighting strategy to study the tradeoff between $\overline{H}$ and $\overline{D}$. From Tables 4.1 and 4.2, it appears that weighting the social graph with Pearson coefficient will result in the best possible $\overline{H}$ and $\overline{D}$, so we apply the weight transformation parametrized by $\alpha, \beta > 0$:

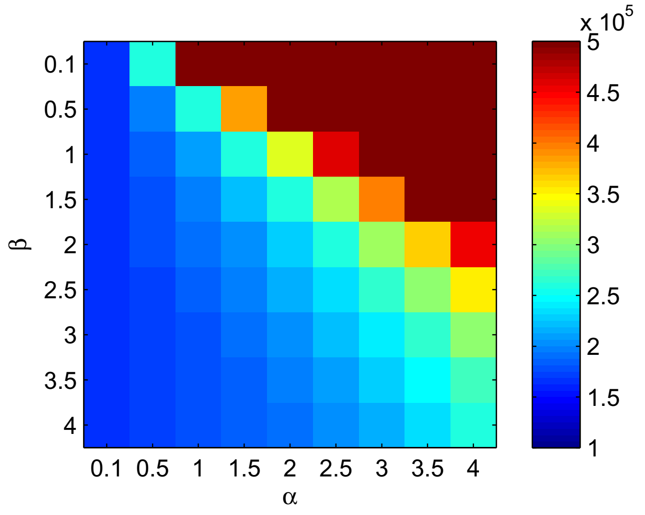
$$W_{i,j} = \begin{cases} (W_{i,j}^{\text{Pearson}} + \beta)^{\alpha} & W_{i,j}^{\text{social}} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Note that $W_{i,j}^{\text{Pearson}} = 0$ means users v_i and v_j have not co-rated anything, and setting $\beta > 0$ ensures the graph is connected. Weight skew, *i.e.,* the difference between the maximum and minimum nonzero edge weights, decreases with decreasing α or increasing β. In the limits $\alpha = 0$ or $\beta \to \infty$, the graph is equivalent to the social graph with uniform weights, *i.e.,* with zero weight skew.

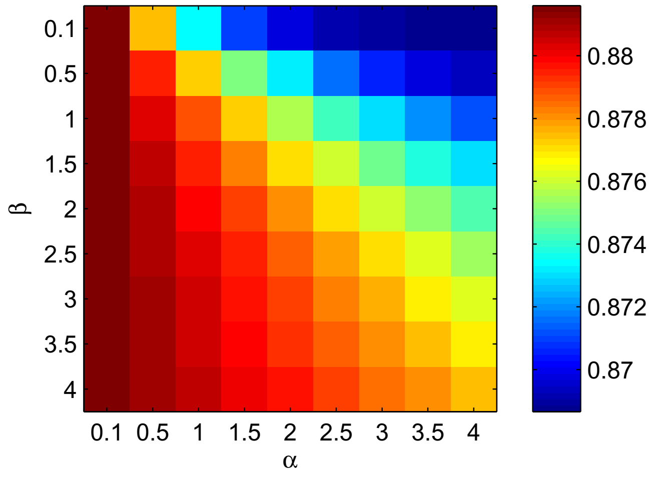
Figures 4.3(a) and 4.3(b) show the tradeoff between $\overline{H}$ and $\overline{D}$ by varying weight skew through (α, β): for (α, β) pairs that result in good $\overline{H}$ (blue areas), the corresponding $\overline{D}$ is bad (red areas), and vice versa. Figure 4.3(c) further illustrates the tradeoff with a scatter-plot.

4.5.4 Does Social Network Help in Collaborative Filtering?

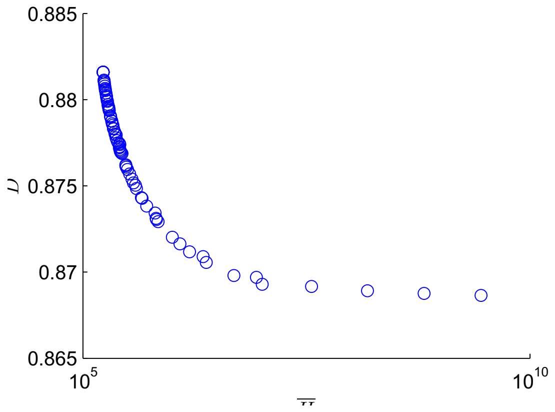
We first investigate the distribution of $\{D_i\}$ associated with Yelp's social graph (Figure 4.4(a)). One can see that most users have their D_i between 0 and 1, but there is also a considerable portion of outliers coming from the population with few friends (Figure 4.4(b)). While local utility improves as a user builds more connections, it does not prove



(a) Less weight skew improves $\overline{H}$.

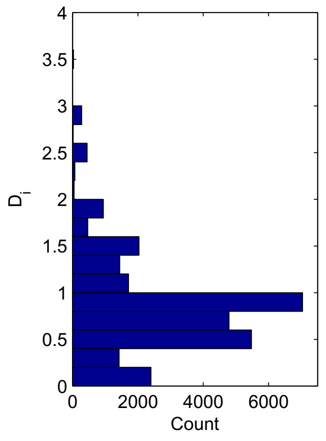


(b) More weight skew improves $\overline{D}$.

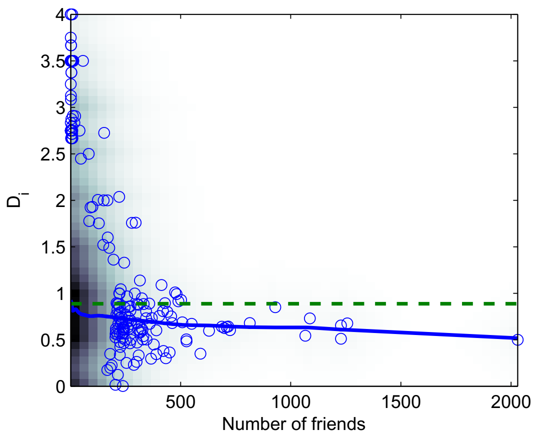


(c) Tradeoff between $\overline{H}$ and $\overline{D}$.

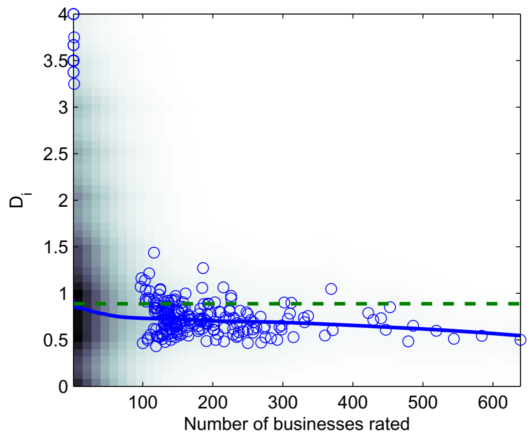
Figure 4.3: Effect of skewing edge weights by controlling (α, β) parameters.



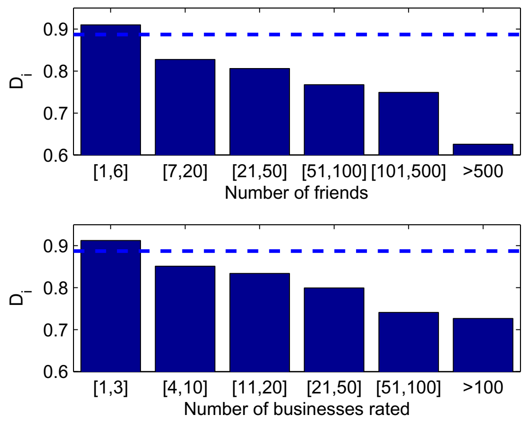
(a) Distribution of D_i.



(b) D_i decreases with degree.



(c) D_i decreases with review activity.



(d) D_i decreases with activity.

Figure 4.4: Distribution of D_i on Yelp social graph. For (b) and (c), 2D histograms are shown with 1% outliers overlaid. Dashed lines are the $\overline{D}$ baseline due to Yelp average, and solid lines are locally weighted linear regression fit of data.

that having more friends improves social recommendations, because it is possible that when a user is more active, she will both have more friends and have better D_i (*i.e.,* being active is a confounder of D_i and number of friends). Indeed, we notice that a user's D_i decreases as she produces more reviews (see Figure 4.4(c)). Also, the number of reviews one produces is positively correlated with the number of friends she has.

To eliminate the effect of confounding, we carry out statistical tests, enabled by our local utility metric, on populations with similar activity levels. First, we run a signed rank test on the two $\{D_i\}$ vectors of the full population due to the social graph and Yelp average. Although the social graph has lower $\overline{D}$, the test is inconclusive because it fails to reject at 5% level (p-value: 0.915) the null hypothesis that the two $\{D_i\}$ are equal. However, if we run the test on the $\{D_i\}$ vectors restricted to users with at least two friends, we successfully reject the null hypothesis (p-value: 6.07×10^{-3}). Thus, we can conclude that when a user makes more than one connection, she will see better recommendations from her friends.

4.6 Social Recommender Network Optimization

4.6.1 Optimization Problem

Even though the Yelp social network performs well in balancing global and local utilities, it is possible to do better through optimization. If a service provider knows what the optimal network is, it can facilitate network formation with strategies like friend recommendation and news feed curation (*e.g.,* promoting the updates of more "useful" friends). Here our goal is to find the optimal edge weights W^*, so that we recommend users v_i and v_j to become friends if $W_{i,j} = 0$ but $W^*_{i,j} > 0$, and news feed curation is equivalent to scaling the frequency of updates by W^*.

Consider the scenario where each user v_i has a constraint on her local utility $D_i \leq \delta_i$, where δ_i is the worst tolerable utility as observed from existing data, *e.g.,* D_i measured on the given social graph. In addition, we are given an allowable edge set E with $|E| = m$.

Then the social recommender network optimization problem becomes maximizing network efficiency, subject to topology and local utility constraints:

$$\begin{aligned} \underset{W\in\mathbb{R}^{n\times n}}{\text{minimize}} \quad & \overline{H}(W) \\ \text{subject to} \quad & D_i(W) \le \delta_i && i = 1, 2, \ldots, n \\ & W_{i,j} = 0 && \text{for } \{i,j\} \notin E \\ & W = W^T \\ & W \ge 0. && (4.10) \end{aligned}$$

Since both $\overline{H}$ and $\{D_i\}$ are invariant to scaling, *i.e.,* W and αW (for any $\alpha > 0$) result in the same $\overline{H}$ and $\{D_i\}$, problem (4.10) can be rewritten in terms of effective resistance (see Section 4.3.2 for a discussion):

$$\begin{aligned} \underset{W\in\mathbb{R}^{n\times n}}{\text{minimize}} \quad & R_{\text{tot}}(W) \\ \text{subject to} \quad & D_i(W) \le \delta_i && i = 1, 2, \ldots, n \\ & W_{i,j} = 0 && \text{for } \{i,j\} \notin E \\ & W = W^T \\ & W \ge 0 \\ & \sum_{i<j} W_{i,j} = m, && (4.11) \end{aligned}$$

where we add the edge weight sum constraint $\sum_{i<j} W_{i,j} = m$; otherwise one can set W arbitrarily large to make R_{tot} small. Then we can show the following.

Lemma 4.6.1. *Optimization problem* (4.11) *is convex.*

Proof. We need to verify the convexity of the objective function and the constraints. R_{tot} has been shown to be convex in [39], and the constraints $W = W^T$, $W \ge 0$ and $\sum_{i<j} W_{i,j} = m$ are affine. Hence it remains to show $D_i \le \delta_i$ is convex. By (4.5) and

rearranging, $D_i \leq \delta_i$ is equivalent to

$$\sum_{j=1}^{n} W_{i,j}\bigg(\sum_{k\in\mathcal{R}_i} |\tilde{R}_{j,k} - R_{i,k}| - \delta_i|\mathcal{R}_i|\bigg) \leq 0,$$

which is another affine constraint. $\square$

4.6.2 Algorithm

Our problem of finding optimal edge weights is solved on networks with at least 100,000 edges. Off-the-shelf tools based on interior point and semidefinite programming methods are prohibitively expensive on problems of this size. Thus we opt for a projected gradient method instead. In the following, it is more convenient to express the variables as the vector $w = \{w_l\}$, for all $l \in E$. Let $E_i = \{l \mid l = \{i,j\} \text{ any } j\}$, then we rewrite problem (2.10) as

$$\begin{aligned} \underset{w\in\mathbb{R}^m}{\text{minimize}} \quad & R_{\text{tot}}(w) \\ \text{subject to} \quad & w \geq 0 \\ & \mathbf{1}^T w = m \\ & \sum_{l=\{i,j\}\in E_i} w_l\bigg(\sum_{k\in\mathcal{R}_i} |\tilde{R}_{j,k} - R_{i,k}| - \delta_i|\mathcal{R}_i|\bigg) \leq 0. \end{aligned} \tag{4.12}$$

Deriving the optimization algorithm involves two steps.

Deriving the gradient. From [39] we have for $l = \{i,j\}$,

$$\begin{aligned} \frac{\partial R_{\text{tot}}}{\partial w_l} &= -n\left\|\Big(L + \frac{1}{n}\mathbf{1}\mathbf{1}^T\Big)^{-1} a_l\right\|_2^2 \\ &= -n\left\|L^\dagger a_l + \frac{1}{n}\mathbf{1}\mathbf{1}^T a_l\right\|_2^2 \\ &= -n\left\|L^\dagger(e_i - e_j) + \frac{1}{n}(e_i - e_j)\right\|_2^2, \end{aligned} \tag{4.13}$$

where $L^\dagger$ is the pseudoinverse of L, a_l is the vector with $(a_l)_i = 1$, $(a_l)_j = -1$, $(a_l)_k = 0$ for $k \neq i, j$, and e_i is the vector with $(e_i)_i = 1$ and $(e_i)_k = 0$ for $k \neq i$. The second step is due to $(L + \mathbf{1}\mathbf{1}^T/n)^{-1} = L^\dagger + \mathbf{1}\mathbf{1}^T/n$ (Eq. (7) of [39]).

Naively computing the pseudoinverse $L^\dagger$ is prohibitively expensive with time complexity $O(n^3)$, but there exist efficient Laplacian solvers (which solves for x in the system $Lx = b$), and we use the solver in [60] to compute $L^\dagger e_i$. The gradient vector ∇R_{tot} can be computed in $\tilde{O}(mn)$ time.

Deriving projections. Projecting w to satisfy the constraints can be cast as a linearly-constrained quadratic program. Define $w_i \in \mathbb{R}^{|E_i|}$ as the vector $\{w_l\}$ for $l \in E_i$, and $b_i \in \mathbb{R}^{|E_i|}$ such that $(b_i)_l = \sum_{k \in \mathcal{R}_i} |\tilde{R}_{j,k} - R_{i,k}| - \delta_i |\mathcal{R}_i|$ for $l = \{i, j\}$. The last constraint can be written as $b_i^T w_i \leq 0$. Then computing the projection is equivalent to solving

$$\begin{aligned} \underset{x \in \mathbb{R}^m}{\text{minimize}} \quad & \|x - w\|_2^2 \\ \text{subject to} \quad & x \geq 0 \\ & \mathbf{1}^T x = m \\ & b_i^T x_i \leq 0 \qquad\qquad i = 1, \ldots, n, \end{aligned} \tag{4.14}$$

where x_i is defined with the same indexing scheme as w_i. We use Gurobi [45] to solve this quadratic program. Algorithm 3 summarizes our solution approach to problem (4.12).

Algorithm 3 Projected gradient descent to solve (4.12).

```
Input: $G = \{V, E\}, R, \tilde{R}, \delta, t$
  Initialize: $w \leftarrow \mathbf{1}$
  repeat
    $w \leftarrow w - t \cdot \nabla R_{\text{tot}}$ with $\nabla R_{\text{tot}}$ as calculated by (4.13)
    $w \leftarrow x^*$, where $x^*$ is solution to problem (4.14)
  until $w$ converges
```

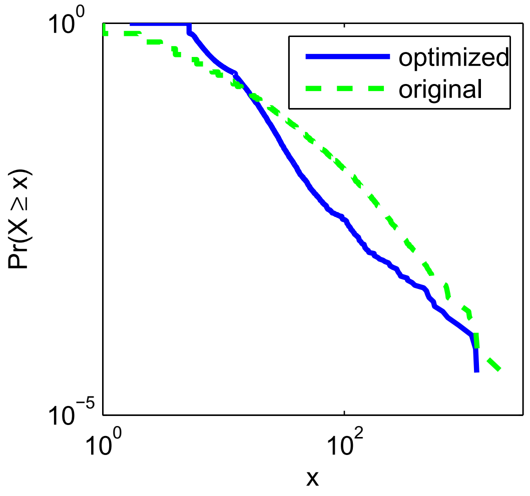
4.6.3 Results

We implement our algorithm on a machine with two Intel E5540 processors and 16GB RAM. Then we input E as the edges of the Yelp social graph to study the how the algorithm modifies the graph with uniform edge weights. While not studied in this chapter, we note that friend recommendation can easily be incorporated by expanding the allowable edge set E. Compared to the graph with uniform edge weights, we see substantial improvement in $\overline{H}$, *i.e.,* the optimal graph achieves $\overline{H} = 5.78 \times 10^4$, a 65% reduction in $\overline{H}$, and slightly better $\overline{D} = 0.879$. The hitting time of this optimized graph is also substantially better than that of random graphs (with uniform edge weights).

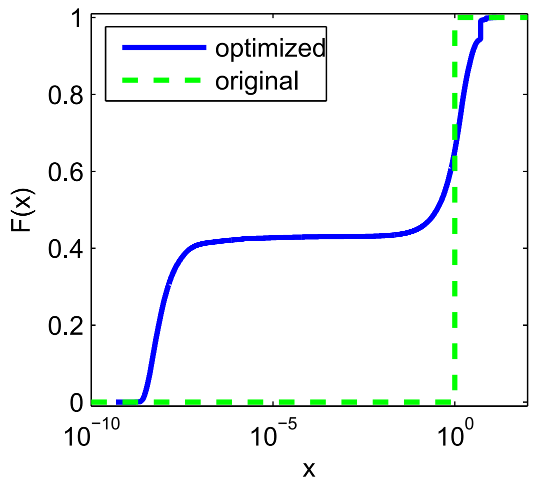
Statistics of optimized graph. We are interested in what the optimized graph looks like. First, we check whether the node degrees still follow a power law distribution. Figure 4.5(a) shows the log-log plot of the degree distribution. We notice that the key difference between the optimized graph and a standard social network is that there are very few nodes with degree ≤ 5. In general, it has been known that having too many small-degree nodes will increase the mixing time of a graph. Our optimization procedure recognizes this problem and avoids the presence of small-degree nodes as much as possible. Nodes with degree ≥ 5 have their degrees follow a power law distribution well.

Second, we would also like to understand the distribution of edge weights (see Figure 4.5(b)). We see a double phase transition effect, *i.e.,* approximately half of the weights concentrate at 10^{-7} (which is essentially 0) and 35% of the edges with a weight of above 1 serve as the backbone that connects 99.6% of the network. The maximum edge weight is 14.2. Thus, it appears that edges with weights below 1 are not needed in the optimal solution.

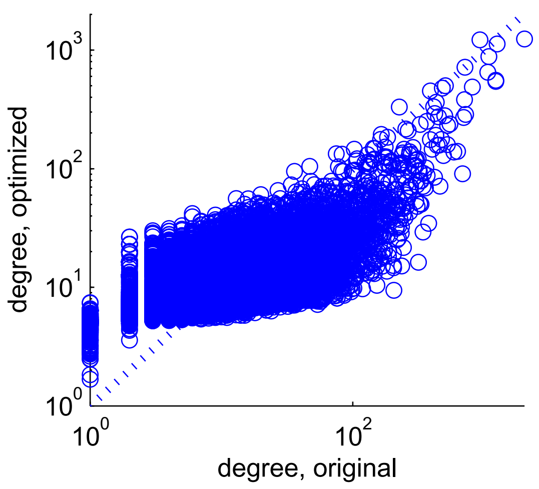
Correlation between original and optimized graph. Next, we study the correlation between the original and the optimized graph. Figure 4.5(c) shows the degree correlation. While the node degrees of both graphs are correlated in general, many nodes have substantial changes in weights. This shows global surgery is needed to optimize the original graph.



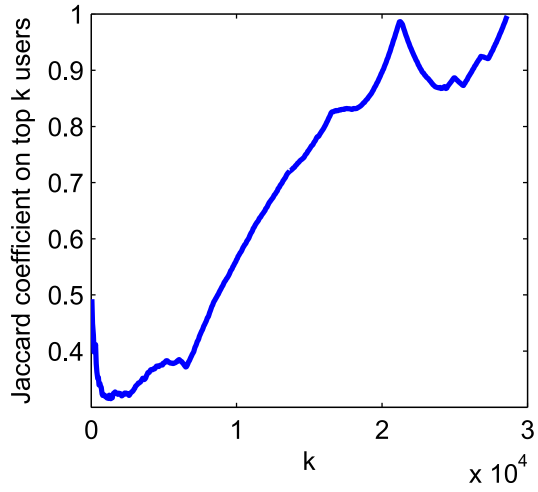
(a) Degree distribution.



(b) CDF of edge weights.



(c) Optimization makes node degrees more equal.



(d) Jaccard coefficient on top k user sets of uniform and optimal social graphs.

Figure 4.5: Results from optimizing edge weights of Yelp social graph.

Second, we also compute the Jaccard coefficients of the k highest degree nodes for all k. We can see that for k below 10^4 (recall there are 28,647 nodes), the Jaccard coefficients are consistently below 0.5. This implies the overlap between the "important" nodes of the two graphs is not substantial.

4.7 Related Work

Our global utility metric based on average hitting time is related to proposals of using effective resistance [27] and related measures [89, 81] to quantify network robustness. These works implicitly assume a network routing model for information dissemination, so redundant paths are beneficial to network robustness, but this may not be the case for our hitting time-based global utility metric.

Empirical analyses of the utility of OSNs have traditionally focused on global properties like mixing time [74] and information diffusion [96], but some recent works have also addressed local utility. Mittal et al. [72] proposed a random walk-based metric to study the tradeoff between privacy and local utility in the context of privacy-preserving graph perturbation. Zadeh et al. [102] showed high precision and fast dissemination can simultaneously be achieved in some broadcast networks. May et al. [68] studied content curation in social media at a microscopic level. Ma [63] studied the correlation between social relationships and user interest similarity. Our work takes an extra step in designing algorithms for network formation that jointly optimizes local and global utilities.

Network topology optimization with second largest eigenvalue modulus [13] or effective resistance [39] have found applications in, *e.g.,* computer networks [40] and power networks [48]. Our contribution is in applying the techniques on OSNs with constraints on a novel local utility metric. We also remark that the scale of our problem ($>$100,000 edges) is much larger than those seen in existing work ($\sim$1,000 edges), and this poses unique computational challenges.

4.8 Conclusions

Motivated by a stochastic model for diffusion of recommendations, we propose a set of metrics to measure the local and global utilities of a social recommender network. We compute these metrics on a recently released dataset and make a number of interesting discoveries: (a) at the individual level, users are good at identifying connections that can help to improve the quality of recommendations they receive. So long as a user has more than one connection, the improvement is statistically significant, and (b) at the macroscopic level, the average hitting time for the social graph is only moderately worse than randomly generated graphs, while the latter ones have been known to be efficient in disseminating information. We also examine how we can construct social networks that have local and global utilities jointly optimized. While our optimization problem is convex, it is a large-scale problem that prompts for a hybrid solution blending in-house developed algorithmic techniques with optimization technologies for sparse matrices and quadratic programs. In general, we believe a better understanding the tradeoff between user experience and network efficiency in information propagation is an important research direction in recommender systems research.

4.A Proof of Proposition 4.3.1

Proof. First, notice that for any b, the path $\{p_t(b)\}_{t\geq 0}$ is a random walk over the undirected but weighted graph induced by W. Specifically, the transition probabilities are defined as

$$\Pr[p_t(b) = v_j \mid p_{t-1}(b) = v_i] = \frac{W_{i,j}}{\sum_j W_{i,j}}.$$

As we are interested in the conditional expectation when $t \to \infty$, we may safely assume that $p_0(b)$ is sampled from the stationary distribution π of the Markov chain. Furthermore, as one can see the Markov chain is time-reversible, we have

$$\Pr[p_t(b) = v_j \wedge p_{t-1}(b) = v_i] = \Pr[p_t(b) = v_j \wedge p_{t+1}(b) = v_i].$$

We need more notation. Let $X_{k,t}(v_i)$ be the indicator random variable that sets to 1 if and only if $p_t(b_k) = v_i$ (*i.e.,* v_i visits the business b_k at time t). When v_i is clear from the context, we write $X_{k,t}$ directly. Also, let $C_t = \sum_k X_{k,t}$. We have

$$\begin{aligned}
& \mathrm{E}\left[\frac{1}{|e_t(v_i)|}\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\middle|\, e_t(v_i) \neq \emptyset\right] \\
= & \sum_{d=1}^{\ell} \mathrm{E}\left[\frac{1}{d}\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\middle|\, C_t = d,\ e_t(v_i) \neq \emptyset\right] \cdot \Pr[C_t = d \mid e_t(v_i) \neq \emptyset].
\end{aligned}$$

Notice that $C_t \geq 1$ implies $e_t(v_i) \neq \emptyset$ and thus $(C_t = d, e_t(v_i) \neq \emptyset)$ is the same as $C_t = d$.

For exposition purposes, let us consider the summands in the above equation for $d = 1$. Then we will derive the equations for general $d \geq 1$. For $d = 1$, we are interested in computing $\Pr[C_t = 1 \mid e_t(v_i) \neq \emptyset]$ and $\mathrm{E}\big[\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\big|\, C_t = 1\big]$.

First, notice that

$$\begin{aligned}\Pr[C_t = 1 \mid e_t(v_i) \neq \emptyset] &= \frac{\Pr[C_t = 1]}{\Pr[e_t(v_i) \neq \emptyset]} \\ &= \frac{\binom{\ell}{1}\pi_i(1-\pi_i)^{\ell-1}}{1-(1-\pi_i)^{\ell}},\end{aligned}$$

where π_i is the probability mass at v_i at the stationary distribution vector π.

Let us define $E_{k,t}$ as the event that v_i makes exactly one review at time t and the review is on business b_k, *i.e.,* the event $e_t(v_i) = \{b_k\}$. One can see that $E_{k,t}$ are mutually exclusive on k, and the event $C_t = 1$ is equivalent to $\vee_{k\leq\ell} E_{k,t}$. Thus,

$$\begin{aligned}&\mathrm{E}\Big[\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \;\Big|\; C_t = 1\Big] \\ =\;& \sum_{k\leq\ell} \mathrm{E}\Big[\sum_{b\in e_t(v_i)} |R_b^t - R_b^{t-1}| \;\Big|\; C_t = 1,\ E_{k,t} = 1\Big] \cdot \Pr[E_{k,t} = 1 \mid C_t = 1] \\ =\;& \sum_{k\leq\ell} \mathrm{E}\big[|R_{b_k}^t - R_{b_k}^{t-1}| \;\big|\; E_{k,t} = 1\big] \cdot \Pr[E_{k,t} \mid C_t = 1].\end{aligned}$$

Now we use the time-reversibility property to compute:

$$\begin{aligned}&\mathrm{E}\big[|R_{b_k}^t - R_{b_k}^{t-1}| \;\big|\; E_{k,t} = 1\big] \\ =\;& \sum_j \Pr[p_{t-1}(b_k) = v_j \mid p_t(b_k) = v_i] \cdot |R_{j,k} - R_{i,k}| \\ =\;& \sum_j \frac{\Pr[p_{t-1}(b_k) = v_j \wedge p_t(b_k) = v_i]}{\Pr[p_t(b_k) = v_i]} \cdot |R_{j,k} - R_{i,k}| \\ =\;& \sum_j \frac{\Pr[p_{t+1}(b_k) = v_j \wedge p_t(b_k) = v_i]}{\Pr[p_t(b_k) = v_i]} \cdot |R_{j,k} - R_{i,k}| \\ &\qquad\qquad\qquad\qquad\text{(Markov chain is time-reversible)} \\ =\;& \sum_j \Pr[p_{t+1}(b_k) = v_j \mid p_t(b_k) = v_i] \cdot |R_{j,k} - R_{i,k}| \\ =\;& \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}}.\end{aligned}$$

Also,

$$\Pr[E_{k,t} \mid C_t = 1] = \frac{\Pr[E_{k,t}]}{\Pr[C_t = 1]} = \frac{\pi_i(1-\pi_i)^{\ell-1}}{\binom{\ell}{1}\pi_i(1-\pi_i)^{\ell-1}} = \frac{1}{\binom{\ell}{1}}.$$

Thus, we conclude that

$$\begin{aligned}
&\mathrm{E}\Bigg[\sum_{b \in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\Bigg|\, C_t = 1\Bigg] \cdot \Pr[C_t = 1 \mid e_t(v_i) \neq \emptyset] \\
&= \frac{\pi_i(1-\pi_i)^{\ell-1}}{1-(1-\pi_i)^{\ell}} \sum_{k \le \ell} \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}}.
\end{aligned}$$

Extending the analysis to fixed $d \geq 1$, let us first define $E_{\{k_1,\ldots,k_d\},t}$ as the event $e_t(v_i) = \{b_{k_1}, \ldots, b_{k_d}\}$. Note that all $E_{\{k_1,\ldots,k_d\},t}$ are mutually exclusive, where $\{k_1, \ldots, k_d\}$ is a size-d combination from $\{1, \ldots, \ell\}$, and the event $C_t = d$ is equivalent to $\vee_{\{k_1,\ldots,k_d\}} E_{\{k_1,\ldots,k_d\},t}$.

Then,

$$\begin{aligned}
&\mathrm{E}\Bigg[\sum_{b \in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\Bigg|\, C_t = d\Bigg] \\
&= \sum_{\{k_1,\ldots,k_d\}} \mathrm{E}\Bigg[\sum_{b \in e_t(v_i)} |R_b^t - R_b^{t-1}| \,\Bigg|\, E_{\{k_1,\ldots,k_d\},t}\Bigg] \cdot \Pr[E_{\{k_1,\ldots,k_d\},t} \mid C_t = d] \\
&= \sum_{\{k_1,\ldots,k_d\}} \mathrm{E}\Bigg[\sum_{k \in \{k_1,\ldots,k_d\}} |R_{b_k}^t - R_{b_k}^{t-1}| \,\Bigg|\, E_{\{k_1,\ldots,k_d\},t}\Bigg] \cdot \Pr[E_{\{k_1,\ldots,k_d\},t} \mid C_t = d] \\
&= \sum_{\{k_1,\ldots,k_d\}} \sum_{k \in \{k_1,\ldots,k_d\}} \mathrm{E}\big[|R_{b_k}^t - R_{b_k}^{t-1}| \mid E_{\{k_1,\ldots,k_d\},t}\big] \cdot \Pr[E_{\{k_1,\ldots,k_d\},t} \mid C_t = d],
\end{aligned}$$

where

$$\begin{aligned}
\Pr[E_{\{k_1,\ldots,k_d\},t} \mid C_t = d] &= \frac{\Pr[E_{\{k_1,\ldots,k_d\},t}]}{\Pr[C_t = d]} \\
&= \frac{\pi_i^d(1-\pi_i)^{\ell-d}}{\binom{\ell}{d}\pi_i^d(1-\pi_i)^{\ell-d}} = \frac{1}{\binom{\ell}{d}}
\end{aligned}$$

and

$$\mathrm{E}\big[|R^t_{b_k} - R^{t-1}_{b_k}| \mid E_{\{k_1,\ldots,k_d\},t}\big] = \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}}.$$

Thus, for general $d \geq 1$,

$$\mathrm{E}\Bigg[\sum_{b \in e_t(v_i)} |R^t_b - R^{t-1}_b| \,\Bigg|\, C_t = d\Bigg] = \sum_{\{k_1,\ldots,k_d\}} \sum_{k \in \{k_1,\ldots,k_d\}} \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}} \frac{1}{\binom{\ell}{d}}.$$

We note that the double summation can be rewritten as

$$\sum_{\{k_1,\ldots,k_d\}} \sum_{k \in \{k_1,\ldots,k_d\}} \equiv \sum_{k=1}^{\ell} \sum_{\substack{\{k_1,\ldots,k_d\} \text{ s.t.} \\ k \in \{k_1,\ldots,k_d\}}}$$

and since the summands are independent of $k_1, \ldots, k_d$ given fixed k, the inner summation is simply the number of combinations $\{k_1, \ldots, k_d\}$ with k inside, which is just $\binom{\ell-1}{d-1}$. Thus we have

$$\mathrm{E}\Bigg[\sum_{b \in e_t(v_i)} |R^t_b - R^{t-1}_b| \,\Bigg|\, C_t = d\Bigg] = \sum_{k=1}^{\ell} \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,k}|}{\sum_j W_{i,j}} \frac{\binom{\ell-1}{d-1}}{\binom{\ell}{d}}.$$

and

$$\begin{aligned}
&\mathrm{E}\Bigg[\frac{1}{d} \sum_{b \in e_t(v_i)} |R^t_b - R^{t-1}_b| \,\Bigg|\, C_t = d\Bigg] \cdot \Pr[C_t = d \mid e_t(v_i) \neq \emptyset] \\
= \;& \frac{1}{d} \sum_k \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,j}|}{\sum_j W_{i,j}} \frac{\binom{\ell-1}{d-1}}{\binom{\ell}{d}} \cdot \frac{\binom{\ell}{d}\pi_i^d (1-\pi_i)^{\ell-d}}{1-(1-\pi_i)^\ell} \\
= \;& \frac{\binom{\ell-1}{d-1}}{d} \frac{\pi_i^d (1-\pi_i)^{\ell-d}}{1-(1-\pi_i)^\ell} \sum_k \sum_j \frac{W_{i,j}|R_{j,k} - R_{i,j}|}{\sum_j W_{i,j}}.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
&\mathrm{E}\Big[\frac{1}{|e_t(v_i)|}\sum_{b\in e_t(v_i)}|R_b^t - R_b^{t-1}|\;\Big|\; e_t(v_i)\neq\emptyset\Big]\\
= &\left(\sum_{d=1}^{\ell}\binom{\ell-1}{d-1}\frac{\ell}{d}\cdot\frac{\pi_i^d(1-\pi_i)^{\ell-d}}{1-(1-\pi_i)^{\ell}}\right)\left(\frac{1}{\ell}\sum_k\sum_j\frac{W_{i,j}|R_{j,k}-R_{i,k}|}{\sum_j W_{i,j}}\right)\\
= &\frac{\sum_{d=1}^{\ell}\binom{\ell}{d}\pi_i^d(1-\pi_i)^{\ell-d}}{1-(1-\pi_i)^{\ell}}\left(\frac{1}{\ell}\sum_k\sum_j\frac{W_{i,j}|R_{j,k}-R_{i,k}|}{\sum_j W_{i,j}}\right),
\end{aligned}$$

but since

$$\sum_{d=0}^{\ell}\binom{\ell}{d}\pi_i^d(1-\pi_i)^{\ell-d} = (\pi_i+(1-\pi_i))^{\ell} = 1,$$

we have

$$\sum_{d=1}^{\ell}\binom{\ell}{d}\pi_i^d(1-\pi_i)^{\ell-d} = 1-\binom{\ell}{0}\pi^0(1-\pi_i)^{\ell-0} = 1-(1-\pi_i)^{\ell}$$

or equivalently,

$$\frac{\sum_{d=1}^{\ell}\binom{\ell}{d}\pi_i^d(1-\pi_i)^{\ell-d}}{(1-(1-\pi_i))^{\ell}} = 1,$$

which we substitute into the above product to obtain our result.

□

Chapter 5

Conclusions

Optimization has proven to be useful in many fields of electrical engineering, including signal processing and communication networks [12, 17]. This dissertation attempts to extend the mentality of optimization to data analytics. While certainly not the only approach, devising data analytics techniques as formulating optimization problems allows us to reason about various issues in a principled manner: variables represent what we want to model, modeling assumptions are embodied in the objective function, and prior knowledge can be incorporated as regularization terms in the objective function, or as constraints to the optimization problem. Taking such a view on optimization in the field of communication networks, *i.e.,* treating optimization as a *language* rather than merely a tool, has led to new insights into existing engineered systems and improvements to them [17]. Our long-term goal is to develop a similar theory for data analytics.

The key to the success of applying optimization techniques to data analytics lies in formulating, as tailored for the application context, the right models and optimization problems. Taking a joint statistical and algorithmic perspective [65], the "right" model should strike a balance between (a) statistical complexity, *i.e.,* how well a model fits reality, and (b) algorithmic complexity, *i.e.,* the efficiency of the resultant inference, prediction or recommendation algorithm. It appears that our proposed models are able to achieve this balance.

On one hand, our models are less detailed than many of those in the machine learning literature [76], but our algorithms based on convex optimization are efficient, while inference on many probabilistic generative models is intractable and one has to rely on approximate inference techniques. Thus we are able to achieve better algorithmic complexity at a slight loss in statistical complexity. On the other hand, when compared to basic regression-based models, our models are slightly more complex (and accurate), but our algorithms are also slightly less efficient. Thus we are also trading algorithmic for statistical complexity. It is interesting to see whether such a tradeoff is indeed necessary.

Bibliography

[1] M. Abdullah, C. Cooper, and M. Draief. Viral processes by random walks on random regular graphs. In *Proc. APPROX-RANDOM*, 2011.

[2] L. A. Adamic and N. Glance. The political blogosphere and the 2004 U.S. election: Divided they blog. In *Proc. LinkKDD*, 2005.

[3] A. Agarwal, E. Hazan, S. Kale, and R. E. Schapire. Algorithms for portfolio management based on the Newton method. In *Proc. ICML*, 2006.

[4] F. Al Zamal, W. Liu, and D. Ruths. Homophily and latent attribute inference: Inferring latent attributes of Twitter users from neighbors. In *Proc. ICWSM*, 2012.

[5] Alexa. The top 500 sites on the web. http://www.alexa.com/topsites, 2015 [January 2, 2015].

[6] J. An, M. Cha, K. P. Gummadi, J. Crowcroft, and D. Quercia. Visualizing media bias through Twitter. In *Proc. ICWSM SocMedNews Workshop*, 2012.

[7] S. Ansolabehere, R. Lessem, and J. M. Snyder. The orientation of newspaper endorsements in U.S. elections. *Quarterly Journal of Political Science*, 1(4):393–404, 2006.

[8] P. Barberá. Birds of the same feather tweet together: Bayesian ideal point estimation using Twitter data. *Political Analysis*, 2014.

[9] B. Bollobás. *Random graphs*. Academic Press, 1985.

[10] A. Borodin, R. El-Yaniv, and V. Gogan. Can we learn to beat the best stock. *Journal of Artificial Intelligence Research*, 21(1):579–594, 2004.

[11] A. Boutet, H. Kim, and E. Yoneki. What's in your tweets? I know who you supported in the UK 2010 general election. In *Proc. ICWSM*, 2012.

[12] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[13] S. Boyd, P. Diaconis, and L. Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46(4):667–689, 2004.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[15] W. S. Chan. Stock price reaction to news and no-news: drift and reversal after headlines. *Journal of Financial Economics*, 70:223–260, 2003.

[16] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, and Y. Yuan. Influence maximization in social networks when negative opinions may emerge and propagate. In *Proc. SDM*, 2011.

[17] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, 2007.

[18] F. Chierichetti, S. Lattanzi, and A. Panconesi. Almost tight bounds for rumour spreading with conductance. In *Proc. STOC*, 2010.

[19] A. Clementi, R. Silvestri, and L. Trevisan. Information spreading in dynamic graphs. In *Proc. PODC*, 2012.

[20] J. Clinton, S. Jackman, and D. Rivers. The statistical analysis of roll call data. *American Political Science Review*, 98(2):355–370, 2004.

[21] R. Cohen and D. Ruths. Classifying political orientation on Twitter: It's not easy! In *Proc. ICWSM*, 2013.

[22] M. D. Conover, B. Gonçalves, J. Ratkiewicz, A. Flammini, and F. Menczer. Predicting the political alignment of Twitter users. In *Proc. IEEE SocialCom*, 2011.

[23] M. D. Conover, J. Ratkiewicz, M. Francisco, B. Gonçalves, A. Flammini, and F. Menczer. Political polarization on Twitter. In *Proc. ICWSM*, 2011.

[24] T. M. Cover. Universal portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

[25] C. Dougal, J. Engelberg, D. García, and C. A. Parsons. Journalists and the stock market. *The Review of Financial Studies*, 25(3):439–479, 2012.

[26] G. Doyle and C. Elkan. Financial topic models. In *Proc. NIPS Workshop on Applications for Topic Models: Text and Beyond*, 2009.

[27] W. Ellens, F. Spieksma, P. Van Mieghem, A. Jamakovic, and R. Kooij. Effective graph resistance. *Linear Algebra and its Applications*, 435(10):2491–2506, 2011.

[28] E. F. Fama. Market efficienty, long-term returns, and behavioral finance. *Journal of Financial Economics*, 49(3):283–306, 1998.

[29] M. Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633, 1975.

[30] S. Finn, E. Mustafaraj, and P. T. Metaxas. The co-retweeted network and its applications for measuring the perceived political polarization. In *Proc. WEBIST*, 2014.

[31] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[32] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. Technical report, Stanford University, 2010.

[33] G. P. C. Fung, J. X. Yu, and W. Lam. News sensitive stock trend prediction. In *Proc. PAKDD*, 2002.

[34] M. Gabielkov, A. Rao, and A. Legout. Studying social networks at scale: Macroscopic anatomy of the Twitter social graph. In *Proc. SIGMETRICS*, 2014.

[35] G. Ganeshapillai, J. Guttag, and A. W. Lo. Learning connections in financial time series. In *ICML*, 2013.

[36] M. Gentzkow and J. M. Shapiro. What drives media slant? Evidence from U.S. daily newspapers. *Econometrica*, 78(1):35–71, January 2010.

[37] S. Gerrish and D. Blei. Predicting legislative roll calls from text. In *Proc. ICML*, 2011.

[38] S. Gerrish and D. Blei. How the vote: Issue-adjusted models of legislative behavior. In *Proc. NIPS*, 2012.

[39] A. Ghosh, S. Boyd, and A. Saberi. Minimizing effective resistance of a graph. *SIAM Review*, 50(1):37–66, 2008.

[40] C. Gkantsidis, G. Goel, M. Mihail, and A. Saberi. Towards topology aware networks. In *Proc. INFOCOM*, 2007.

[41] J. Golbeck and D. Hansen. A method for computing political preference among Twitter followers. *Social Networks*, 36:177–184, 2014.

[42] G. H. Golub, S. Nash, and C. van Loan. A Hessenberg-Schur method for the problem $AX + XB = C$. *IEEE Transactions on Automatic Control*, 24(6):909–913, 1979.

[43] M. Granovetter. The strength of weak ties: A network theory revisited. *Sociological Theory*, 1:201–233, 1982.

[44] T. Groseclose and J. Milyo. A measure of media bias. *The Quarterly Journal of Economics*, 120(4):1191–1237, November 2005.

[45] I. Gurobi Optimization. Gurobi optimizer reference manual, 2014. URL `http://www.gurobi.com`.

[46] M. Hagenau, M. Liebmann, M. Hedwig, and D. Neumann. Automated news reading: Stock price prediction based on financial netws using context-specific features. In *Proc. HICSS*, 2012.

[47] D. E. Ho and K. M. Quinn. Measuring explicit political positions of media. *Quarterly Journal of Political Science*, 3(4):353–377, 2008.

[48] J. K. Johnson and M. Chertkov. A majorization-minimization approach to design of power transmission networks. In *Proc. CDC*, 2010.

[49] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proc. KDD*, 2003.

[50] A. S. King, F. J. Orlando, and D. B. Sparks. Ideological extremity and primary success: A social network approach. In *Proc. MPSA Conference*, 2011.

[51] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proc. STOC*, 2000.

[52] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):42–49, 2009.

[53] H. Lam, Z. Liu, M. Mitzenmacher, X. Sun, and Y. Wang. Information dissemination via random walks in d-dimensional space. In *Proc. SODA*, 2012.

[54] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[55] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan. Mining of concurrent text and time series. In *Proc. KDD-2000 Workshop on Text Mining*, 2000.

[56] D. Lazer, A. Pentland, L. Adamic, S. Aral, A.-L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. Van Alstyne. Life in the network: The coming age of computational social science. *Science*, 323(5915):721–723, 2009.

[57] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2008.

[58] Y.-R. Lin, J. P. Bagrow, and D. Lazer. More voices than ever? quantifying media bias in networks. In *Proc. ICWSM*, 2011.

[59] A. Livne, M. P. Simmons, E. Adar, and L. A. Adamic. The party is over here: Structure and content in the 2010 election. In *Proc. ICWSM*, 2011.

[60] O. E. Livne and A. Brandt. Lean algebraic multigrid (LAMG): Fast graph laplacian linear solver. *SIAM Journal on Scientific Computing*, 34(4):B499–B522, 2012.

[61] J. R. Lott and K. A. Hassett. Is newspaper coverage of economic events politically biased? Online, October 2004. http://dx.doi.org/10.2139/ssrn.588453.

[62] C. Lumezanu, N. Feamster, and H. Klein. #bias: Measuring the tweeting behavior of propagandists. In *Proc. ICWSM*, 2012.

[63] H. Ma. On measuring social friend interest similarities in recommender systems. In *Proc. SIGIR*, 2014.

[64] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *Proc. WSDM*, 2011.

[65] M. W. Mahoney. *Algorithmic and Statistical Perspectives on Large-Scale Data Analysis*, chapter 16. Combinatorial Scientific Computing. CRC Press, 2012.

[66] H. Mao, S. Counts, and J. Bollen. Predicting financial markets: Comparing survey, news, Twitter and search engine data. *preprint*, 2011.

[67] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[68] A. May, A. Chaintreau, N. Korula, and S. Lattanzi. Filter & follow: How social media foster content curation. In *Proc. SIGMETRICS*, 2014.

[69] Y. Mejova, P. Srinivasan, and B. Boynton. GOP primary season on twitter: "Popular" political sentiment in social media. In *Proc. WSDM*, 2013.

[70] P. T. Metaxas and E. Mustafaraj. Social media and the elections. *Science*, 338: 472–473, 2012.

[71] M. Minev, C. Schommer, and T. Grammatikos. News and stock markets: A survey on abnormal returns and prediction models. Technical report, University of Luxembourg, 2012.

[72] P. Mittal, C. Papamanthou, and D. Song. Preserving link privacy in social network based systems. In *Proc. NDSS*, 2013.

[73] M.-A. Mittermayer and G. F. Knolmayer. NewsCATS: A news categorization and trading system. In *Proc. ICDM*, 2006.

[74] A. Mohaisen, A. Yun, and Y. Kim. Measuring the mixing time of social graphs. In *Proc. IMC*, 2010.

[75] S. A. Munson, S. Y. Lee, and P. Resnick. Encouraging reading of diverse political viewpoints with a browser widget. In *Proc. ICWSM*, 2013.

[76] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[77] E. Mustafaraj, S. Finn, C. Whitlock, and P. T. Metaxas. Vocal minority versus silent majority: Discovering the opinions of the long tail. In *Proc. SocialCom/PASSAT*, 2011.

[78] S. Park, M. Ko, J. Kim, Y. Liu, and J. Song. The politics of comments: Predicting political orientation of news stories with commenters' sentiment patterns. In *Proc. CSCW*, 2011.

[79] M. Pennacchiotti and A.-M. Popescu. A machine learning approach to Twitter user classification. In *Proc. ICWSM*, 2011.

[80] K. T. Poole and H. Rosenthal. A spatial model for legislative roll call analysis. *American Journal of Political Science*, 29(2):357–384, 1985.

[81] G. Ranjan and Z.-L. Zhang. Geometry of complex networks and topological centrality. *Physica A: Statistical Mechanics and its Applications*, 392(17):3833–3845, 2013.

[82] R. P. Schumaker and H. Chen. Textual analysis of stock market prediction using breaking financial news: The AZFinText system. *ACM Transactions on Information Systems*, 27(2), 2009.

[83] E. Seneta. *Non-negative Matrices and Markov Chains*. Springer, 2007.

[84] P. Sprechmann, I. Ramírez, G. Sapiro, and Y. C. Eldar. C-HiLasso: A collaborative hierarchical sparse modeling framework. *IEEE Transactions on Signal Processing*, 59(9):4183–4198, 2011.

[85] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009.

[86] P. C. Tetlock. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168, 2007.

[87] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and K. A. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558, 2010.

[88] J. D. Thomas and K. Sycara. Integrating genetic algorithms and text learning for financial prediction. In *Proc. GECCO-2000 Workshop on Data Mining with Evolutionary Algorithms*, 2000.

[89] A. Tizghadam and A. Leon-Garcia. Autonomic traffic engineering for network robustness. *IEEE Journal on Selected Areas in Communications*, 28(1):39–50, 2010.

[90] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proc. ICWSM*, 2010.

[91] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[92] F. Viger and M. Latapy. Fast generation of random connected graphs with prescribed degrees. In *Proc. COCOON*, 2005.

[93] S. Volkova, G. Coppersmith, and B. Van Durme. Inferring user political preferences from streaming communications. In *Proc. ACL*, 2014.

[94] M. Wall. Big data: Are you ready for blast-off? *BBC News* (March 3, 2014).

[95] F. E. Walter, S. Battiston, and F. Schweitzer. A model of a trust-based recommendation system on a social network. *Autonomous Agents and Multi-Agent Systems*, 16 (1):57–74, 2008.

[96] D. Wang, H. Park, G. Xie, S. Moon, M.-A. Kaafar, and K. Salamatian. A genealogy of information spreading on microblogs: A Galton-Watson-based explicative model. In *Proc. INFOCOM*, 2013.

[97] J. Wang, W. X. Zhao, Y. He, and X. Li. Infer user interests via link structure regularization. *ACM Transactions on Intelligent Systems and Technology*, 5(2), 2014.

[98] I. Weber, V. R. K. Garimella, and E. Borra. Mining web query logs to analyze political issues. In *Proc. WebSci*, 2012.

[99] I. Weber, V. R. K. Garimella, and A. Teka. Political hashtag trends. In *Proc. ECIR*, 2013.

[100] F. M. F. Wong and P. Marbach. 'Who are your friends?'—A simple mechanism that achieves perfect network formation. In *Proc. INFOCOM*, 2011.

[101] B. Wüthrich, D. Permunetilleke, S. Leung, V. Cho, L. Zhang, and W. Lam. Daily prediction of major stock indices from textual WWW data. In *Proc. KDD*, 1998.

[102] R. B. Zadeh, A. Goel, K. Munagala, and A. Sharma. On the precision of social and information networks. In *Proc. COSN*, 2013.

[103] W. Zhang and S. Skiena. Trading strategies to exploit blog and news sentiment. In *Proc. ICWSM*, 2010.

[104] Y. Zhang. An alternating direction algorithm for nonnegative matrix factorization. Technical report, Rice University, 2010.

[105] D. X. Zhou, P. Resnick, and Q. Mei. Classifying the political leaning of news articles and users from user votes. In *Proc. ICWSM*, 2011.