Models and Methods for Dissemination of Information and Knowledge Online

Thesis approved by
the Department of Computer Science
Technische Universität Kaiserslautern
for the award of the Doctoral Degree
Doctor of Natural Sciences (Dr. rer. nat.)

to

Utkarsh Upadhyay

Date of Defense:	1 December 2021
Dean:	Prof. Dr. Jens Schmitt
Reviewer:	Prof. Dr. N. Ganguly
Reviewer:	Dr. M. Gomez Rodriguez
Reviewer:	Prof. Dr. R. Majumdar

DE-386

Summary

In the past, information and knowledge dissemination was relegated to the brick-and-mortar classrooms, newspapers, radio, and television. As these processes were simple and centralized, the models behind them were well understood and so were the empirical methods for optimizing them. In today's world, the internet and social media has become a powerful tool for information and knowledge dissemination: Wikipedia gets more than 1 million edits per day,[1] Stack Overflow has more than 17 million questions,[2] 25% of US population visits Yahoo! News for articles and discussions,[3] Twitter has more than 60 million active monthly users,[4] and Duolingo has 25 million users learning languages online.[5] These developments have introduced a paradigm shift in the process of dissemination of information and knowledge. Not only has the nature of the task moved from being centralized to decentralized, but the developments have also blurred the boundary between the *creator* and the *consumer* of the *content*, *i.e.*, information and knowledge. These changes have made it necessary to develop new models, which are better suited to understanding and analysing the dissemination of knowledge, and to develop new methods to optimize them.

At a broad level, we can view the participation of users in the process of dissemination as falling in one of two settings: collaborative or competitive. In the collaborative setting, the participants work together in crafting *knowledge* online, *e.g.*, by asking questions and contributing answers, or by discussing news or opinion pieces. In contrast, as competitors, they vie for the attention of their *followers* on social media. This thesis investigates both these settings.

The first part of the thesis focuses on the understanding and analysis of content being created online collaboratively. To this end, I propose models for understanding the complexity of the content of collaborative online discussions by looking exclusively at the signals of agreement and disagreement expressed by the crowd. This leads to a formal notion of complexity of opinions and online discussions. Next, I turn my attention to the participants of the crowd, *i.e.*, the creators and consumers themselves, and propose an intuitive model for both, the evolution of their expertise and the *value* of the content they collaboratively contribute and learn from online Q&A based forums. The second part of the thesis explores the competitive setting. It provides methods to help the creators gain more attention from their followers on social media. In particular, I consider the problem of controlling the timing of the posts of users with the aim of maximizing the attention that their posts receive under the idealized setting of full-knowledge of timing of posts of others. To solve it, I develop a general reinforcement learning based method which is shown to have good performance on the *when-to-post* problem and which can be employed in many other settings as well, *e.g.*, determining the reviewing times for spaced repetition which lead to optimal learning. The last part of the thesis looks at methods for relaxing the idealized assumption of full knowledge. This basic question of determining the *visibility* of one's posts on the followers' feeds becomes difficult to answer on the internet when constantly observing the feeds of all the followers becomes unscalable. I explore the links of this problem to the well-studied problem of web-crawling to update a search engine's index and provide algorithms with performance guarantees for feed observation policies which minimize the error in the estimate of visibility of one's posts.

---

[1] https://stats.wikimedia.org

[2] https://stackoverflow.com/questions

[3] Reuters Institute Digital News Report — 2017

[4] Twitter Q4 2018 Metrics report

[5] https://www.fastcompany.com/40555712/duolingo-suddenly-has-over-twice-as-much-language-learning-material

Zusammenfassung

In der Vergangenheit waren Informations- und Wissensvermittlung immer auf klassische Klassenräume, Zeitungen, Radio und Fernsehen beschränkt. Da diese Prozesse einfach und zentralisiert waren, waren sowohl ihre Modelle als auch die empirischen Optimierungsmethoden gut verständlich. In der heutigen Welt ist das Internet zu einem mächtigen Instrument der Vermittlung von Informationen und Wissen geworden: Bei Wikipedia finden täglich mehr als 1 Million Überarbeitungen statt[6], bei Stack Overflow wurden über 17 Millionen Fragen gestellt[7], 25% der US-Bevölkerung besuchen die Yahoo! News-Website[8], um Artikel zu lesen und darüber zu diskutieren, Twitter hat monatlich mehr als 60 Millionen aktive User[9] und Duolingo verzeichnet 25 Millionen User, die online Fremdsprachen lernen[10]. Diese Entwicklungen haben einen Paradigmenwechsel im Vermittlungsprozess initiiert. Es hat sich nicht nur die Art der Aufgabe von der Zentralisierung zur Dezentralisierung gewandelt, die Entwicklungen haben auch die Grenze zwischen dem *Schöpfer* und dem *Konsumenten der Inhalte*, d. h. der Information und des Wissens, verwischt. Diese Veränderungen haben es notwendig gemacht, neue Modelle zu entwickeln, die besser geeignet sind, die Vermittlung zu verstehen und zu analysieren, und neue Methoden zu entwickeln, um sie zu optimieren.

Auf breiter Ebene kann man die Beteiligung der User am Vermittlungsprozess in einem der folgenden Bereiche sehen: Zusammenarbeit oder Wettbewerb. Im kollaborativen Bereich arbeiten die Teilnehmer zusammen, um online Wissen zu erarbeiten, z. B. indem sie Fragen stellen und Antworten beisteuern oder indem sie Nachrichten und Meinungsbeiträge diskutieren. Im Gegensatz dazu wetteifern sie als Konkurrenten in sozialen Medien um die Aufmerksamkeit ihrer *Follower*. Die vorliegende Arbeit untersucht diese beiden Bereiche.

Der erste Teil der Arbeit konzentriert sich auf das Verständnis und die Analyse von Inhalten, die in kollaborativer Weise online erstellt werden. Dazu schlage ich ein Modell zum Verständnis der Komplexität der Inhalte von kollaborativen Online-Diskussionen vor, indem ich ausschließlich die Signale der Zustimmung und Ablehnung betrachte, die die Community verwendet. Dies führt zu einer formalen Vorstellung der Komplexität von Meinungen und Online-Diskussionen. Daraufhin richte ich meine Aufmerksamkeit auf die Teilnehmer der Community, d. h. auf die Schöpfer und Konsumenten selbst, und schlage ein intuitives Modell sowohl für die Entwicklung ihres Fachwissens als auch für den *Wert* der Inhalte vor, die sie gemeinsam in auf Fragen und Antworten basierenden Foren beisteuern und von denen sie lernen. Der zweite Teil der Arbeit befasst sich mit dem Wettbewerbsbereich. Er liefert Methoden, die den Schöpfern helfen, mehr Aufmerksamkeit von ihren Followern in den sozialen Netzwerken zu gewinnen. Insbesondere betrachte ich das Problem des *richtigen Timings* für die Beiträge der User, die das Ziel verfolgen, die Aufmerksamkeit auf ihre Beiträge zu maximieren, und dabei ständig im Konkurrenzkampf mit ihren Mitbewerber stehen, denen immer das ideale Timing zu gelingen scheint. Um dieses Problem zu lösen, entwickle ich eine allgemeine, auf dem bestärkenden Lernen basierende Methode, die nachweislich bei dem Problem des richtigen Timings für die Beiträge hilft und auch in vielen anderen Bereichen eingesetzt werden kann, z. B. bei der Bestimmung, wann Zeit für regelmäßig durchgeführte Wiederholungen ist, die zum optimalen Lernen führen. Der letzte Teil der Arbeit befasst sich mit den Methoden zur Lockerung der idealisierten Voraussetzung des vollständigen Wissens. Diese grundlegende Frage, zu ermitteln, ob die eigenen Beiträge in den Follower-Feeds *sichtbar* sind, ist im Internet

---

[6] https://stats.wikimedia.org

[7] https://stackoverflow.com/questions

[8] Reuters Institute Digital News Report – 2017

[9] Twitter Q4 2018 Metrics report

[10] https://www.fastcompany.com/40555712/duolingo-suddenly-has-over-twice-as-much-language-learning-material

schwierig zu beantworten, angesichts dessen, dass eine ständige Beobachtung der Feeds aller Follower schwer durchführbar ist. Ich untersuche die Verknüpfung dieses Problems mit dem gut erforschten Problem des Web-Crawling, um den Index einer Suchmaschine zu aktualisieren und Algorithmen mit Leistungsgarantien für Feed-Beobachtungsrichtlinien zu liefern, die den Fehler bei der Einschätzung der Sichtbarkeit der eigenen Beiträge minimieren.

Acknowledgements

I would like to thank my advisor Prof. Rupak Majumdar for his advice and support; I still cringe when I think of my first meeting with him while I was about to graduate my MS at EPFL to join a startup. However, he was gracious enough to overlook my naïvety and seriously consider me for a position at MPI-SWS.

Dr. Manuel Gomez-Rodriguez was my thesis advisor and easily the one from whom I learned the most. I was introduced to the world of Network Learning in exploring Information Dissemination under his tutelage. Equally important were discussions I had with Prof. Krishna Gummadi and my friends and Post Doctoral researchers at the Networks Learning lab during my stay there Isabel Valera and Abir De. Similar discussions, some technical and some philosophical, were had with numerous other interns who visited the lab, some of whom I had the fortue to collaborate with: Behzad Tabibian, Harry Mavroforakis and Ali Zarezade foremost among them. There were many people who I collaborated with and they all also taught me something special each: Steven Le Blond, Aasish Pappu, Robert Busa-Fekete, Wojciech Kotlowski, David Pal, and Balasz Szorenyi, among many others. Besides research, my work was also supported and made possible by the extensive help I received from other parts of MPI-SWS: with administration (Maria-Louise, Corinna, Susanne, Mouna and Vera), with IT (Christian Klein, Chriatian Mickler, Tobias, Carina, and Pascal), and my communication skills (Rose).

On the personal side, my family supported me through thick and thin and they never second guessed my decisions even when I did so myself. Apart from my rock solid mother, my discerning and loving brother Shreyes, and vivacious wife ET (Amita), this also included my late father who I lost during this journey and whose tempered advice is still sorely missed. I was also propelled forth by the good natured jibes from my mamajis. I also made many friends at Kaiserslautern made me a better person, be it with board games (Marko, Dmitry, Filip, Johannes), with lentil related culinary recommendations (Aastha and Arpan), kernel modules mixed with worldly matters (Soham, Manohar), programming competitions (Simin), or, in all fairness, just talking (Bilal, Ram). They and many other friends made my life rich and made it all worth it.

I would like to thank Ruzica Piskac who I was fortunate to meet while I was an intern at Microsoft Research in Seattle in 2007, then again at EPFL in 2009, and then again at MPI-SWS in 2015. She was who proverbially started me on this path and has always been an academic guiding angel to me. A special thanks to Gregory for showing me the ropes while doing my first research projects. Several other friends who deserve to be named here are those who have dissuaded me from doing stupid things (The Great Saxena Ji, Daddy Agent, Nagpal, RK), and encouraged me to try exciting things (Ramnik, Bheekling, and Miss Sarajevo).

Contents

1 Problem and Motivation **11**

2 Background and Contributions of this Thesis **15**
2.1 Related work 15
2.2 Primary Contributions 17

3 Collaborative Knowledge Creation and Consumption **19**
3.1 On Complexity of Opinions and Online Discussions 20
3.1.1 Modeling opinions and votes 21
3.1.2 Complexity of discussions 23
3.1.3 Estimating multisided opinions 27
3.1.4 Experiments 28
3.1.5 Conclusions 32
3.2 Understanding Evolution of Expertise 32
3.2.1 A Crowdlearning Model 34
3.2.2 Experiments on Synthetic Data 37
3.2.3 Experiments on Real data 40
3.2.4 Discussion 44
3.2.5 Conclusion 45

4 Competitive Knowledge and Information Dissemination **47**
4.1 Deep Reinforcement Learning for *when-to-post* problem 48
4.1.1 Formulation 49
4.1.2 Control 51
4.1.3 Experiments on smart broadcasting 54
4.1.4 Summary 56
4.2 Learning to schedule 56
4.2.1 Problem Formulation 57
4.2.2 Learning with Poisson Processes and Partial Observability 58
4.2.3 Parameter Estimation and Sensitivity Analysis with Partial Observations 60
4.2.4 Explore-Then-Commit Algorithm 61
4.2.5 Experimental Evaluation 63
4.2.6 Conclusion 65

5 Conclusions and Future work **67**

A Appendix: Deep reinforcement Learning for *when-to-post* problem **77**
A.1 Proof of Proposition 4 77
A.2 Proof of Proposition 5 79
A.3 Sampling event times from the intensity $\lambda^*_\theta(t)$ 79

A.4 Experimental details . . . 80
A.5 Student model . . . 81
A.6 Feed sorting algorithm . . . 82
A.7 Experiments on feeds sorted in reverse chronological order . . . 82
A.8 Baseline with $w_t = 0$ . . . 83

B Appendix: Learning to Schedule 85
B.1 Proof of Lemma 1 . . . 85
B.2 Proof of Lemma 3 . . . 86
B.3 Proof of Lemma 2 . . . 87
B.4 Proof of Lemma 4 . . . 89
B.5 Sensitivity of Harmonic Staleness to Accuracy of Parameter Estimates . . . 91
B.6 Sensitivity of Accumulated Delay objective to Accuracy of Parameter Estimates . . . 93
B.7 Proof of Lemma 5 . . . 95
B.8 Utility of $\boldsymbol{\kappa}^{\text{UI}}$ and $\boldsymbol{\rho}^{\text{UR}}$ . . . 96
B.9 Exploration using $\boldsymbol{\rho} \in \Delta_R$ and Committing to use $\boldsymbol{\kappa} \in \mathcal{K}_R$ . . . 97
    B.9.1 Parameter Estimation using $\boldsymbol{\rho}^{\text{UR}} \in \Delta_R$ . . . 97
    B.9.2 Sensitivity of Utility to Parameter Estimation Accuracy using $\widehat{\boldsymbol{\kappa}} \in \mathcal{K}_R$ . . . 97
B.10 Performance of Moment Matching Estimator . . . 97
B.11 Parameter Estimation with Full Observations . . . 99

C Curriculum vitae 103

Chapter 1

Problem and Motivation

For a long time in human history, the primary modes of information and knowledge dissemination were centralized. For general information and news, there was mass-media (*i.e.*, television, radio, and newspapers), and for knowledge, there were brick-and-mortar schools and colleges. The primary reason behind the invention of the World Wide Web (WWW) was information dissemination, though initially it was limited to sharing documents and information among research institutes. With the advent of WWW, almost everyone got an accessible platform to disseminate their ideas, opinions, and knowledge. This started the move from centralized to de-centralized dissemination. On the heels of the first web-browser, AOL came onto the scene with the iconic "You've got mail!" chime and people almost immediately started *discussions* on the internet as messaging boards and mailing lists. This ushered in an era of *available-when-needed* and *near-real-time* dissemination.

The messaging boards and e-mails have clearly stood the test of time. They had the key features of allowing crowd participation in near-real time, but they were far from perfect for all forms of dissemination. As a result, several different online platforms have been developed in the past few years which offer a spectrum of features aimed at making dissemination easier and more effective. The features can be classified broadly into two categories based on which setting of dissemination they are designed to improve: the collaborative setting or the competitive setting.

The websites which aim to promote the collaborative form of dissemination added features aimed towards improving the way users interact with the content which they create together, *e.g.*, allowing editing posts, up/down voting to signal agreement or quality, posting nested replies, following discussions, better visualization/exploration/searchability, *etc.* Example of websites which have specialized for the collaborative setting are Wikipedia, Stack Overflow, Reddit, HackerNews, *etc.* In this thesis, I develop models and methods to analyse and understand the collaboratively created content, as well as the creators and consumers of the said content.

On the other hand, the websites that promote competitive dissemination added features to allow users to create their own *social network* overlaid on all the members of the site. A common feature which characterises such websites is that there is a *feed* for each user, which could be customized by choosing *who* to follow. Examples of websites which focus on this setting of dissemination are Twitter, Facebook, Instagram, *etc.* When looked at from the perspective of a content creator, *i.e.*, a poster or a *broadcaster*, this personalization results in *sharing* of the attention of their followers. This leads to competition among the broadcasters who share followees at a level of granularity never seen before[1]. We study this competitive dissemination with the aim of helping broadcasters gain more attention from their followers in this thesis as well.

---

[1] Micro Influencers: https://www.forbes.com/sites/barrettwissman/2018/03/02/micro-influencers-the-marketing-force-of-the-future/.

stackoverflow
YAHOO!
reddit
facebook
Collaborative dissemination
Content creators and consumers
Competitive dissemination

Figure 1.1: Outline of the dissertation topics. There are two distinct settings in which online information dissemination occurs: collaborative (*e.g.*, Stack Overflow, Yahoo! News comments sections, Reddit, *etc.*) and competitive (*e.g.*, Twitter, Instagram, Facebook, *etc.*). The first part of this thesis is devoted to models for the information and knowledge dissemination process under the collaborative setting. The second half of the thesis is devoted to the problem of gaining user attention in the competitive setting.

Different platforms/websites offer their unique set of features, well adapted for their domains. However, there are several websites (*e.g.*, Quora) and apps (*e.g.*, Pinterest) which offer a mix of aforementioned features. For example, all of Quora, DeviantArt, and Pinterest allow users to have a content-centric (collaborative) as well as a feed-centric (competitive) view. The models, methods, and insights uncovered in this thesis will be relevant for these platforms as well.

Collaborative dissemination. The information dissemination process has a long history, nearly as long as the human race itself. In pre-historic times, after the advent of language, word-of-mouth was the only method of information dissemination. With Gutenberg's press (the 1500s), it became possible to disseminate information on a *mass* scale. Modern versions of educational institutes were contemporary with the industrial age (19th century) and mass dissemination of information was furthered by radio (late 19th century) and television (mid 20th century). Thanks to the long history behind the dissemination process of mass media and educational institutes, we have a relatively good understanding of the basic tenets underlying them. We have honed and improved curriculums and have designed examination systems to optimize the *just-in-case* learning for students, *i.e.*, arriving at the *core* knowledge which everyone should have. We have also gained some basic understanding of mass media, by similar experimentation as with educational institutes. For example, the effects of constraining *freedom of press* are broadly agreed on, even though we may not agree on the finer points.[2] At the same time, detailed study of the dissemination process was difficult and the models used for them were simplistic because the observability of the data was very coarse, *i.e.*, only available at an aggregate level after laborious surveys, which are usually self-reported and imprecise. The methods for controlling them were also simple and coarse as these institutions and mass-media were both centralized and could be regulated.

However, today's world has become much less structured than that. In some regards, we have regressed to the peer-to-peer dissemination, but the dissemination now has permanence. Unlike the spoken word, contributions made on the public internet may last forever[3] and, unlike book and newspapers, are easily searchable. Moreover, one's peers are not bound by geographical proximity any more. Knowledge consumption is also moving from *just-in-case* to *just-in-time*, when students and practitioners learn the material when they need it by visiting sites like Wikipedia or Stack Overflow, instead of learning it with the hope of eventually using it someday. The pursuit and consumption of knowledge is moving online at an unprecedented rate and is not exhibiting any signs of slowing down. These sites are rich with detailed and long traces of user

---

[2] See, for example, the FCC Fairness Doctrine of 1949.

[3] See https://archive.org/

activity, sometimes spanning years, which can lead to insights in evolution of expertise of the users, user-user interactions, and in quantifying knowledge content, or constructiveness, of items on the site. Though this data is being collected and stored, we have neither models which can help us in understanding or predicting user learning activity, nor methods for improving quality and efficiency of learning.

Formation and consumption of knowledge, information, and opinions has become a multifaceted and complex process which spans many areas of research pertaining to humans: cognitive science (expertise and memory), game theory (altruistic collaboration), and, behavioral psychology (gamification, mechanism design), without even mentioning engineering challenges of building systems which can scale to millions of people. In this thesis, I develop models for dissemination which use insights from these disparate areas of research and help us understand and analyse the dissemination process.

Hence, the first part of my PhD is dedicated to gaining a holistic understanding of the creators, consumers, content, and process of collaborative human-learning on crowdlearning sites with the aim of helping the users in seeking *arete* on their own.

> **arete**: *excellence of any kind*; bound up with the notion of the fulfillment of purpose or function: the act of living up to one's full potential.
>
> —Greek moral virtue

Competitive dissemination. The attention of users in the online world is a scarce commodity. In the world of mass dissemination, this attention was often shared by the major information sources: newspapers, television, and radio. Newspapers allowed the reader to resume reading later while television/radio required immediate attention. Also, lacking immediate feedback, the programming on the television/radio and the content of the newspapers could not be altered *on-the-fly*. Determination of what qualified for the attention of the users was centrally done, including of items requiring "immediate" attention, *e.g.*, breaking news.

However, modern dissemination is not bound by the same restrictions. Immediate notifications for consumers, and instant feedback for content creators, are now the norm rather than the exception. Moreover, with many social media sites allowing users to curate their own *feeds* by choosing who to follow, the attention of each follower has become divided far more granularly than ever before. The emergent field of Social Media Marketing [Constantinides, 2014] is a testament to this changing nature of information consumption which can no longer be ignored. The general focus of the field is determining how to generate viral content, growing the audience, and keeping them engaged by curating the content [Berger and Milkman, 2012]. However, there is a different dimension along which the broadcasters are also competing, which is the *timing of the posts*. Exploring the competition at the level of timing and visibility of posts has unique advantages. Firstly, the visibility of posts is more fundamental than content: Followers cannot engage with a post they do not see. The aggregated likes and reshares of a post are contingent on the post being visible to the followers. Without knowing whether the post was visible or not, the content creator will forever be in dark about whether the lack of engagement with a certain post was due to the content of the post or due to major breaking news which flooded the feeds of the followers at the same time. Secondly, the visibility of posts is an objective metric one can target for optimizing dissemination in real-time rather than the likes/shares which are confounded with other factors, such as the popularity of the content creator.

In this thesis, I consider the problem of determining and maximizing the *visibility* of the posts a user makes on the feeds of his followers in an online fashion. First, I describe a reinforcement learning based method for *controlling* the timing of the posts of a user to help her capture more attention by increasing the visibility of her posts. Next, I develop methods for approximating whether a given post by a user is visible to her followers given the constraints imposed by the social media platform on direct querying, *i.e.*, API rate-limits on requests to learn about the followers' feeds. This relaxes the idealized assumptions of having full-knowledge of others' actions which many previous approaches to solving online problems make.

The outline of the thesis and the state of online information dissemination is succinctly outlined in Figure 1.1. The next Section (*i.e.*, Section 2) gives some background and related literature to place the contributions of the thesis in context. The ensuing sections introduce the models and methods that have been developed for collaborative (Sections 3) and competitive (Section 4) dissemination. Finally, I summarize the contributions of the PhD in Section 5.

Chapter 2

Background and Contributions of this Thesis

2.1 Related work

This thesis builds on and draws inspiration from multiple areas of research ranging from expertise/opinion identification and evolution to modeling visibility on social media. In this section, I give a brief background of the current state of the art in each of these areas.

Opinion formation. Modeling user opinions has been a rich area of research with extensive theoretical and empirical studies. The seminal work in opinions formation in a social network was done by DeGroot [1974] while the connections of opinions to votes cast were explored even earlier by Downs [1957]. However, these works and significant research inspired by them, both theoretical [Rowley, 1984, Axelrod, 1997, Hegselmann and Krause, 2002, Holme and Newman, 2006, Yildiz et al., 2010, 2013] and empirical [Pang et al., 2008, Choi et al., 2010, Conover et al., 2011, Guerra et al., 2013, De et al., 2014, Mejova et al., 2014, Barberá, 2015, Garimella et al., 2018], have represented the opinions as being uni-dimensional. This assignment of a real number to the opinions of people allows their coarse characterization into two groups, *e.g.*, left leaning and right leaning. This representation is amenable to analysis [Rowley, 1984], but at the same time, is severely limited in its ability to capture complex and multisided opinions. There exist many *voting patterns* which are often observed in online discussions which cannot be explained by uni-dimensional opinion models at all. Hence, these models are not easily applicable to collaborative online discussions where members of the crowd may bring in a nuanced and unique point of view with their comment and express their agreement and disagreement with the opinions expressed in the existing comments by voting.

There are some notable exceptions to the uni-dimensional modeling of opinions which have been explored in the literature—aspect-oriented sentiment analysis [Liu, 2012, Broß, 2013] and collaborative filtering based on matrix factorization [Bennett et al., 2007, Koren and Bell, 2015, Linden et al., 2003]. Aspect-oriented sentiment analysis is primarily designed for online reviews and relies heavily on textual analysis or hand-crafted dictionaries to map products to their *aspects*, *i.e.*, sides of restaurant or products, *etc.* Collaborative filtering, on the one hand, models the *taste* of users and *score* of items (*e.g.*, movies) across multiple factors (*i.e.*, sides), with the aim of predicting how much a user will *like* (*i.e.*, *rate*) an item after observing a partial set of user-item ratings. They usually use empirical methods (*e.g.*, cross-validation) to determine the number of sides and consider the observed user-item ratings to be real-valued numbers instead of binary like/dislike votes. Both of these approaches use ad hoc methods for determining the sides of opinions which work well in their respective domains. This thesis proposes models and inference methods for collaboratively sourced discussions which provide a theoretically grounded measure of complexity which is based on crowdsourced binary human judgement.

Expertise identification and evolution. Identifying topical authorities or experts, *i.e.*, users who provide high-quality contributions, on Q&A [Hanrahan et al., 2012, Jurczyk and Agichtein, 2007, Pal and Konstan, 2010, Zhang et al., 2007] and microblogging sites [Ghosh et al., 2012, Pal and Counts, 2011], has received much attention recently. The problem of expert finding in Q&A sites was first studied by Zhang et al. [2007], who formulated it as a ranking problem and developed several PageRank based methods. Shortly after, Jurczyk and Agichtein [2007] tackled the problem using link analysis techniques. More recently, Pal and Konstan [2010] approached the problem from the perspective of supervised learning and developed Gaussian classification models to distinguish between ordinary and (potential) expert users, and Hanrahan et al. [2012] described a method to find experts given a specific target question. In the context of microblogging, the problem of expert finding was first studied by Pal and Counts [2011], who proposed a set of features for characterizing contributors and then formulated the problem using unsupervised learning in this feature space. Since then, Ghosh et al. [2012] have mined Twitter users' lists to find topical authorities and Kao et al. [2010] and Paulina and Marta [2015] leveraged temporal statistics on the users' activity to identify experts. Finally, in the context of web search, White et al. [2009] studied how expertise influences search and Eickhoff et al. [2014] investigated how users can increase their expertise as they look for *procedural* and *declarative* knowledge using a search engine. This work on expert identification, however, does not capture the evolution of users' expertise over extended periods of time, nor accounts for the *value* of their contributions. This thesis proposes a probabilistic model which can help us understand the evolution of expertise as well as the *effective* knowledge users contribute to the site.

The interest in the field of modeling and measuring learning, *i.e.*, evolution of expertise, is very old and has led to the development of several paradigms over the last century in the experimental psychology literature [Embretson and Reise, 2013, Means et al., 2009, Norman and Schmidt, 1992]. Most of the research has, however, either happened in strictly controlled environments (*i.e.*, schools or study groups) or used centralized assessments (*e.g.*, SAT or local testing). Closer to the model proposed in this thesis are probabilistic models developed by the intelligent tutoring communities: Bayesian knowledge tracing [Corbett and Anderson, 1994, Gonzalez-Brenes and Mostow, 2013, Qiu et al., 2016, Yudelson et al., 2013], performance factor analysis [Pavlik et al., 2009] and ensembles [Baker et al., 2011]. However, the work typically relies on controlled assessment and manually annotated knowledge items, even if allowing for different knowledge values per item [Beck and Mostow, 2008]. Piech et al. [2015] solved this limitation by resorting to recurrent neural networks to model the learning of students but, unfortunately, they use metrics that are not suitable for crowdlearning. These models capture the evolution of expertise in a very narrow sense and under controlled settings. The model for collaborative crowdlearning proposed in this thesis, on the other hand, provides a general understanding of crowdlearning dynamics, from uncovering the evolution of users' expertise over time and understanding the interplay between learning and contributing, to identifying questions with a high knowledge value.

Controlling dissemination. The problem of *when-to-post* to optimize for visibility has recently received popular media attention.[1] It provides an ideal setup to study the problems of competitive dissemination, where the creators of content, *i.e.*, the broadcasters, are vying for attention of their followers. This thesis uses temporal point processes to model the posting behavior of users on social media. Previous work has used the theory of marked temporal point processes [Aalen et al., 2008b] to model the temporal dynamics of human activity and information dissemination on the internet. Recent research has shown that these processes are sufficiently expressive to model a large variety of human activities [Farajtabar, 2018], show superior performance on predictive tasks [Du et al., 2016], and provide a theoretically sound foundation for designing optimal strategies for several important control problems involving humans [Kim et al., 2018, Zarezade et al., 2018]. In particular, this problem of deciding *when-to-post* has been studied from multiple angles before [Karimi et al., 2016, Wang et al., 2017, Zarezade et al., 2017a], but all approaches have relied

---

[1] https://sproutsocial.com/insights/best-times-to-post-on-social-media/, https://www.oberlo.com/blog/best-time-post-social-media, https://blog.hubspot.com/marketing/best-times-post-pin-tweet-social-media-infographic, https://buffer.com/library/best-time-to-post-on-facebook, *etc.*

on either the dynamics of the other broadcasters being known [Karimi et al., 2016, Wang et al., 2017], the feed-ranking mechanism being known [Karimi et al., 2016, Wang et al., 2017, Zarezade et al., 2017a], or the functional form of the objective being amenable to analysis [Karimi et al., 2016, Zarezade et al., 2017a]. In contrast, the reinforcement learning based method described in this thesis makes no assumptions about the behavior of other broadcasters, can adapt to algorithmic feeds, and allows optimization of any arbitrary objective. The method, without minor modifications, is also applicable to the more general problem of controlling actions of an agent where the actions and feedback can be represented as marked temporal point processes.

Online learning of dissemination dynamics. Most previous work makes one or both of the following assumptions: (i) the parameters describing the behavior of the environment (*i.e.*, competing broadcasters) do not change and can be learned by means of past history of posting [Karimi et al., 2016], or (ii) an idealized setting wherein the agent can observe the posts of all the competing broadcasters in real-time. However, the dissemination process is dynamic and, hence, can change over time and the assumption of independence may not hold. Additionally, keeping track of posts of all other broadcasters may not be feasible given the scale of the network. In some aspects, this problem is similar to the extensively studied problem of web-indexing faced by search engines, where the crawler is limited by the bandwidth available to the indexing server and has to decide to poll the websites at different rates, based on how often they change. The problem of effectively estimating the rate and that of determining the optimal crawling policy have been studied the seminal papers by Cho and Garcia [Cho and Garcia-Molina, 2003b,a]. Closer to our setup, the problem of keeping up-to-date with changing news has been studied by Sia et al. [2007], where optimal polling policies are described. Recently, Azar et al. [2018] have uncovered tractable optimizing algorithms to uncover the optimal policy for updating the web-pages. The problem of determining the optimal refresh policy while also learning the rates at which the sources (*i.e.*, web pages or feeds of followers) change can be cast as an online optimization problem with bandit feedback, where the feedback is received when a source (*i.e.*, a webpage, a feed, or a competing broadcaster) is polled and only for that source. Very recently, Kleinberg and Immorlica [2018] have investigated a similar multi-arm bandit problem but in the discrete setting, where the reward accumulates the longer an arm (*i.e.*, a source) goes without a pull (*i.e.*, a poll). I am actively working towards arriving at methods for solving the problem which provably optimize information which an agent has while also learning the parameters describing the dissemination process.

2.2 Primary Contributions

The contributions of the thesis can broadly be divided into two sections: the first section models the collaborative dissemination of information via latent space modeling of opinions and expertise (Section 3), and the second section studies the competitive dissemination of information, delving into efficiently analysing and flexibly controlling dynamic process of posting messages on social networks (Section 4).

Section 3.1 describes the first model for collaborative dissemination of information for the up/down votes cast by voters on comments in an online discussion. The model can determine the complexity of opinions being expressed in online discussions via crowdjudgement, as well as the relative agreement among the comments. This work was done in collaboration with researchers from Yahoo! Research Labs and was presented at the ACM conference on Web Search and Data Mining (WSDM) [Upadhyay et al., 2019].

Next, I model the evolution of expertise among users of a knowledge-based social network (Stack Overflow). The model leverages the detailed traces of learning and contributing activity users leave behind in their activity logs. Section 3.2 describes the model, its analysis, and the resulting insights. Since this model too requires knowledge of the identity of the voters along with the votes, I wrote a proposal to the Digital Ecologies Research Partnership[2], which was accepted by Stack Overflow, and I was given access to the de-anonymized data. The results and insights were presented at the ACM conference on Web Search and Data Mining (WSDM) [Upadhyay et al., 2017].

---

[2] http://derp.institute/

In the competitive dissemination setting, in Section 4.1, I propose a general deep reinforcement learning based method for controlling marked temporal point processes which can be used to determining the best time to post to maximize the visibility of posts, under the idealized setting that the agent can observe the posts made by competing broadcasters instantaneously. This work was presented at the Neural Information Processing Systems (NeurIPS) conference [Upadhyay et al., 2018].

I am currently exploring the problem of learning the optimal scheduling policy for polling the feeds of followers to determine the visibility of one's posts when the parameters describing the behavior of other broadcasters are not known. Having such a policy with theoretical performance guarantees can be used to relax the assumption of having instantaneous information about the competing broadcasters in many problems related to controlling human activities, including the problem of *when-to-post* described above. This work is being done in collaboration with researchers from the Yahoo! Research Labs. Section 4.2 describes the problem formulation in more detail. This work was presented at the 34th AAAI conference on Artificial Intelligence [Upadhyay et al., 2020].

Other contributions. During the progress of my thesis, I also made other contributions to other research projects, which complimented the contributions of my thesis. I contributed to the development of a recurrent neural network based method (called Recurrent Marked Temporal Point Processes, RMTPP) to embed human activity history to real-valued vectors, leading to a model which excelled at predictive tasks. This work was presented at ACM conference for Knowledge Discovery and Data mining (KDD) [Du et al., 2016]. I also worked on using semi-supervised learning to categorize malicious documents [Le Blond et al., 2017] (presented at the Network and Distributed System Security Symposium, NDSS). I also contributed to the development of the theory of stochastic optimal control of temporal point processes, which led to state-of-the-art control policies for social activities [Zarezade et al., 2017a,b] (presented at ACM conference for Web Search and Data Mining, WSDM, and published in the Journal of Machine Learning, respectively), for controlling epidemics [Lorch et al., 2018] (presented at the Workshop on Machine Learning for Health, NeurIPS), and for scheduling the spacing of reviews for optimal recall [Tabibian et al., 2019b] (published in Proceedings for National Academy of Sciences). The limitations of the approaches based on stochastic optimal control were the inspiration behind the reinforcement learning based approach I develop in the latter part of the thesis [Upadhyay et al., 2018]. Finally, I have also worked on complementary problems of investigating whether users strategically change their posting behavior based on feedback their posts receive, *i.e.*, based on what their followers want [De et al., 2019].

Chapter 3

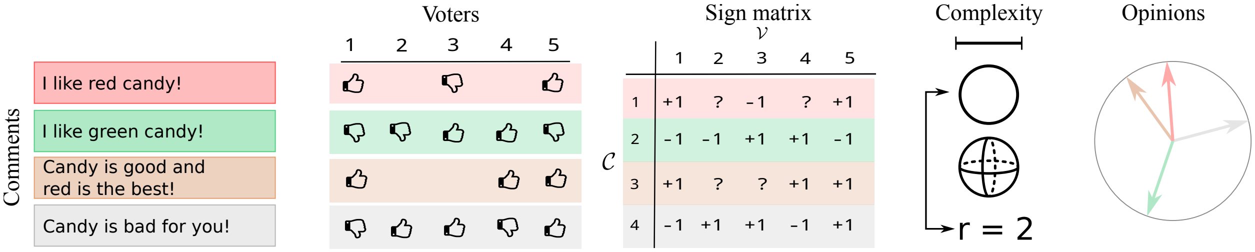
Collaborative Knowledge Creation and Consumption

Since its invention, the internet has brought about a precipitous change in our world. Undeniably, one of its major contributions has been making the collaboration of people from across the world and all phases of life possible. There are many testaments to the fruits of such mutually beneficial collaborations. Because the internet has made it possible for each person to find his or her niche [Day, 2015], the contributions also come in the form of Q&A forums where technical knowledge is celebrated (*e.g.*, Stack Overflow), or expression of views which may act as catalyst for changing (and cementing) views on various current and historically relevant topics (*e.g.*, comments on Reddit, news articles, *etc.*). Online forums are the quintessential form of crowd-contributed, crowd-judged (by means of up/down votes), and crowd-consumed sources of knowledge and information on the internet. These forums can take a variety of forms, ranging from of question answering websites (*e.g.*, Stack Overflow) to commenting sections of news sites (*e.g.*, Yahoo! News). This thesis explores some key questions about the collaborative processes happening on the internet:

1. What is the *complexity* of a discussion happening online, and by how much do the opinions of the participants agree/disagree with each other?
2. How does expertise of users evolves with time and what is the *value* of each item of knowledge they contribute?

To this end, the following sections introduce models of crowd contributed and crowd judged content on social media. A key idea underlying both approaches is to extract signals from the traces of activity which the users have left behind on social networks either as signals of their learning (*e.g.*, upvotes on answers) or as signals of their (dis)agreement with opinions of others (*e.g.*, up/down votes on comments).

In particular, Section 3.1 considers the problem of understanding the *complexity* of discussions happening in comment sections of websites (Yahoo News!, Yahoo Finance!, *etc.*) by leveraging the pattern of up and down votes left on the comments by the crowd. I propose a simple model for the voting process which can able to quantify the relative *disagreement* in the discussions, and be able to infer where do the *opinions* of the participants lie relative to each other. This work was done in collaboration with researchers from Yahoo! Research Labs and MPI-SWS. The results and insights were presented at the ACM conference on Web Search and Data Mining (WSDM) [Upadhyay et al., 2019]. Next, Section 3.2 tackles the closely related problem of modelling both the evolution of expertise as well as the *value* of knowledge items on the Q&A website Stack Overflow. The model allows us to earn insights into the new world of crowdlearning. This work was done in collaboration with co-researchers at MPI-SWS and the results and insights were presented at the ACM conference on Web Search and Data Mining (WSDM) [Upadhyay et al., 2017].



I like red candy!

I like green candy!

Candy is good and red is the best!

Candy is bad for you!

$\mathcal{C}$ \ $\mathcal{V}$	1	2	3	4	5
1	+1	?	-1	?	+1
2	-1	-1	+1	+1	-1
3	+1	?	?	+1	+1
4	-1	+1	+1	-1	+1

Figure 3.1: The complexity and opinion modeling framework. From left to right, given a (toy) online discussion with a set of comments $\mathcal{C}$ and voters $\mathcal{V}$, the framework maps the upvotes and downvotes into a partially observed sign matrix $\boldsymbol{S}$. Within $\boldsymbol{S}$, each row corresponds to a comment and each column corresponds to a voter. Each $+1$ entry indicates that the voter upvoted the comment, -1 indicates that she downvoted the comment, and ? indicates that the voter did not vote. Then, the framework represents the opinions expressed in the comments and those held by the voters as r-dimensional real-valued vectors lying in the same latent space of opinions. Finally, it provides practical algorithms to both estimate the dimension r of the latent space of opinions as well as infer the vectors of opinions which are consistent with the partially observed sign matrix $\boldsymbol{S}$.

3.1 On Complexity of Opinions and Online Discussions

Discussions have always been part of the online world since the invention of Internet. Starting from Usenet boards, the discussions have evolved and continued in a multitude of forms to the present day, be it question answering forums or opinion sharing; with different variants showing different features, most notably, the feature of expressing agreement or disagreement with a comment by means of casting a *vote* on the comment. This feedback mechanism, which is present in several current incarnations of the messaging boards (Reddit, HackerNews, Yahoo! News, Stack Overflow, Quora, *etc.*.), made it even easier to participate in online discussions, bringing a larger swath of the crowd into the fold of either explicitly or implicitly expressing their opinions online. Paradoxically, even as more and more people join the online world, the online discussions which are happening seem to be becoming more and more polarized. The reasons behind this phenomenon are multi-faceted, but it is suspected that demagogues are exploiting the “us” versus “them” mentality to further deepen the divide [Bovet and Makse, 2019]. Hence, reduction of discussions to a single dimension might be one of the reasons behind the polarization. It does not help that most classical academic approaches to modeling opinions, both theoretical [Axelrod, 1997, Hegselmann and Krause, 2002, Holme and Newman, 2006, Yildiz et al., 2010, 2013] and empirical [De et al., 2014, Pang et al., 2008, Choi et al., 2010, Mejova et al., 2014, Conover et al., 2011, Guerra et al., 2013, Garimella et al., 2018, Barberá, 2015], have also represented the opinions as only a number on the real line, primarily for the reasons of interpretability and for ease of analysis.

This section in the thesis presents a framework for analysing and understanding the multisided opinions of online discussions where the users not only express their opinions, but also are able to express their agreement or disagreement with the opinions of others by means of upvotes and downvotes. This work was done in collaboration with researchers from Yahoo! Research Labs and was presented at the ACM conference on Web Search and Data Mining (WSDM) [Upadhyay et al., 2019]. The approach described involves first quantifying the *complexity* of such discussions and then representing the opinions being expressed by the comments in a latent space with a unique property: the comments which agree with each other in their *intent*, as judged by the crowd voting on them, lie *close* to each other in the latent space. Moreover, the method is able to circumvent the problems one faces while trying to use Natural Language Processing based techniques for the same task as it will use only the up/down voting data associated with the discussions. Hence, the methods can compliment the methods based on NLP while providing a more theoretically grounded notion of complexity and latent opinions.

More concretely, given an online discussion consisting of a set of comments, which are upvoted and downvoted by a set of voters, first a latent multidimensional representation of the opinions expressed in the comments and the opinions held by the voters is introduced. Then, two voting models, one deterministic and another probabilistic, are proposed which leverage the above multidimensional representation to characterize the voting patterns within an online discussion. Under this characterization, it becomes apparent that the dimensionality of the latent space of opinions is a measure of complexity of the online discussion—along how many different axis can the opinions expressed in the comments and the opinions held by the voters differ. The representation of opinions in this latent space of opinions has a remarkable property: if two opinions are close (far away) in the latent space it is because the voters—the crowd—think that they are *similar* (*dissimilar*). Such a property may not hold for other representations of the opinions, *e.g.*, those based only on the textual data in the comments [Le and Mikolov, 2014, Palangi et al., 2016] because of nuances in the use of language. Motivated by these observations, the following is developed[1]:

1. A polynomial time algorithm to determine an upper bound on the minimum dimensionality that a latent space of opinions needs to have so that they are able to explain a particular voting pattern under the deterministic voting model.
2. An inference method based on quantifier elimination to recover the latent opinions from the observed voting patterns under the deterministic voting model.
3. An inference method based on maximum likelihood estimation to recover the latent opinions from the observed voting patterns under the probabilistic voting model.

Finally, the modeling framework is evaluated using a large dataset from Yahoo! News, Yahoo! Finance, Yahoo! Sports, and the Newsroom app, which consists of one day of online discussions about a wide variety of topics. The analysis yields several interesting insights. We find that only $\sim$25% of the online discussions we analyzed can be explained using a unidimensional representation of opinions, $\sim$60% of them require a two dimensional representation, and the remaining ones require a greater number of dimensions. This provides empirical evidence that, to provide opinions representations that are coherent with human judgments, it may be often necessary to move beyond one dimension. The presence of multisided opinions is an indication that the discussion may not be falling prey to demagoguery. This finding is also supported by a positive correlation between the dimension of a discussion and its linguistic diversity. Moreover, the estimated r-dimensional opinions allow prediction of upvotes/downvotes in a discussion more accurately than a state of the art matrix factorization method [Mazumder et al., 2010] and a logistic regression classifier [Murphy, 2012a]. In this context, whenever an online discussion can be represented using one dimensional opinions, the deterministic model achieves higher predictive performance than the probabilistic model. However, for discussions with multisided opinions, the probabilistic model, which allows for noisy voting, provides more accurate predictions. This suggests that, whenever humans face more complex discussions, their judgments become less *predictable*. Moreover, there is a positive correlation between the complexity of the discussions and the level of agreement among comments. Lastly, by looking at particular examples of online discussions (see Figure 3.10 and 3.11), we see that the modeling framework, by relying on human judgments, may be able to circumvent language nuances like sarcasm and humor, which are often difficult to detect using natural language processing. The examples also illustrate how the dimensions uncover the different *sides* of the discussion.

3.1.1 Modeling opinions and votes

At the very outset, the underlying mechanism behind voting on online discussions is fairly commonplace and straight-forward. Every time a user expresses an opinion by posting a new comment in an online discussion, other users can upvote (downvote) the comment to indicate that they agree (disagree) with the expressed opinion.

In this context, whenever a user upvotes or downvotes a comment, she reveals the relative position of her opinion with respect to the opinion expressed in the comment. By leveraging this observation to multiple comments, upvotes and downvotes, our modeling framework will be able to infer the relative positioning

---

[1] Implementation at `https://github.com/Networks-Learning/discussion-complexity`.

of comments in an online discussion, as judged by the crowd. Moreover, by doing so, it will also find a meaningful joint latent representation for the opinions expressed in an online discussion as well as the opinions held by the users who voted. In the remainder of the section, we formally introduce our modeling framework, starting from the data it is designed for.

Online voting data. We observe an online discussion consisting of a set of comments $\mathcal{C}$ which are upvoted and downvoted by a set of voters $\mathcal{V}$. Here, we keep track of *who* voted *what* by means of the variables $y_{ij} = \{+, -, \circ\}$, which indicate that voter $j \in \mathcal{V}$ upvoted, downvoted, or did not vote on comment $i \in \mathcal{C}$, respectively. Then, we define a (partially) observed sign matrix $\boldsymbol{S} = [s_{ij}]$, where each (i, j)-th entry is given by

$$s_{ij} = \begin{cases} +1 & \text{if } y_{ij} = + \\ -1 & \text{if } y_{ij} = - \\ \;\;? & \text{if } y_{ij} = \circ, \end{cases} \tag{3.1}$$

the sign ? indicates that the voter did not vote and thus we cannot know whether she agrees (or disagrees) with the comment. We denote by Ω the set of indexes where we have observations, *i.e.*, $\Omega = \{(i,j) \,|\, s_{ij} \neq ?\}$. Figure 3.1 illustrates the above definitions for a given toy example.

Next, we introduce our multidimensional representation of the opinions expressed in the comments and those held by the voters and then elaborate on our voting model, which relates these opinions to the observed voting data.

Opinion representation. Unidimensional (scalar) real-valued representations of opinions, owing largely to their interpretability, have been used most commonly in the literature, following the example set by the seminal works by DeGroot [1974] and Rowley [1984]. Thus, we could think of using such unidimensional representation of opinions in our work. However, under that choice, we would be unable to explain certain voting patterns illustrated below, which are common in many online discussions.

Given an online discussion, assume we represent the opinions expressed in each comment $i \in \mathcal{C}$ as $c_i \in \mathbb{R}$ and the opinion held by voter $j \in \mathcal{V}$ as $v_j \in \mathbb{R}$. Now, we elaborate separately on two of the most popular voting models in the literature [Merrill and Grofman, 1999]: the proximity model and the directional model. Under the proximity model, the voters use the Euclidean distance as a similarity measure and decide to cast an upvote if $|v_j - c_i| \leq \theta$, where θ is a threshold, and a downvote otherwise. Now consider the voting pattern 1 in Figure 3.2a. We can show that there are no real-valued scalar opinions v_1, v_2, v_3 and c_1, c_2, c_3 leading to such a voting pattern: assume that $v_1 \leq v_2 \leq v_3$ (as the pattern is symmetric, we can always relabel the voters and comments to make this true) and $c_2 < v_2$. Then $|v_2 - c_2| > \theta \implies v_2 > c_2 + \theta$, and $|v_3 - c_2| \leq \theta \implies v_3 \leq c_2 + \theta$. This contradicts the assumption that $v_2 \leq v_3$. We arrive at a similar contradiction with the assumption $c_2 > v_2$.

Under the directional model, the voters use the dot product as a similarity measure and decide to cast an upvote if $v_j \cdot c_i \geq 0$ and a downvote otherwise. Here, consider the voting pattern 2 in Figure 3.2b. Again, it is easy to show that there are no real-valued non-zero scalar opinions v_1, v_2, v_3 and c_1, c_2, c_3 leading to such a voting pattern. The first row requires $v_1 \cdot c_1 \geq 0$ and $v_2 \cdot c_1 < 0$, which implies $\text{sign}(v_1) \neq \text{sign}(v_2)$. However, the second row requires $v_1 \cdot c_2 \geq 0$ and $v_2 \cdot c_2 \geq 0$, which implies $\text{sign}(v_1) = \text{sign}(v_2)$ and this leads to a contradiction.

Motivated by the above examples, given an online discussion, we represent the opinions expressed in the comments and those held by the voters as r-dimensional real-valued vectors lying in the same latent space. More formally, we represent the opinion expressed in each comment $i \in \mathcal{C}$ as $\mathbf{c}_i \in \mathbb{R}^r$ and we stack all these opinions into a matrix $\boldsymbol{C}$, in which the i-th row corresponds to the opinion $\mathbf{c}_i^T$. Similarly, we represent the opinions held by each voter $j \in \mathcal{V}$ as $\boldsymbol{v}_j \in \mathbb{R}^r$ and stack all these opinions into a matrix $\boldsymbol{V}$, in which the j-th row corresponds to the opinion $\boldsymbol{v}_j^T$. Here, one can think of the dimension r as a measure of the complexity of the online discussion—along how many different *axis* can the opinions expressed in the comments and the opinions held by the voters differ. Figure 3.1 illustrates the above definitions using a toy example.

	← $\mathcal{V}$ →		
	1	2	3
↑ 1	−	+	+
$\mathcal{C}$ 2	+	−	+
↓ 3	+	+	−

(a) Voting pattern 1

	← $\mathcal{V}$ →		
	1	2	3
↑ 1	+	+	+
$\mathcal{C}$ 2	+	−	+
↓ 3	+	+	+

(b) Voting pattern 2

Figure 3.2: Examples of unfeasible voting patterns under the proximity and directional voting models with unidimensional (scalar) real-valued representation of opinions.

Voting model. Given a comment i which expresses an opinion $\mathbf{c}_i$ and a voter j who holds an opinion $\boldsymbol{v}_j$, we introduce two voting models, one deterministic and another probabilistic, inspired by the directional model of voting discussed above.

— *Deterministic voting model:* In this model, we can uniquely determine each vote y_{ij} from the comment's opinion $\mathbf{c}_i$ and voter's opinion $\boldsymbol{v}_j$ by means of the following deterministic rule:

$$y_{ij} = \begin{cases} + & \text{if } \langle \mathbf{c}_i, \boldsymbol{v}_j \rangle \geq 0 \\ - & \text{if } \langle \mathbf{c}_i, \boldsymbol{v}_j \rangle < 0. \end{cases} \tag{3.2}$$

In the above rule, the vote y_{ij} depends on the *angle* between the opinion vectors $\mathbf{c}_i$ and $\boldsymbol{v}_j$—if the angle is greater (less) than 90°, *i.e.*, $\mathbf{c}_i$ and $\boldsymbol{v}_j$ lie in the same (different) half-plane in the latent space, then $y_{ij} = +\,(-)$.

Under this voting model and a partially observed sign-matrix S derived from votes using Eq. 3.1, two natural question emerge:

1. What is the minimum dimension r of the latent space needed to recover the observed entries in S from the above decision rule without errors?
2. Once we know the minimum dimension r, can we infer the opinion vectors $\mathbf{c}_i$ and $\boldsymbol{v}_j$?

We will answer both questions affirmatively in Section 3.1.2 and 3.1.3, respectively.

— *Probabilistic voting model:* In the definition of our deterministic model, we have implicitly assumed that voters do not make any *errors* while casting their votes. However, this assumption might be rather restrictive in some scenarios. To overcome this, we also propose a probabilistic voting model in which votes are binary random variables Y_{ij}, and,

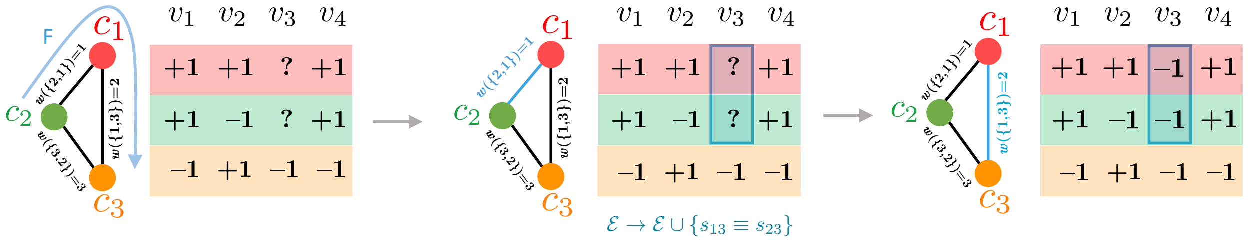
$$\mathbb{P}[Y_{ij} = y_{ij}] = p(y_{ij}) = \frac{1}{1 + \exp(-s_{ij} \langle \mathbf{c}_i, \boldsymbol{v}_j \rangle)}, \tag{3.3}$$

where $s_{ij} = +1$ if $y_{ij} = +$ and $s_{ij} = -1$ if $y_{ij} = -$. Similarly, as in the case of the deterministic model, we will propose a method to infer the opinion vectors $\mathbf{c}_i$ and $\boldsymbol{v}_j$ under this model in Section 3.1.3. In doing so, we will make the assumption that all the latent opinions are finite, *i.e.*, $\exists\, \alpha > 0.\, ||\boldsymbol{C}||_\infty \leq \alpha \wedge ||\boldsymbol{V}||_\infty \leq \alpha$.

Remark. In the above model definitions, we opt for a similarity metric based on dot products because the euclidean distance, used in the proximity model, does not *scale* well with increasing dimensionality: the relative *volume* of the opinion space where a voter will cast an upvote is proportional to $(\theta/\alpha)^r$ where θ is the threshold for the user, r is the dimension of the latent space and α is the upper bound on the opinion values.

3.1.2 Complexity of discussions

In this section, we present an algorithm which can determine an upper bound on the minimum dimension r that a latent space of opinions needs to have so that $\boldsymbol{C}$ and $\boldsymbol{V}$ are able to *explain* voting patterns



Case 2

Case 1

Figure 3.3: Illustration of Case 1 and Case 2 while selecting the edges of the minimum spanning tree which will help determine the permutation of the rows of the matrix that minimizes $SC^*(\boldsymbol{S})$. First, the edge (c_1, c_2) is selected and constraint $s_{13} \equiv s_{23}$ is added to $\mathcal{E}$. Next, the edge (c_1, c_3) is selected and s_{13} and s_{23} are filled with $s_{33} = -$.

exhibited by the voters which result in a particular vote-matrix $\boldsymbol{S}$ under the deterministic voting model, *i.e.*, $\forall\,(i,j) \in \Omega,\; s_{ij} = \operatorname{sign}(m_{ij})$, where $[m_{ij}] = \boldsymbol{C}\boldsymbol{V}^T$. To this aim, we will first introduce the notion of sign-rank of a sign matrix. Then, we will show that the problem of determining r reduces to finding the sign-rank of a partially observed sign matrix. Finally, we will present an efficient algorithm to estimate the sign-rank.

Sign-rank of a sign matrix. Paturi and Simon [1984] introduced the classical notion of sign-rank of a sign matrix, which is closely related to the VC dimension of concept classes Alon et al. [2016], as follows:

Definition 1. *Let $\boldsymbol{M}$ be a real matrix and* $\operatorname{sign}(\boldsymbol{M})$ *denote a matrix such that* $\forall\, i,j.\ (\operatorname{sign}(\boldsymbol{M}))_{ij} = \operatorname{sign}(\boldsymbol{M}_{ij})$. *Then, the sign-rank of a sign matrix $\boldsymbol{S}$ is defined as:*

$$\text{sign-rank}(\boldsymbol{S}) = \min\left\{rank(\boldsymbol{M}) \mid \operatorname{sign}(\boldsymbol{M}) = \boldsymbol{S}\right\}.$$

Here, we extend the above definition to partially observed sign matrices as follows:

Definition 2. *The sign rank of a partially observed sign matrix $\boldsymbol{S}$ is defined as:*

$$\text{sign-rank}(\boldsymbol{S}) = \min\left\{rank(\boldsymbol{M}) \mid \forall\,(i,j) \in \Omega.\ \operatorname{sign}(\boldsymbol{M})_{ij} = s_{ij}\right\}.$$

If the rank of a matrix $\boldsymbol{M}$ is r, then we can decompose the matrix into two components of the form $\boldsymbol{C}\boldsymbol{V}^T$ using, *e.g.*, the singular value decomposition (SVD). Hence, the problem of determining r reduces to the problem of finding sign-rank($\boldsymbol{S}$).

Note that the sign-rank of a sign matrix can be much lower than its actual rank, as was noticed by Hsieh et al. [2012] in the context of signed graph models. For example, consider the sign-rank of the matrix $\boldsymbol{B} = 2\mathbb{I}_n - 1_n$, where $\mathbb{I}_n$ is the identity matrix and 1_n is the matrix of all 1 of size $n \times n$. For $n \geq 3$, sign-rank($\boldsymbol{B}$) remains 3 though the matrix itself is always of full rank n. Moreover, note that, in our setting, the sign-rank does not merely correspond to the *number* of topics being discussed in an online discussion. Instead, the complexity may be manifest in the combination of the topics under discussion: the voters may agree with some opinions in a comment while disagreeing with others.

Estimating the sign-rank of a partially observed sign matrix. The problem of determining whether sign-rank($\boldsymbol{S}$) is 1 can be solved by a simple breath-first search (BFS). We first create a signed bi-partite graph of comments and voters with adjacency matrix $\boldsymbol{S}$. Then for each connected component in the graph, pick one $(i,j) \in \Omega$, set $\mathbf{c}_i = +1$ and $\boldsymbol{v}_j = s_{ij}$, and fill in the remaining values using BFS by multiplying the source node value with the sign of the edge to arrive at the destination node value. The intuition is that if voter j has down (up) voted comment i, then i and j have opposite (same) polarity. If a consistent assignment of ± 1 to all the nodes is possible, then sign-rank($\boldsymbol{S}$) $= 1$.

However, this algorithm does not generalize to multiple dimensions. To estimate the sign-rank of a partially observed sign matrix, we adapt the algorithm for (fully observed) sign matrices proposed recently by Alon et al. [2016]. First, we explain the main ideas behind the original algorithm and then describe the necessary, non trivial modifications we propose.

The original algorithm upper-bounds the sign-rank of a (fully observed) sign matrix $\boldsymbol{S}$ by the number of *sign-changes* in the columns of the matrix. More formally, define the function $SC(\boldsymbol{S})$ as the maximum number of sign changes in any column of the matrix $\boldsymbol{S}$, *i.e.*, $SC(\boldsymbol{S}) = \max_j |\{\, i \,|\, s_{i,j} \neq s_{(i+1),j} \}|$, $Sym(\boldsymbol{S})$ as the set of all possible row permutations of $\boldsymbol{S}$, and the function $SC^*(\boldsymbol{S})$ as:

$$SC^*(\boldsymbol{S}) = \min_{\boldsymbol{S}' \in \, Sym(\boldsymbol{S})} SC(\boldsymbol{S}').$$

Then, the following lemma establishes the relationship between the sign-rank of matrix $\boldsymbol{S}$ and $SC^*(\boldsymbol{S})$, which the original algorithm exploits Alon et al. [1985]:

Lemma 3. *For a sign matrix $\boldsymbol{S}$,* sign-rank$(\boldsymbol{S}) \leq SC^*(\boldsymbol{S}) + 1$.

To use the above result, our algorithm needs to do two tasks:

- Find a matrix $\bar{\boldsymbol{S}} = [\bar{s}_{ij}]$ such that it is a *completion* of $\boldsymbol{S}$, *i.e.*,

$$\bar{s}'_{ij} = \begin{cases} s_{ij} & \text{if } (i,j) \in \Omega \\ \pm 1 & \text{if } (i,j) \notin \Omega. \end{cases} \tag{3.4}$$

- Find $\boldsymbol{S}' \in Sym(\bar{\boldsymbol{S}})$ such that it minimizes the maximum number of sign-changes in its columns.

The algorithm will output $SC(\boldsymbol{S}') + 1$ as the estimated sign-rank.

Our algorithm does both tasks together while it computes an estimation of $SC^*(\boldsymbol{S})$ using an algorithm by Welzl *et al.* [Welzl, 1988, See Ex. 4]. In a nutshell, we construct a graph in which each node corresponds to a row of the partially observed matrix $\boldsymbol{S}$ and the weight of each edge between nodes u and v is given by the number of columns where the signs of the corresponding rows disagree. Then, we extract a spanning tree from the completely connected graph which minimizes the number of sign-changes between pairs of vertices connected by an edge, as shown in Algorithm 1. In the process of creating the spanning tree, we also fill the matrix. Finally, we derive a permutation of the rows based on the tree to construct $\boldsymbol{S}'$.

More in detail, to understand how Algorithm 1 fills the matrix as it computes the spanning tree, we distinguish two different cases:

Case 1: When, given a column, only one of the rows has a missing entry. Consider, for example, we have two rows $u = (+1, +1, \ ? \ , +1)$ and $v = (-1, +1, -1, -1)$. To calculate the weight of the edge between u and v, *i.e.*, $w(\{u, v\})$ in line 4 of Algorithm 1, we ignore the second column, as $s_{u,2} = s_{v,2}$, and the third column, as it has a missing entry $s_{u,3} = \ ?$. Hence, we report $w(\{u, v\}) = 2$, because signs of u and v differ in 1st and 4th column. Now, if this edge was chosen in line 6 of Algorithm 1, we *modify* u, such that this weight indeed is the true weight of the edge: we replace $s_{u,3} = \ ?$ with the corresponding value in v, *i.e.*, $s_{v,3} = -1$, via line 5 in Algorithm 2.

Case 2: When, given a column, both entries are unknown. If $u = (+1, +1, \ ? \ , +1)$ and $v = (+1, -1, \ ? \ , +1)$, we would still calculate the weight the same way as above. Hence, $w(\{u, v\}) = 1$. However, if this edge was chosen in line 6 of Algorithm 1, we could keep the weight the same by merely ensuring that both u and v have the *same* value in the third column, *i.e.*, $s_{u,3} \equiv s_{v,3}$. Hence, we create and save the constraint that the third column of u and v must always have the same value in line 9 of Algorithm 2. Now, say a few steps into the creation of the spanning tree, we find that the missing value in u has to be set to -1 as it was being picked as part of an edge under Case 1 above. Then, we can also set the same value in the third column of v, *i.e.*, $s_{v,3} \leftarrow +1$, via line 5 in Algorithm 2. Note that since each column in $\boldsymbol{S}$ contains at

Algorithm 1 Construct a sign-minimizing spanning tree for the columns of $\boldsymbol{S}$.

```
Input: A $[|\mathcal{C}|] \times [|\mathcal{V}|]$ sign-matrix $\boldsymbol{S} = [s_{ij}]$
Output: Spanning-tree of the rows.
$l \leftarrow 0$;   $F \leftarrow \{\}$;   $w : \mathbb{Z}^2 \rightarrow \mathbb{R}$;   $Z \leftarrow [|\mathcal{C}|]$;   $Y \leftarrow [|\mathcal{V}|]$
while $l < |\mathcal{C}|$ do
    for $u \in Z,\ v \in [|\mathcal{C}|]$; $(u, v) \cup F$ is cycle-free do
        $w(\{u, v\})$ += $|\{j \in Y \mid s_{uj} \neq s_{vj} \ \wedge\ s_{uj} \neq\ ?\ \wedge\ s_{vj} \neq\ ?\ \}|$
    end for
    $\{u^*, v^*\} \leftarrow \operatorname{argmin} w(\cdot)$;   $\boldsymbol{S}, Y, Z \leftarrow$ UPDATE$(\{u^*, v^*\}, \boldsymbol{S})$
    $F \leftarrow F \cup (u^*, v^*)$;   $l \leftarrow l + 1$;
end while
return   $F$
```

Algorithm 2 UPDATE procedure used in Algorithm 1.

Operation $\xleftarrow[Y,Z]{\mathcal{E}}$ assigns RHS to all entries equivalent to the LHS in $\mathcal{E}$ (line 9) and updates Y, Z.

```
Input: $\{u, v\}$ edge chosen; and $\boldsymbol{S} = [s_{ij}]$ sign-matrix;
Output: Updated $\boldsymbol{S}$ and all rows/columns with updated signs.
 1: // $\mathcal{E}$ initialized as an empty set and persisted across calls.
 2: $Z \leftarrow \{\}$;   $Y \leftarrow \{\}$
 3: for $j \in [|\mathcal{V}|]$ do
 4:     if $s_{uj} =\ ?$ and $s_{vj} \neq\ ?$ then
 5:         $s_{uj} \xleftarrow[Y,Z]{\mathcal{E}} s_{vj}$
 6:     else if $s_{uj} \neq\ ?$ and $s_{vj} =\ ?$ then
 7:         $s_{vj} \xleftarrow[Y,Z]{\mathcal{E}} s_{uj}$
 8:     else if $s_{uj} =\ ?$ and $s_{vj} =\ ?$ then
 9:         $\mathcal{E} \leftarrow \mathcal{E} \cup (s_{uj} \equiv s_{vj})$
10:     end if
11: end for
12: return   $\boldsymbol{S}$, $Y$, $Z$
```

least 1 entry which is ±1, as each voter has voted at least once, we will eventually hit Case 1 and fill in all missing entries. This process is illustrated in Figure 3.3, where the first step creates an equivalence $s_{13} \equiv s_{23}$, and the second step fills in the missing entires with s_{33}.

Note that we are *conservative* and greedy while filling in the missing entries, *i.e.*, the edge selected at the lth step will have the same *weight* $w(e)$ at the lth iteration if the algorithm was to be run on the filled matrix and it will be the minimum weight amongst all valid edges at step l (though the edge may not be unique in having that weight). Additionally, our algorithm ensures that the weight of the edge selected at iteration l is the minimum possible, given the history of selections. After obtaining a spanning tree, one can *walk* the tree by performing a depth-first search starting from any source node and create a permutation of the rows by dropping the duplicate nodes in the walk. Hence, we can obtain $\boldsymbol{S}' \in Sym(\boldsymbol{S})$ and report $SC(\boldsymbol{S}') + 1$ as r.

Then, we can establish the upper bound on the dimension by using the following series of self-evident inequalities: sign-rank$(\boldsymbol{S}) \leq$ sign-rank$(\boldsymbol{S}') \leq SC^*(\boldsymbol{S}') + 1 \leq SC(\boldsymbol{S}') + 1$.

Finally, we would like to highlight that the spanning tree algorithm presented above minimizes the *average* number of sign-changes in $\boldsymbol{S}'$. Welzl *et al.* Welzl [1988] also describe a variant of the algorithm which produces guarantees on the worst case number of sign-changes in $\boldsymbol{S}'$; the way the *weight* $w(\cdot)$ is calculated is more involved in the variant. This variant was used by Alon *et al.* Alon et al. [2016] to design the first polynomial time algorithm with approximation guarantees for the sign-rank of the matrix $\boldsymbol{S}'$. Remarkably, the UPDATE procedure in Algorithm 2 can be ported to that variant without any changes, to complete a partially observed matrix $\boldsymbol{S}$ matrix with worst case guarantees as well. However, that version is computationally more expensive, more complex, and does not offer significantly better results in practice in our dataset. Hence, for ease of exposition, we have described the simpler of the two versions.

Computational complexity. The computational complexity of Algorithm 1 can be determined by the computations needed for each missing element in the matrix $\boldsymbol{S}$. Except on the first initialization iteration ($l = 0$), the loop on line 3 will be executed once for each missing element (u, i) in the initial $\boldsymbol{S}$, right after s_{ui} is fixed via UPDATE. Hence, the work done for each missing entry, in the worst case, will be incrementing $w(\cdot)$ by 1 $\forall v \in [|\mathcal{V}|]$ such as $s_{uj} \neq s_{vj}$ in the loop on line 3. This can be done in $\mathcal{O}(|\mathcal{C}| \log |\mathcal{C}|)$ time if $w(\cdot)$ is implemented as a priority queue. As there are at most $|\mathcal{C}| \times |\mathcal{V}|$ missing entries, the computational cost of Algorithm 1 is $\mathcal{O}(|\mathcal{C}|^2 |\mathcal{V}| \log |\mathcal{C}|)$. All other operations, *i.e.*, initialization of $w(\cdot)$, calculating argmin, checking for cycles in the tree, the creation of an walk, maintaining $\mathcal{E}$, *etc.*, have lower complexity.

Remark. In our implementation, we do a (non-exhaustive) search over walks with different sources to improve our estimate of r and break ties in calculating the $\operatorname{argmin} w(\cdot)$ randomly. Also, as sign-rank$(\boldsymbol{S})$ = sign-rank$(\boldsymbol{S}^T)$, we run the algorithm on both matrices and report the smaller value.

3.1.3 Estimating multisided opinions

Given an online discussion, we infer the corresponding r-dimensional opinions $\boldsymbol{C}$ and $\boldsymbol{V}$ for the deterministic and probabilistic voting models introduced in Section 3.1.1 as follows.

Deterministic voting model. By definition, under the deterministic voting model, we know that the corresponding r-dimensional opinions $\boldsymbol{V}$ and $\boldsymbol{C}$ and the partially observed sign matrix $\boldsymbol{S}$ need to satisfy the following inequalities:

$$\bigwedge_{(i,j)\in\Omega} s_{ij} \left(\sum_{k=1}^{r} c_{ik} \cdot v_{jk} \right) > 0 \tag{3.5}$$

where c_{ik} and v_{jk} are the k-th entry of the opinions $\mathbf{c}_i$ and $\boldsymbol{v}_j$, respectively, and Ω is the set of observed entries in $\boldsymbol{S}$. However, we also know from Section 3.1.2 that, for each voting pattern, there will be a minimum dimension $r_{\min}$ under which such an opinion embedding will not exist.

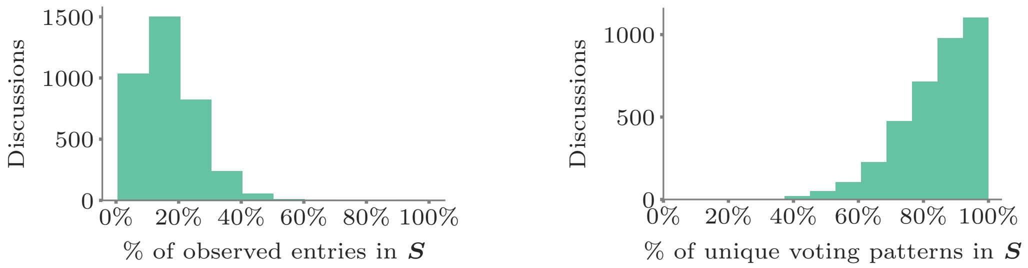
This reduces the problem of finding the opinions $\boldsymbol{V}$ and $\boldsymbol{C}$ to the existential theory of reals [Tarski, 1998] and, for small values of r and moderate number of comments and voters, $|\mathcal{C}|$ and $|\mathcal{V}|$, this problem can be solved via a procedure called quantifier elimination [Jovanović and de Moura, 2012], using, *e.g.*, the solver Z3 [De Moura and Bjørner, 2008]. This procedure eliminates the inequalities in the disjunction in Eq. 3.5 one by one by discovering subsets of $\mathbb{R}^{r\times(|\mathcal{V}|+|\mathcal{C}|)}$ where they are satisfied, backtracking to select a different region when an inequality cannot be satisfied, and using sound heuristics to prune the search. The procedure terminates when *any* assignment to each variable is found or when all regions of $\mathbb{R}^{r\times(|\mathcal{V}|+|\mathcal{C}|)}$ are eliminated.

Note that, if $r < r_{\min}$, the solver will conclude that the problem is unsatisfiable. Hence, by iteratively increasing r and checking for satisfiability of Eq. 3.5, one could determine the true sign-rank of any matrix $\boldsymbol{S}$. However, as the most efficient method known for quantifier elimination is doubly exponential in the number of variables, calculating the minimum dimension r in this way would be computationally more expensive than using the polynomial algorithm introduced in Section 3.1.2.

Probabilistic voting model. Given a partially observed matrix $\boldsymbol{S}$, under the probabilistic voting model, we estimate $\boldsymbol{V}$ and $\boldsymbol{C}$ by solving the following constrained maximum likelihood estimation problem with hyperparameters α:

$$\begin{aligned} \underset{\boldsymbol{C}, \boldsymbol{V}}{\text{maximize}} \quad & -\sum_{(i,j)\in\Omega} \log\Big(1 + \exp\big(-s_{ij}\,\langle \mathbf{c}_i, \boldsymbol{v}_j\rangle\big)\Big) \\ \text{subject to} \quad & ||\boldsymbol{C}||_\infty \le \alpha, ||\boldsymbol{V}||_\infty \le \alpha. \end{aligned} \tag{3.6}$$

The structure of the above problem allows us to adapt an efficient 1-bit matrix completion method based on stochastic gradient descent [Bhaskar and Javanmard, 2015]. Finally, note that unlike in the deterministic



(a) Sparsity of votes cast

(b) Unique voting patterns

Figure 3.4: Distribution of number of observed elements and fraction of unique voting patterns in matrix $\boldsymbol{S}$.

Dim.	Discuss.	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{V}\rvert$	Patterns
1^*	1,139	19.7 ± 16.7	19.9 ± 20.1	16.2 ± 14.8
2^*	2,820	51.5 ± 85.4	57.6 ± 75.2	46.0 ± 57.3
3	97	43.7 ± 35.3	56.0 ± 34.5	45.4 ± 24.6
4	88	148 ± 147	195 ± 194	149 ± 125
5	245	247 ± 272	296 ± 267	218 ± 179
6	126	445 ± 461	470 ± 366	354 ± 275
7	89	598 ± 555	706 ± 518	512 ± 355
≥ 8	112	2596 ± 2867	2785 ± 2540	1831 ± 1602

Table 3.1: Number of comments, voters and unique voting patterns seen in the dataset for discussions with different dimensions. The numbers in each column are the mean values ± the standard deviation. Dimensions marked with * indicate that they were determined using Z3 and are the true dimensions of the discussions. While the dimension of discussions is positively correlated with the size and participants, discussions of different complexity can be found on the entire spectrum, as is also shown in Figure 3.5.

model, for each voting pattern and dimension r, there will always exist opinions $\boldsymbol{C}$ and $\boldsymbol{V}$ that best fit the data.

Remark. In both models, the estimated opinions are unique up-to orthogonal transformations since the inequalities in Eq. 3.5 and the likelihood in Eq. 3.6 only depend on entries of $\boldsymbol{C}\boldsymbol{V}^T$ and $(\boldsymbol{C}\boldsymbol{O})(\boldsymbol{V}\boldsymbol{O})^T = \boldsymbol{C}(\boldsymbol{O}\boldsymbol{O}^T)\boldsymbol{V}^T = \boldsymbol{C}\boldsymbol{V}^T$ for any orthogonal matrix $\boldsymbol{O}$.

3.1.4 Experiments

Data description. Our dataset contains ∼19,800 online discussions, each associated to an article from Yahoo! News (including contributed articles), Yahoo! Finance, Yahoo! Sports, and the Newsroom app, which contain ∼5 million votes, cast by ∼200,000 voters on ∼685,000 comments, posted by ∼151,000 users. These votes were randomly sampled from all votes which were cast on comments made by users in the US on August 8, 2017.

As a pre-processing step, we discard discussions with less than 10 comments, as they contain too little data to provide meaningful results. After this step, our dataset consists of 4,700 discussions, with ∼4.5 million votes, cast by ∼199,000 voters, on ∼645,000 comments, posted by ∼137,000 users. Figure 3.4 shows the *richness* of the data in the votes gathered in the online discussions by means of the sparsity of $\boldsymbol{S}$ and the number of unique columns of $\boldsymbol{S}$, which we name as *voting patterns*.

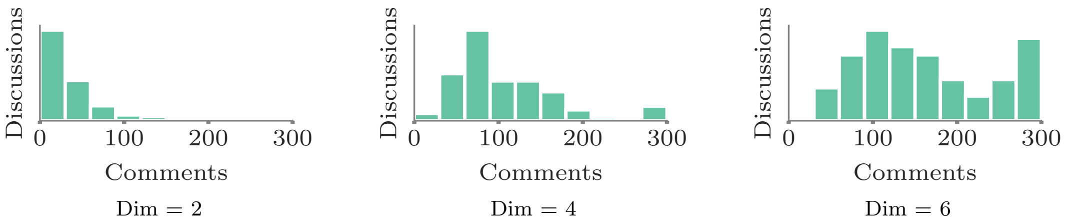
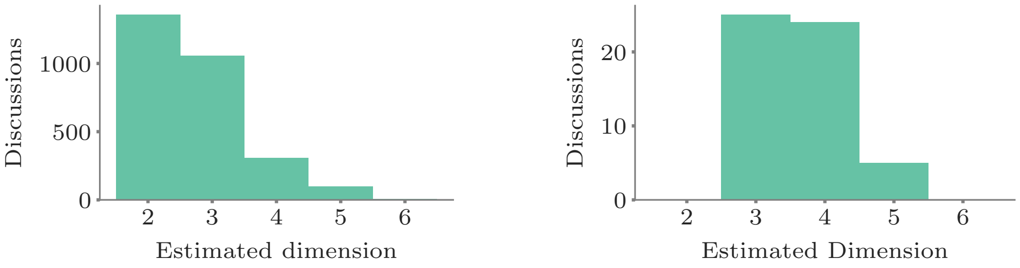


Figure 3.5: Distributions of the number of comments $|\mathcal{C}|$ per discussion for different dimension values (Dim). The distributions of the number of voters and voting patterns exhibit similar behavior.



(a) Discussions with dimension 2

(b) Discussions with dimension 3

Figure 3.6: Performance of our dimensionality estimation algorithm. For discussions whose opinions can be explained using two (three) dimensions, our algorithm recovers the true dimension for 48% (46%) of the discussions and is off by one for 37% (44%) of them.

Complexity of discussions. In this section, we compute the complexity of the discussions, *i.e.*, the dimensionality of the latent space of opinions, for the online discussions in our dataset. For each online discussion, we determine whether it can be explained using an unidimensional space of opinions using the linear time algorithm presented at the beginning of Section 3.1.2. If it cannot be explained using one dimension, we determine whether it can be explained using a two- or three-dimensional space of opinions via quantifier elimination[2], following Section 3.1.3. Finally, if it cannot be explained using two or three dimension, we resort to the algorithm presented in Section 3.1.2, which provides an upper bound on the true dimension.[3].

Table 3.1 summarizes the results, which show that the opinions of about 1,139 (25%) of the discussions can be explained using one dimension, 2,820 (60%) of the discussions require two dimensions, while the remaining 757 discussions (15%) require a higher number of dimensions. This allows us to conclude that the opinions in most of the online daily discussions (85%) can be explained using a latent space of relatively low dimensions, *i.e.*, $r \leq 3$. Moreover, while discussions with a higher number of participants ($|\mathcal{C}| + |\mathcal{V}|$) and richness (*i.e.*, higher number of voting patterns and lower sparsity) require, in general, a latent space of opinions with a larger number of dimensions, there is a large variability spanning the entire spectrum of online discussions, as shown in Figure 3.5. Next, we evaluate how tight is the upper bound on the true dimension provided by our algorithm for online discussions, which we used above for discussions whose dimension we could not find using quantifier elimination. To this aim, we run our algorithm on discussions whose true dimension we could find using quantifier elimination and compare the upper bound with the true dimension. Figure 3.6 summarizes the results, which show that, for discussions whose opinions can be explained using two (three) dimensions, our algorithm recovers the true dimension for 48% (46%) of the discussions and is off by one for

---

[2] In practice, we found quantifier elimination to be sufficiently scalable to test whether an online discussion can be explained using up to two dimensions.

[3] The source code for the experiments is available at https://github.com/Networks-Learning/discussion-complexity

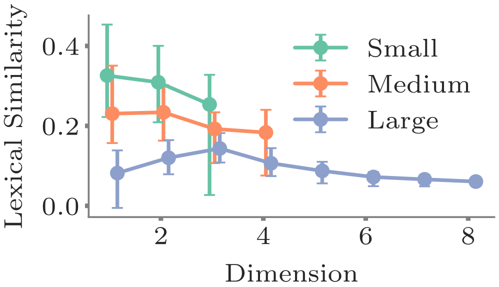
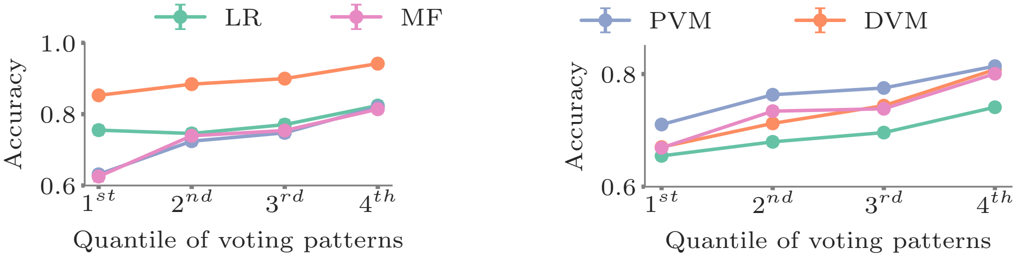


Figure 3.7: Lexical similarity of an online discussion against its dimension. The lexical similarity is the mean Jaccard similarity of the lexical tokens used in all pairs of comments in the discussion. The discussions are classified by the number of comments as small (smallest 33%), medium, and large (largest 33%) and data is shown only for dimensions which contain more than 5 discussions.



(a) Discussions with dimension 1

(b) Discussions with dimension 2

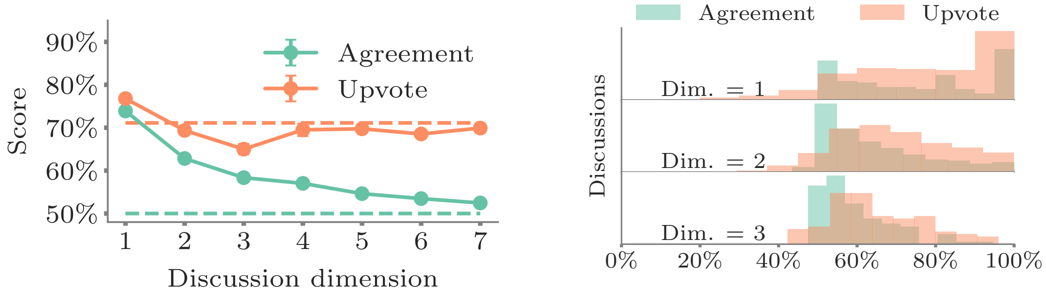
Figure 3.8: Vote prediction accuracy for the deterministic voting model (DVM), the probabilistic voting model (PVM), a state of the art matrix factorization method [Mazumder et al., 2010] (MF), and a logistic regression classifier [Murphy, 2012a] (LR) using textual features extracted using Rake [Rose et al., 2010]. Moreover, the performance for DVM, PVM and MR uniformly increases as the number of unique voting patterns increases, in contrast, the performance for LR remains relatively constant.

37% (44%) of them.

Finally, we investigate the relationship between the complexity of the discussions, estimated using human judgments, and their linguistic diversity, estimated using textual features. To this aim, for each online discussion, we compute the average Jaccard similarity of the lexical tokens used in all pairs of comments as a measure of lexical similarity. Figure 3.7 summarizes the results, which show a positive correlation between the complexity of a discussion and its linguistic diversity, as one may have expected.

Opinions in online discussions. In this section, we first evaluate both quantitatively and qualitatively the *quality* of the estimated r-dimensional opinions in the online discussions and then leverage the estimated opinions to shed some light on the level of controversy in online discussions. Here, we used the opinion estimation method for the probabilistic voting model introduced in Section 3.1.3, which scales graciously with the dimension r.

In terms of quantitative evaluation, we assess to which extent the deterministic voting model (DVM) and the probabilistic voting model (PVM) can predict whether a voter will upvote or downvote a comment from the estimated opinions in comparison with two baseline methods: (i) a state of the art matrix factorization method [Mazumder et al., 2010] (MF), which assumes the entries in $\boldsymbol{S}$ are real valued, and (ii) a logistic regression classifier [Murphy, 2012a] (LR) that uses 200,000 keywords extracted using Rake [Rose et al.,



(a) Mutual agreement and upvotes

(b) Agreement/upvote distribution

Figure 3.9: Agreement and percentage of upvotes among all votes in discussions. Agreement is measured in terms of percentage of comment pairs $(\mathbf{c}_i, \mathbf{c}_j)$ for which $\mathbf{c}_i^T \mathbf{c}_j > 0$. The dashed lines show the values of the metrics if the up/down votes or the latent dimensions were randomly distributed.

2010] as features. To this aim, for each discussion, we held out some of the observed upvotes and downvotes, estimate the opinions from the remaining votes, and then predict the votes from the held-out set. However, since our data is very sparse, as shown in Figure 3.4, and even holding out a small fraction of votes may change the underlying dimension of the latent space of opinions, we resort to leave-one-out validation. Moreover, we randomly select 200 discussions to tune the hyperparameters of PVM and these discussions are excluded from the validation set. LR was regularized using 10-fold cross-validation and MF was provided the true rank of the underlying matrix as input. Figure 3.8 summarizes the results, which show that:

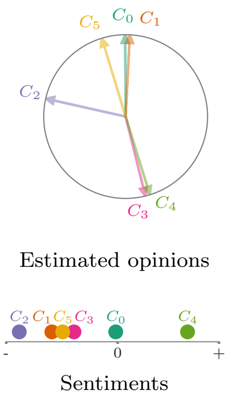
1. DVM (PVM) beats all other methods for discussions with dimension 1 (2).
2. The performance of DVM, PVM and MF increases as the number of unique voting patterns in the dataset increase, in contrast, the performance of LR, which uses text features, does not benefit much from additional voting patterns.
3. While for discussions where opinions can be explained using two dimensions, PVM achieves better performance, for discussions which require only one dimension, DVM beats PVM. A potential explanation for this behavior is that, whenever humans face simpler decisions, *i.e.*, their opinions can be explained using one dimension, they become more *predictable*.

In terms of qualitative evaluation, we first assess to which extent comments in online discussions agree (or disagree) by analyzing the estimated opinion embeddings of the comments. More specifically, for each online discussion, we compute the percentage of distinct comment pairs $(\mathbf{c}_i, \mathbf{c}_j)$ for which $\mathbf{c}_i^T \mathbf{c}_j > 0$ and compare this quantity with the percentage of upvotes among all votes (upvotes and downvotes). Figure 3.9a summarizes the results, which show that the higher the dimensionality of the latent space of opinions, the lower the agreement between comments, as one may have expected. Remarkably, such finding would not be apparent directly from the relatively constant fraction of upvotes. However, relative upvotes are typically the measure of *controversy* (or, rather, consensus) employed by various websites, like Reddit, to sort articles/comments.

Finally, we take a close look into the comments, inferred multidimensional opinions and unidimensional sentiment[4] of a discussion about politics, shown in Figure 3.10, and a discussion about finance, shown in Figure 3.11. The discussion about politics shows that, even if the lexical overlap between comments which express a similar opinion is low, *e.g.*, C_0 and C_2 or C_4 and C_5, our opinion estimation method is able to identify they are similar, as a human would do, by leveraging the judgments of the voters. Note that, due to their low lexical overlap, it would be difficult to identify such similarity using methods based on textual analysis, as revealed by the unidimensional sentiments. The discussion about the price of Twitter stock

---

[4] Comment sentiments were calculated using Convolutional Neural Networks trained on Stanford Sentiment Treebank [Kim, 2014, Socher et al., 2013].



C_0: [...] [Donald Trump] has [...] Enquirers[a] [which] he considers a treasure trove of information.

C_1: He should change his name to Donald J Dubious.

C_2: [...] Trump can be an #$%$, and Islam can be cancer [...] they are not mutually exclusive [...]

C_3: Why not? Try anything. Terrorism has got to stop now!

C_4: It is a great idea

C_5: Trump family motto-“It’s not a lie if you believe it.”

---

[a] *National Enquirers* is a well known entertainment magazine in US.

Figure 3.10: A subset of comments, estimated multidimensional opinions and unidimensional sentiments for an online discussion about politics. Two pairs of comments, *i.e.*, (C_0, C_1) and (C_3, C_4), express a similar opinion, however, the lexical overlap between comments within each pair is low. By leveraging the judgments of the voters, our method is able to identify they are similar, and their estimated opinions lie close to each other in the latent space of opinions, however, sentiment analysis is unable to do so and the unidimensional sentiments do not lie close to each other. Moreover, the estimated opinion of a comment expressing an opposite view to the ones above, C_2, lies in an orthogonal direction.

(see Figure 3.11) shows that our method is able to capture objective opinions about the price (whether it stays at $16 or goes up, C_3, or stays down, C_0 and C_1), along one axis and subjective opinions questioning the reason behind the price drop along a different axis (suggesting management is the reason, C_2, or media bias/corruption in Wall Street, C_4). Note that C_2 suggests both, that the price should go up, and that the reason for the decrease is the management. Also in this case, an analysis of the unidimensional sentiments would not reveal these rich relationships among the comments.

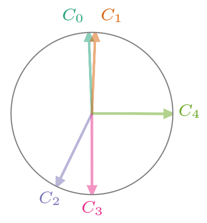
3.1.5 Conclusions

In this work, we have proposed a modeling framework to generate latent representations of opinions using human judgments, as measured by online voting. As a consequence, such representations exhibit a remarkable semantic property: if two opinions are close in the latent space of opinions, it is because the voters—the crowd—think that they are similar. Our modeling framework is theoretically grounded and establishes an unexplored, surprising connection between opinion and voting models and the sign-rank of a matrix. Moreover, it also provides a set of practical algorithms to both estimate the dimension of the latent space of opinions and infer where opinions expressed in comments and held by voters lie in this space. Experiments on a large dataset from Yahoo! News show that many discussions are multisided and avoid falling prey to demagoguery, provide insights into human judgments and opinions, and show that our framework is able to circumvent language nuances, *e.g.*, sarcasm and humor, by relying on human judgments.

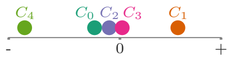
3.2 Understanding Evolution of Expertise

Previous section showed how one can model the up/down voting process during an online discussion and infer both the complexity of the discussion as well as the relative agreement among the comments. The complexity of the discussion was an indication of along how many dimensions could the users agree or disagree with each other, *i.e.*, a lower bound on the number of different *topics* were under discussion. This is particularly important for unstructured discussions on the Internet.

However, some online platforms have evolved to add more structure to the collaborative dissemination process. One such example are online Q&A forums, *e.g.*, Stack Overflow and Quora, where members of the crowd



Estimated opinions



Sentiments

C_0: You can forget about \$16 for a while.

C_1: Bye-Bye twitter sweet 16.

C_2: [...] when world leaders speak, they turn to Twitter first. [...] How is it trading at \$16? [...] How come Dorsey[a] can't monetize this instantaneous platform?

C_3: [...] It's about time [for] more positive news to get it [...] up again. [...] Seems to have support \$16ish. [...]

C_4: [Wall Street/CNBC] only want to pump [selective] stocks [...] Twitter of China, Weibo, is selling for \$88.00 a share [...]

[a] Jack Dorsey is the CEO of Twitter.

Figure 3.11: A subset of comments, estimated multidimensional opinions and unidimensional sentiments for an online discussion about finance. There are two distinct issues being discussed: (i) the price of a stock (C_0, C_1, C_2, C_3) and (ii) a discussion about reasons for the price (C_2, C_4). C_0 and C_1 say humorously that the price will stay below \$16, while C_2 and C_3 suggest that the price *may* rise up. C_4 suggests Wall Street/media bias against the stock and is neutral about its price, while C_2 questions the management of the company.

ask questions and members of the same crowd answer them. Curiously, the Q&A forums also use the same mechanism of up/down votes we see being used for indicating agreement/disagreement in online discussions. However, the purpose of having the up/down votes is very different. From the point of view of platforms, up/down votes are a means for the members of the crowd to *curate* the knowledge on the site for later consumption by other members. Hence, not only is the content on the forums *crowdsourced*, the platforms are also designed for *crowdlearning*.

In this section, a *crowdlearning* model is developed which can be inferred by leverage the same voting process but finding more nuanced signals in it. The model will help in:

1. better understanding how people learn over time and become experts;
2. identifying questions with high knowledge value, which systematically help users increase their expertise; and
3. investigating the interplay between learners and contributors on the platform.

A probabilistic generative model of crowdlearning, especially designed to fit fine-grained crowdlearning event data [Aalen et al., 2008b] is proposed in this thesis. The key idea behind the modeling framework is simple: every time users learn from a *knowledge item* contributed by other users, they may increase their expertise and, as a consequence, their subsequent contributions will be more knowledgeable and assessed more highly by others in terms of, *e.g.*, upvotes, likes or shares. Thus, by jointly modeling *learning events*, in which users acquire *effective knowledge*, and *contributing events* (in short, *contributions*), in which users contribute with their expertise to a knowledge item, the framework will reach the above mentioned goals. The aim of the model is to measure those aspects of the learning process for which there is evidence in the observed data, *i.e.*, a measure of *effective knowledge* that leads to measurable increase in users' *effective learning*.

In more detail, each user's expertise is modeled as a latent stochastic process that evolves over time and the other users' assessment of her contributions are interpreted as noisy samples from this stochastic process localized in time. Moreover, this stochastic process is driven by two types of learning: off-site learning and on-site learning. The proposed formulation also captures characteristic properties of the learning process, previously studied in the literature, such as forgetting [Loftus, 1985] and initial expertise [Posnett et al., 2012]. Then I develop an efficient parameter estimation method that finds the model parameters that maximize the likelihood of an observed set of learning and contributing events using convex optimization. Finally, we will see the effectiveness of the model by tracing learning and contributing events in data gathered from Stack

Overflow over a 4.5 year period. The experiments reveal several interesting insights:

1. The knowledge value of items follow a log-normal distribution.
2. Users with very low or very high initial expertise, *i.e.*, newbies and experts, tend to increase their knowledge the least and, in contrast, users in the middle range tend to increase it the most. This suggests that the learning curve may be sigmoidal, in agreement with existing literature [Leibowitz et al., 2010].
3. Although there are fewer contributors than learners in absolute numbers, the distribution of knowledge in the contributions is fat tailed while the distribution of knowledge learned is heavy tailed.
4. Users who learn from high knowledge items are also more proficient at providing answers with high knowledge value.

3.2.1 A Crowdlearning Model

In a crowdlearning site, users often play two different functional roles: *contributors* and *learners*. In the former role, they share their knowledge on a topic (or topics) with other users within the site; and, in the latter role, they acquire knowledge by reading what other users contributed to the site. Then, we can think of users' expertise as latent stochastic processes that evolve over time, and think of the assessments of their contributions to the site as noisy samples from these stochastic processes localized in time. Here, we propose a modeling framework that uncovers the evolution of these processes by modeling two types of learning:

1. *Off-site learning*, which accounts for the knowledge that the user accumulates outside the site; and,
2. *On-site learning*, which accounts for the knowledge that the user gains by reading other users' contributions within the site.

Next, we formulate our generative model, starting from the data it is design for.

Crowdlearning data. Given a crowdlearning site with a set of users $\mathcal{U}$ and a set of learning areas (or topics) $\mathcal{A}$, we first define a knowledge item q as the smallest quantum of knowledge a user can learn from within the site. For example, in a Q&A site, a knowledge item corresponds to a question and its answer(s); in Twitter, it corresponds to a tweet; and in a wiki site, it corresponds to a wiki page. Intuitively, each knowledge item q provides certain (latent) knowledge value, $k_q \in \mathbb{R}^+$, and contains knowledge about a subset of topics $\mathcal{A}_q \in \mathcal{A}$. Here, we assume that knowledge is additive, *i.e.*, $k_q = \sum_{a \in \mathcal{A}_q} k_{qa} = \boldsymbol{w}_a^T \boldsymbol{k_q}$, where $k_{qa} \in \mathbb{R}^+$ is the knowledge value contained in item q about topic a, $\boldsymbol{k_q} = [k_{qa}]_{a \in \mathcal{A}}$, and $w_{qa} = 1$ if $a \in \mathcal{A}_q$ and $w_{qa} = 0$, otherwise. The model can be extended to non-binary weights to represent fractional presence of topics in a knowledge item [Blei et al., 2003].

Then, we define two types of events: *learning events*, in which users acquire knowledge by reading contributions by other users, and *contributing events* (or *contributions*), in which users contribute to the crowd by sharing their knowledge. Formally, we represent each learning event as a triplet

$$l \;:=\; \Big(\, \underset{\text{user}}{\underset{\uparrow}{u}},\quad \underset{\text{time}}{\underset{\uparrow}{t}},\quad \overset{\text{knowledge item}}{\overset{\downarrow}{q}} \,\Big), \tag{3.7}$$

which means that a user $u \in \mathcal{U}$ *learned* from knowledge item q at time t. Here, a knowledge item q may contain one or more contributions by other users. For example, in a Q&A site, a knowledge item corresponds to a question and its answers, typically contributed by different users. In a learning event, we do not distinguish the knowledge provided by individual contributions, but instead, consider the knowledge of the item as a whole. Moreover, note that, if the knowledge value of an item is zero, the learning event will not increase the expertise of the learner. Then, we denote the history of learning events associated to user u up to time t by $\mathcal{H}_u^l(t) = \bigcup_{i:t_i<t} \{l_i \mid u_i = u\}$, and the history of learning events in the whole crowdlearning site up to time t by $\mathcal{H}^l(t) = \bigcup_{i:t_i<t} \{l_i\}$.

Similarly, we represent each contribution as a quadruplet

$$c \;:=\; (\underset{\text{user}}{\underset{\uparrow}{u}},\; \underset{\text{time}}{\underset{\uparrow}{t}},\; \overset{\text{knowledge item}}{\overset{\downarrow}{q}},\; \underset{\text{score}}{\underset{\uparrow}{s}}), \tag{3.8}$$

which means that a user $u \in \mathcal{U}$ contributed to a knowledge item q at time t, and other users assigned a score s to her contribution. For example, in a Q&A site, this may be the number of upvotes an answer receives. We gather the history of contributions in the whole crowdlearning site up to time t by $\mathcal{H}^c(t) = \bigcup_{i:t_i<t} \{c_i\}$, and the history of contributions and learning events up to time t by $\mathcal{H}(t) = \mathcal{H}^c(t) \bigcup \mathcal{H}^l(t)$.

Crowdlearning generative process. We represent each user's expertise as a multidimensional (latent) stochastic process $\boldsymbol{e}^*_u(t)$, in which the a-th entry, $e^*_{ua}(t) \in \mathbb{R}^+$, represents the user u's expertise on topic a at time t. Here, the sign * means that the expertise $e^*_{ua}(t)$ depends on her learning history $\mathcal{H}^l_u(t)$. Then, every time a user u contributes to a knowledge item q at time t, we draw the contribution's score from a distribution $p(s|\mathcal{A}_q, \boldsymbol{e}^*_u(t))$. Further, we represent the times of the learning and contributing events within the site by two sets of counting processes, denoted by two vectors $\boldsymbol{N}^l(t)$ and $\boldsymbol{N}^c(t)$, in which the u-th entries, $N^l_u(t)$ and $N^c_u(t)$, count the number of times user u learned from and contributed to the crowdlearning site up to time t. Then, we can characterize these counting processes using their corresponding intensities as

$$\mathbb{E}[d\boldsymbol{N}^l(t) \,|\, \mathcal{H}(t)] = \boldsymbol{\lambda}^l(t)\, dt \;\text{ and }\; \mathbb{E}[d\boldsymbol{N}^c(t) \,|\, \mathcal{H}(t)] = \boldsymbol{\lambda}^c(t)\, dt$$

where $d\boldsymbol{N}^l(t) := [dN^l_u(t)]_{u\in\mathcal{U}}$ and $d\boldsymbol{N}^c(t) := [dN^c_u(t)]_{u\in\mathcal{U}}$ denote the number of learning and contributing events in the window $[t, t+dt)$ and $\boldsymbol{\lambda}^l(t) := [\lambda^l_u(t)]_{u\in\mathcal{U}}$ and $\boldsymbol{\lambda}^c(t) := [\lambda^c_u(t)]_{u\in\mathcal{U}}$ denote the vector of intensities associated to all the users. Here, there is a wide variety of intensity functions one can choose from [Aalen et al., 2008b]. However, modeling the times of learning and contributing events is not the main focus of this work – we refer the reader to the growing literature on social activity modeling using point processes [Farajtabar et al., 2014, Valera and Gomez-Rodriguez, 2015, Zhou et al., 2013]. Next, we specify the functional form of each user's expertise $\boldsymbol{e}^*_u(t)$ and the score distribution $p(s|\mathcal{A}_q, \boldsymbol{e}^*_u(t))$.

Stochastic process for expertise. The expertise $e^*_{ua}(t)$ of a user u on a topic a at time t takes the following form:

$$e^*_{ua}(t) := \overbrace{\alpha_{ua}}^{\text{initial expertise}} + \underbrace{\mu_{ua} \cdot t}_{\text{off-site learning}} + \overbrace{\sum_{i:q_i\in\mathcal{H}^l_u(t)} k_{q_i a} \cdot \kappa_\omega\,(t - t_i)}^{\text{on-site learning}}$$

where the first term, $\alpha_{ua} \in \mathbb{R}^+$, models the initial expertise of user u on a topic a when she joined the crowdlearning site; the second term, $\mu_{ua} \in \mathbb{R}^+$, assumes a linear trend for the off-site learning process as a first order approximation[5]; and, the third term models the knowledge a user acquires by means of learning events within the crowdlearning site. Here, $\kappa_\omega(t)$ is a nonnegative kernel function that models the rate at which users forget the knowledge they learn from knowledge items. Following previous work on the psychology literature [Averell and Heathcote, 2011, Loftus, 1985], which argues that people forget at an exponential rate, we opt for an exponential kernel $\kappa_\omega(t) := \exp(-\omega t)\mathbb{I}(t \geq 0)$. However, our model estimation method does not depend on this particular choice.

For compactness, we write each user's expertise as a row vector of length $|\mathcal{A}|$, *i.e.*,

$$\boldsymbol{e}^*_u(t) = \boldsymbol{\alpha_u} + \boldsymbol{\mu_u} \cdot t + \sum_{i:q_i\in\mathcal{H}^l_u(t)} \boldsymbol{k_{q_i}} \cdot \kappa_\omega(t - t_i) \tag{3.9}$$

---

[5] Several other shapes for the learning curve have been proposed in Heathcote et al. [2000]. However, we chose the linear form for its simplicity and ease in model estimation, as suggested by Skinner [2010].

where $\boldsymbol{\alpha_u} = [\alpha_{ua}]_{a\in\mathcal{A}}$, $\boldsymbol{\mu_u} = [\mu_{ua}]_{a\in\mathcal{A}}$ and $\boldsymbol{k_{q_i}} = [k_{q_i a}]_{a\in\mathcal{A}}$. Here, by definition, $k_{q_i a} = 0$ if $a \notin \mathcal{A}_{q_i}$. Then, we can gather the model parameters for all users in three matrices $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$ and $\boldsymbol{k}$ with sizes $|\mathcal{U}| \times |\mathcal{A}|$, $|\mathcal{U}| \times |\mathcal{A}|$ and $|\mathcal{Q}| \times |\mathcal{A}|$.

Score distribution. Given a contribution $c = (u, t, q, s)$, the particular choice of score distribution $p(s|\mathcal{A}_q, \boldsymbol{e}^*_u(t))$ depends on the observed data. In this work, we consider discrete non-negative scores, $s \in \mathbb{N}$, which fit well several scenarios of interest. For example, in Stack Overflow, scores may correspond to the number of upvotes that answers receive; in Twitter, to the number of likes or retweets that tweets receive; and, in Pinterest, to the repins that a pin receives. A natural choice in such cases is the Poisson distribution:

$$p(s|\mathcal{A}_q, \boldsymbol{e}^*_u(t)) \sim \text{Poisson}\left(\frac{\boldsymbol{w}_q^T \boldsymbol{e}^*_u(t)}{\boldsymbol{w}_q^T \mathbf{1}}\right), \tag{3.10}$$

Here, $\mathbf{1}$ is a column vector of ones with length $|\mathcal{A}|$. With this choice, the average of the score distribution is simply the average expertise of user u at time t across the topics $\mathcal{A}_q$ the knowledge item q is about. Moreover, the greater the expertise of a user, the greater the scores given by other users to her contributions, as one may expect in real-world data.

Note that depending on the recorded data, we could choose a different score distribution, *e.g.*, for continuous assessments like time elapsed between the question and the answer, one may choose a continuous distribution. Our model estimation method can be easily adapted to any distribution that is jointly log-concave with respect to the model parameters $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$ and $\boldsymbol{k}$.

Efficient Parameter Estimation. Given a collection of learning and contributing events, $\mathcal{H}^l(T)$ and $\mathcal{H}^c(T)$, recorded during a time period $[0, T)$, we find the optimal model parameters $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$ and $\boldsymbol{k}$ by solving the following maximum likelihood estimation problem:

$$\underset{\boldsymbol{\alpha}\geq 0, \boldsymbol{\mu}\geq 0, \boldsymbol{k}\geq 0}{\text{maximize}} \; \mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\mu}, \boldsymbol{k}), \tag{3.11}$$

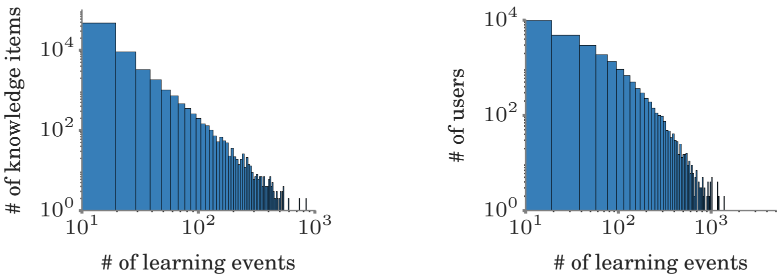
where we compute the log-likelihood $\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\mu}, \boldsymbol{k})$ using Eq. 3.9 and Eq. 3.10, *i.e.*,

$$\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{\mu}, \boldsymbol{k}) = \sum_{\substack{(u,t,q,s) \\ \in \mathcal{H}^c(T)}} s \cdot \log\left(\frac{\boldsymbol{w}_q^T \boldsymbol{e}^*_u(t)}{\boldsymbol{w}_q^T \mathbf{1}}\right) - \frac{\boldsymbol{w}_q^T \boldsymbol{e}^*_u(t)}{\boldsymbol{w}_q^T \mathbf{1}}. \tag{3.12}$$

Since $\boldsymbol{e}^*_u(t)$ is linear in the model parameters $\boldsymbol{\alpha}, \boldsymbol{\mu}$ and $\boldsymbol{k}$, the function $\log x - x$ is concave, and a composition of a linear function with a linear combination of concave functions is concave, the optimization problem above is jointly convex in $\boldsymbol{\alpha}$, $\boldsymbol{\mu}$, and $\boldsymbol{k}$. As a consequence, the global optimum can be efficiently found by many algorithms. In practice, the limited memory BFGS with bounded variables (L-BFGS-B) algorithm [Zhu et al., 1997] worked best for our problem.

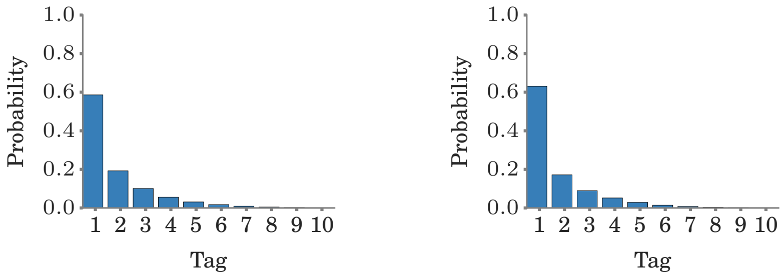
Remarks. In this work, we are measuring *effective learning*, which accounts for the ability of a user to get better assessment of her posts, and *effective knowledge*, which accounts for the gain in this ability that learning from a knowledge item causes. Making these quantities correspond to real-life expertise and knowledge value on a crowdlearning website requires careful mapping from the features on that website to learning events and scores.

Moreover, using our model, one can only measure learning and knowledge if there is overlap between the topics of a user's learning and contributing events. Therefore, there is a trade-off between the granularity of the topics chosen and the amount of data available for inference: increasing the granularity ensures accurate mappings between learning and contributing events, but reduces the amount of data available to learn the model parameters. We discuss this further in Section 3.2.4.



(a) LE per question

(b) LE per user



(c) QT per user

(d) AT per user

Figure 3.12: Statistics of learning events (LE), question tags (QT) and answer tags (AT) in the Stack Overflow dataset. In Panels (c) and (d), the x-axis denotes the tag index in order of popularity for each user.

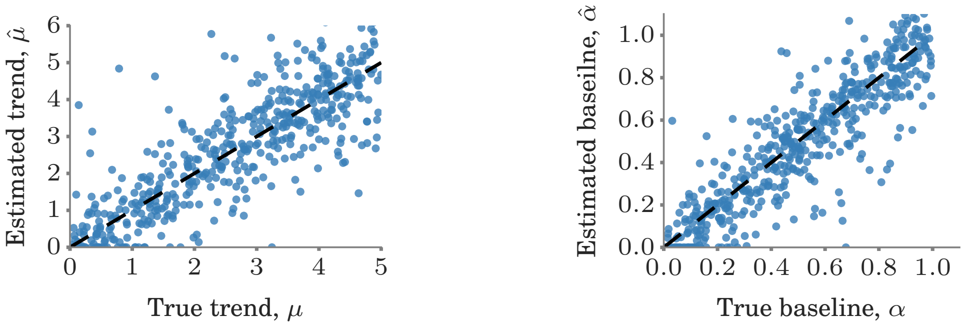
3.2.2 Experiments on Synthetic Data

In this section, we first show that our model estimation method can accurately recover the true model parameters from learning and contributing events generated under *realistic* conditions. We then show that, as long as there are a sufficient number of contributions *per* learning event, the estimation becomes more accurate as we feed more events into the estimation procedure. Finally, we show that our estimation method can easily scale up to millions of users, knowledge items, and learning and contributing events.

Experimental setup. We carefully craft an experimental setup to closely mimic some of the empirical patterns observed in real crowdlearning data, as given by Figure 3.12. Here, for simplicity, we assume the topics associated to each knowledge item are specified by means of tags.

Given a set of users and knowledge items, we draw the users' offsite learning rates $\{\mu_{ua}\}$ and initial expertise $\{\alpha_{ua}\}$ from $U(0,5)$ and $U(0,1)$, and the knowledge value of the items from the rescaled log-normal distribution $0.05 \times \ln \mathcal{N}(0,1)$. These choices ensure that the distribution of scores which users receive resembles the distribution in real data. We set the users' forgetting decay rate to $\omega = (11.6 \text{ days})^{-1}$, such that 50% of the knowledge is forgotten roughly after the first week, and assume that the intensities of both users' learning and contributing events are (homogeneous) Poisson processes. We denote the total simulation time by T. We set each user's learning event rate to T/n, where n is drawn from a log-normal distribution, so that the number of events per user fits well the empirical distribution (see Figure 3.12b), and each user's contributing rate to T/m, where m is drawn from an uniform distribution for easy control. Finally, for each user, we shuffle the tag labels and set her tag learning propensity, defined as the probability that she up-votes a knowledge item with a given tag, and her tag contributing propensity, defined as the probability that she contributes to a knowledge item with a given tag, using the empirical distributions (see Figures 3.12c and 3.12d).

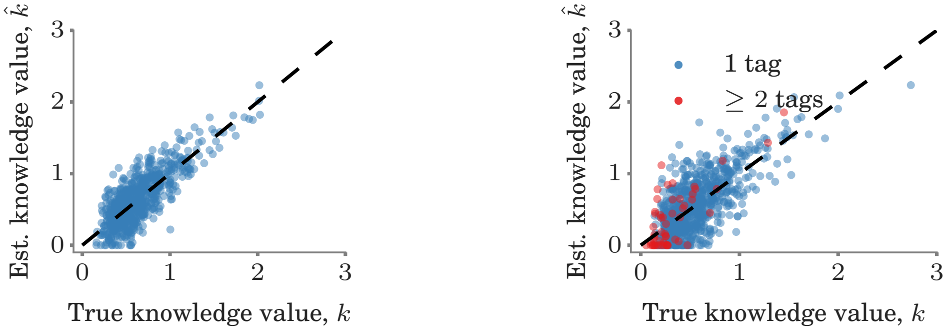
Then, we generate learning and contributing events as follows. First, we generate the timings of each user's



(a) $\boldsymbol{\mu}$ (1-tag)

(b) $\boldsymbol{\alpha}$ (1-tag)

Figure 3.13: Estimated (y-axis) against true (x-axis) model parameters for the 1-tag synthetic datasets. Each point corresponds to a user's (a) trend μ_u or (b) baseline α_u variable, and the line defined by $x = y$ corresponds to zero estimation error. Our estimation method achieves a Spearman's correlation $\rho_\mu = 0.82$ and $\rho_\alpha = 0.89$. The results for 10-tag synthetic datasets are qualitatively similar.



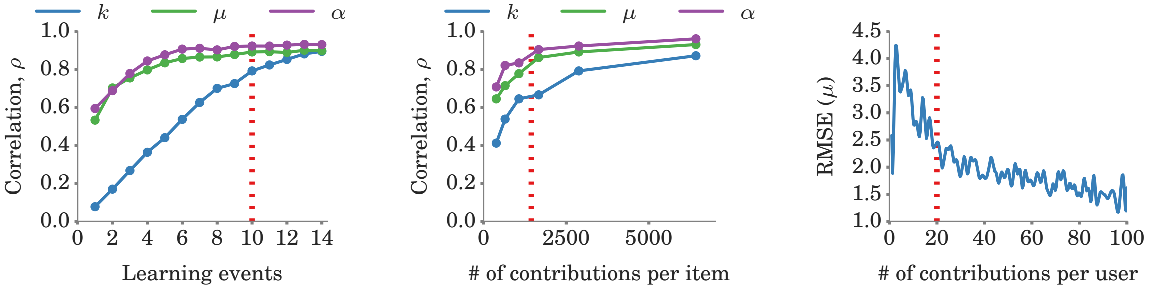
(a) 1 tag

(b) 10 tags

Figure 3.14: Estimated (y-axis) against true (x-axis) knowledge item values. Each point corresponds to a knowledge item variable, and the line defined by $x = y$ corresponds to zero estimation error. Our estimation method achieves Spearman's correlations $\rho_{\text{1-tag}} = 0.74$ and $\rho_{\text{10-tag}} = 0.64$.

learning events by drawing samples from the corresponding Poisson process, and assign each learning event to a knowledge item such that the user's tag learning propensity is satisfied. Then, we generate the timings of each user's contributions by drawing samples from the corresponding Poisson process, assign each contributing event to a knowledge item such that the user's tag contributing propensity is satisfied, and draw the quality score from a Poisson distribution that depends on the user's expertise on the item tags at the time of the event, as given by Eq. (3.10). Unless explicitly stated, we only consider knowledge items with at least 10 associated learning events. Given this data, our goal is to find the knowledge value of the items users learned from, as well as the users' offsite learning rates and initial expertise by solving the maximum likelihood estimation problem defined in Eq. (3.11).

Parameter estimation accuracy. We evaluate the accuracy of our model estimation procedure across all users and knowledge items for a 1- and 10-tag dataset with ∼800 knowledge items. Figure 3.14 summarizes the results for the estimation of the knowledge item values by means of two scatter plots. In all cases, we find that points lie close to the line $x = y$, *i.e.*, their estimation error is close to zero. We also observe that the estimation of knowledge items in the 10-tag dataset is more challenging than in the 1-tag dataset. Additionally, Figure 3.13 summarizes the results for the estimation of the user's expertise baseline and trend



(a) CR vs. learning events (b) CR vs. contributions (c) RMSE vs. contributions

Figure 3.15: Estimated against true model parameters for the 1-tag synthetic dataset. Panels (a) and (b) show the correlation (CR) between the estimated and true model parameters against number of learning events and median number of contributions per knowledge item, respectively. Panel (c) shows the RMSE for the estimated trends, $\boldsymbol{\mu}$, against number of contributed events per user. In Panel (a), the number of contributing events is 255,000 and the red dotted line shows the threshold (10) we chose for the learning events per knowledge item in our dataset (see Section 3.2.3). In Panels (b) and (c), the number of learning events is always 13,000 and the red dotted lines show the median number of contributions per knowledge item and the minimum number of contributions per user in the experiments on our dataset, respectively (see Section 3.2.3).

variable for the 1-tag dataset using scatter plots. Results for the 10-tag dataset are qualitatively similar although the estimation is more challenging. In particular, if we look at the estimation of the knowledge values, trends and baseline for the 1-tag dataset, our estimation method achieves a Spearman's correlations $\rho_k = 0.74$, $\rho_\mu = 0.82$ and $\rho_\alpha = 0.89$ while, for the 10-tag dataset, it achieves $\rho_k = 0.64$, $\rho_\mu = 0.76$ and $\rho_\alpha = 0.81$. This is most likely due to the mixing of tag knowledge variables within the same knowledge item, *i.e.*, the linear combination of knowledge variables in Eq. (3.12).

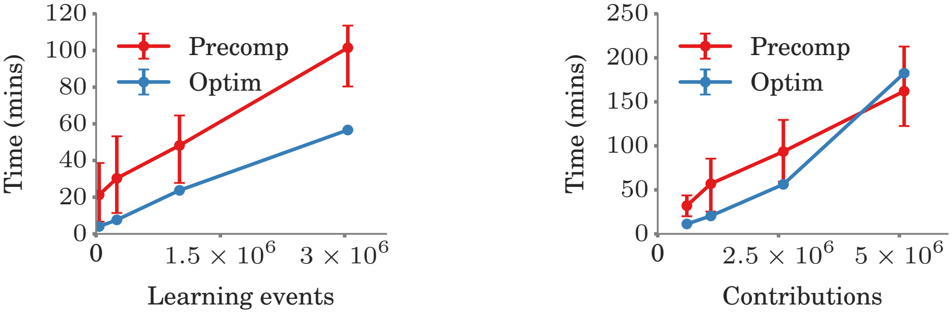
Parameter estimation accuracy vs. number of learning events. In our model, we can think of learning events as measurements of the amount of knowledge in a knowledge item, which are accumulated over time in the users' expertise, and of contributing events as noisy samples of the users' expertise at particular points in time. Therefore, intuitively, the more users learn from a knowledge item the easier it should become to accurately estimate their associated knowledge value, as long as these users also contribute to other knowledge items with overlapping topics. Figure 3.15a confirms this intuition by showing the Spearman's correlation against minimum number of learning events per knowledge item in a 1-tag dataset with 255,000 contributions.

Parameter estimation accuracy vs. number of contributing events. As pointed out above, we can think of the score of contributing events as noisy samples of users' expertise at particular points in time. Therefore, one may expect the accuracy of our model parameter estimation to improve as the number of contributions increases, due to a more fine-grained sampling of each user's expertise. Figure 3.15b gives empirical evidence that this indeed happens, by showing the Spearman's correlation against average number of answers *per* learning event in a 1-tag dataset with 13,000 learning events. Figure 3.15c shows how the RMSE of the estimation of $\boldsymbol{\mu}$ decreases as the number of contributions made by the user increases.

Scalability of parameter estimation. Crowd-learning sites such as Stack Overflow or AskReddit are rapidly increasing their number of active users, questions and answers. For example, Stack Overflow recently crossed the ∼10 million questions mark[6]. The pre-computation of all coefficients in Eq. (3.12), which is the running time bottleneck, can be readily parallelized. Figure 3.16 shows that our model estimation method

---

[6] http://meta.stackoverflow.com/questions/303045/10-million-questions



(a) RT vs. learning events

(b) RT vs. contributions

Figure 3.16: Running time (RT) of our model estimation method. In Panel (a), we consider ∼2 million contributing events while varying number of learning events (and knowledge items); in Panel (b), we consider ∼1.8 million learning events while varying the number of contributions (per learner). For pre-processing, we used ten machines with 48 cores and, for the optimization itself, we used a single machine with 48 cores. The memory requirements were below 16 GB at all points of the pre-processing and optimization.

easily handles up to millions of learning events and contributing events, and scales almost linearly with the number of learning events and contributions.

Thus, it should be possible to scale up our estimation method even further.

3.2.3 Experiments on Real data

In this section, we apply our model estimation method to a large-scale crowdlearning dataset from Stack Overflow. First, we evaluate our model quantitatively by means of a prediction task: given two different answers to a question, predict which one will receive a higher score. Then, we discuss the distribution of the knowledge values and the effect of the kernel parameter on the estimation, identify different types of learners and derive insights into their main characteristics. Finally, we study the interplay between learners and contributors in crowdlearning sites and investigate to which extent users switch between learning and contributing over time.

Data description. Our Stack Overflow dataset comprises ∼8 million questions, ∼13.7 million answers, and ∼47.2 million upvotes. These questions and answers were posted by ∼1.9 million users during a six year period, from the site's inception on July 31, 2008 to September 14, 2014. Importantly, for each upvote, our dataset contains its associated user identity, question or answer identity and timestamp[7]. We discard the events which happened before 2010-01-01 (before the site had fully matured) and after 2014-06-01 (the extent of the data-dump we had access to). Whenever in the data a user *upvotes* (*writes*) an answer, we record it as a learning (contributing) event involving the user and the knowledge item containing the answer. Moreover, we select the number of upvotes a user's answer received in the first week after posting it as the score of the contribution; downvotes were discarded because they constitute less than 3% of total votes cast. Here, we consider only the first week of voting to prevent old contributions from gaining an unfair advantage as they have more time to accumulate upvotes. Figure 3.12 provides general statistics on learning and contributing events and tags usage. We find that the learning events per user (per question) follow a log-normal (power-law) distribution. As shown, the tag usage is highly skewed towards few tags; most users contribute and learn only from their favorite tags.

Data preprocessing. In Section 3.2.2, we have shown that the accuracy of our estimation method depends

---

[7] Stack Overflow generously gave us access to these additional metadata, which allows us to readily fit our model.

Score difference	# of pairs	Off-site only	Our model
$\geq$ 1.0	31,639	52.5%	**61.9%**
$\geq$ 2.0	19,253	52.9%	**64.8%**
$\geq$ 3.0	10,804	53.2%	**67.0%**
$\geq$ 4.0	5,910	53.7%	**70.7%**
$\geq$ 5.0	3,250	55.0%	**71.6%**
$\geq$ 6.0	1,935	56.0%	**73.3%**
$\geq$ 7.0	1,159	56.8%	**73.8%**

Table 3.2: Performance of our model against a linear baseline model at predicting which one of two answers to a question will receive a higher score. As the difference in score between the answers (and, hence, the users' expertise) increases, the competitive advantage of our model becomes more pronounced.

dramatically on the number of learning and contributing events per question and user (refer to Figure 3.15). As a consequence, we can only expect our model estimation method to provide reliable and accurate results in real data if the data we start with contains enough learning and contributing events per question and user. To this aim, we carefully pre-process our large-scale dataset of learning and contributing events. We only consider:

(i) Knowledge items with more than 10 associated learning events, which corresponds to a correlation value ≥ 0.8 between true and estimated knowledge parameters in synthetic data, as shown in Figure 3.15a.

(ii) Users that contribute (answer) more than 20 times in at least 10 unique months, which corresponds to a RMSE value ≤ 2 for the estimated users' baseline and trend parameters in synthetic data, as shown in Figure 3.15c; and,

(iii) Top 10 tags in terms of number of learning events in the recorded data (*i.e.*, `java`, `c#`, `javascript`, `php`, `android`, `jquery`, `python`, `html`, `c++`, and `mysql`).

After these preprocessing steps, our dataset consists of ~25 thousand users who learn from ~66 thousand knowledge items by means of ~1.4 million learning events, and contribute to ~2.5 million knowledge items, by means of ~3.8 million contributing events. Then, we correct for the overall decreasing trend on number of upvotes per answer over time[8] and since, for each knowledge item, most learning events occur after all the contributions (answers) to the knowledge item took place, we assume its knowledge value to be constant. We use the first event of each user in our dataset as as an estimate of her joining time.

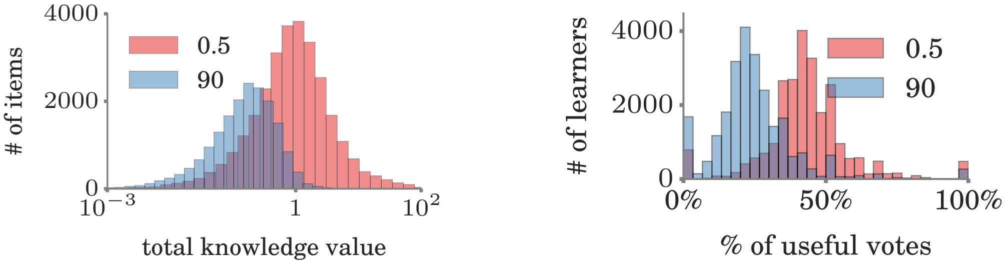
Finally, we would like to highlight that the preprocessing steps above do not aim to reduce the size of the original dataset but to increase the accuracy of our estimated model parameters and the reliability of our derived qualitative insights — our model estimation method does easily scale to millions of learning and contributing events. In this case, the pre-processing of the raw data using five machines with 48 cores each took ~30 minutes and our estimation method, implemented using the Intel MKL libraries, took ~11.5 hours on a single machine with 48 cores. The memory requirements were below 16 GB at all points of the pre-processing and optimization.

Quantitative evaluation. We evaluate our model quantitatively by means of the following prediction task: given two different answers to a question, predict which one will receive a higher score, *i.e.*, more number of upvotes in the week after posting it.

— *Experimental setup:* We train our model using the first 80% of the answers provided by each learner, as well as the learning events which occurred before them. Then, we match pairs of answers to the same questions from the remaining 20% and predict which one will receive a higher score. Here, we only consider questions

---

[8] The number of upvotes per answer decreases over time because the number of answers grows at a faster rate than the number of learners.



(a) Overall knowledge values

(b) Fraction of upvotes leading to learning

Figure 3.17: Estimated knowledge values for knowledge items and *useful upvotes* for two different kernel parameters, with *half-life* 0.5 days (12 hours) and 90 days. Panel (a) shows the distribution of knowledge value per knowledge item follows a log-normal distribution with longer half-life leading to smaller knowledge values and higher sparsity. Panel (b) shows what fraction of upvotes were useful (*i.e.*, led to learning) per learner: higher half-life leads to higher sparsity, which leads to fewer fraction of upvotes causing effective learning.

with pairs of answers provided by users from our dataset such that their scores differ by at least one upvote. There are ∼32 thousand such pairs in our dataset. Finally, we compare our model against a baseline linear model which only accounts for off-site learning to show the benefits of including knowledge item variables.

— *Results:* Table 3.2 compares the performance of our model against the baseline model as the difference in score between the answers (and, hence, the users' expertise) increases. Our model consistently outperforms the baseline for any score difference and the competitive advantage becomes more pronounced as the score difference increases, reaching >73% accuracy when the score difference is ≥6.

Knowledge value and forgetting rate. In this section, we leverage our model to give insights on the knowledge values across items in Stack Overflow for different forgetting rates, *i.e.*, the kernel decay parameter ω. We express the kernel decay parameter ω in units of *half-life* in days, *i.e.*, the time to forget 50% of the knowledge in an item. Figure 3.17a shows the distribution of estimated knowledge value across knowledge items for two kernel parameters with half-life 0.5 days and 90 days. We find several interesting patterns. Knowledge values in both settings follow a log-normal distribution, in which ∼10% of the items account for ∼75% of the overall knowledge. However, while for a half-life of 0.5 days, ∼53% of the knowledge items do not contribute knowledge, this fraction increases to ∼70% for 90 days. A potential explanation for this difference is that, by increasing the half-life, a knowledge item must show evidence of *effective learning* over longer stretches of time to contribute knowledge and this happens more rarely. As a consequence, a smaller fraction of upvotes lead to effective learning (*i.e.*, being *useful*) when the half-life is high, as shown in Figure 3.17b – when the half-life is 0.5 days (90 days), 42% (24%) of upvotes lead to learning.

In the remaining sections, for ease of exposition, we set the kernel parameter such that the half-life of knowledge is 7 days (refer Figure 3.18), however, the insights obtained in the following sections are robust to changes in the kernel parameter.

Types of learners. Here, our goal is to better understand the type of learners that use crowd-learning sites as well as their characteristic properties. To this aim, we start by visualizing the estimated learning trajectory for four different users — an average learner, an on-site learner, an off-site learner and an expert — in Figure 3.19. Each of the users exhibits different characteristic properties. For example, the average learner contributes answers with much less knowledge value (0.005) than the expert (0.034), and the on-site learner acquires 55% of the knowledge by learning from items in Stack Overflow in contrast with the off-site learner, who only learns 0.4% of the knowledge by those means.

Next, we investigate the interplay between on-site and off-site learning across all users. Here, given user u, we de-

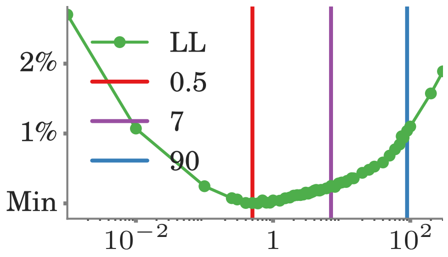
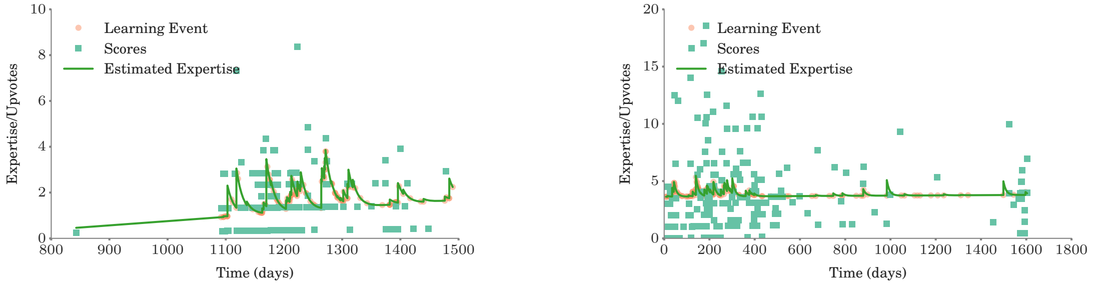
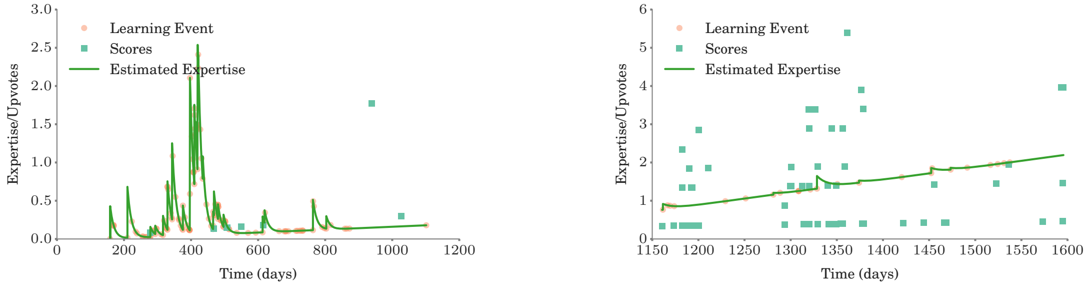


Figure 3.18: Negative log-likelihood plotted for different values of kernel parameter (expressed as half-life in days). The y-axis shows the relative difference with respect to the minimum value. The likelihood nearly plateaus (∼1%) for half-life between 0.5 and 90 days. The results we present are robust to parameter changing within the range and we chose 7 days as a representative value.



(a) Avg. learner (Avg. knowledge / contribution: 0.005)

(b) Expert: (Avg. knowledge / contribution: 0.034)

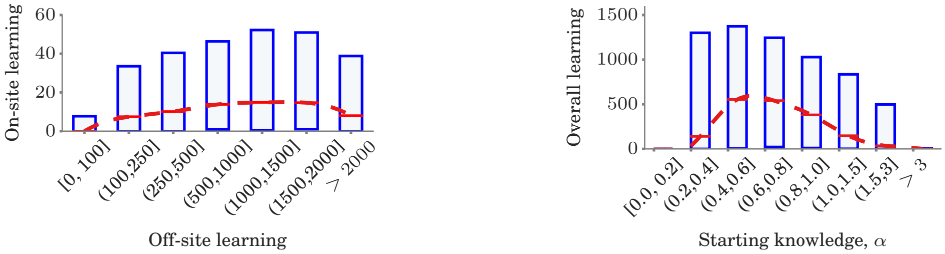


(c) On-site learner (on-site learning: 55%)

(d) Off-site learner (on-site learning: 0.4%)

Figure 3.19: Estimated learning trajectory for four characteristic Stack Overflow users. The (average) learner (a) contributes answers with much less knowledge value than the expert (b), *i.e.*, 0.005 vs 0.034. The on-site learner (c) acquires 55% of her knowledge by learning from items in Stack Overflow in contrast with the off-site learner (d), who only learns 0.4% of her knowledge by those means. Day 0 in the plots is the date 2010-01-01.

fine on-site learning as the total expertise gathered by reading the knowledge items, $\sum_{a\in\mathcal{A}}\sum_{q\in\mathcal{H}^{l}_{u}(T)}\int k_{qa}\kappa_{\omega}(t)\,dt$, off-site learning as the expertise gathered outside Stack Overflow, $\sum_{a\in\mathcal{A}}\int \mu_{ua}t\,dt$, and overall learning as the sum of both. One can think of these quantities as the aggregate number of upvotes (*i.e.*, score) users would have received on the site through either their on or off-site learning if they were posting answers at the same rate. Note that unlike the reputation on Stack Overflow, which is a measure of how much a user has effected others on the site, the on-site and off-site learning reflects how much a user has learned. Figure 3.20a compares users' on-site and off-site learning by means of a box plot. For $x \leq 2000$, users achieving higher on-site learning also achieve higher off-site learning, but over $x > 2000$, off-site learning becomes more dominant. Our results seem to indicate that quick learners rely less on on-site learning, in relative terms.



(a) On-site and off-site learning (b) Starting and learned expertise

Figure 3.20: Behavior of learners in for tag `c#`. Panel (a) compares users' on-site and off-site learning in a box plot. For $x \leq 2000$, users achieving higher on-site learning also achieve higher off-site learning, however, over $x > 2000$, users off-site learning becomes more dominant. Panel (b) shows users' overall learning against starting expertise in a box plot. Users with very low or very high initial expertise, *i.e.*, newbies and experts, tend to increase their knowledge the least, in contrast, users in the middle of the range tend to increase it the most. In both panels, the limits of the boxes are the 25%–75% percentiles and the red dashed lines shows the median value.

Finally, we investigate the role that a user's starting expertise plays on her overall learning over time by means of a box plot, shown in Figure 3.20b. Here, the x-axis corresponds to a user's starting expertise, α_u, and the y-axis to her overall learning. Interestingly, we find that users with very low or very high initial expertise, *i.e.*, newbies and experts, tend to increase their knowledge the least, in contrast, users in the middle of the range tend to increase it the most. This is in agreement with previous research, which indicated that in presence of only positive reinforcement, the gain in expertise has a sigmoidal shape for learners, *i.e.*, the newbies and experts increase their expertise at lower rates than learners with medium levels of expertise [Leibowitz et al., 2010].

Learners vs contributors. A crowd-learning site is only useful if it has both learners and contributors. Here, we investigate two natural questions that emerge in such context:

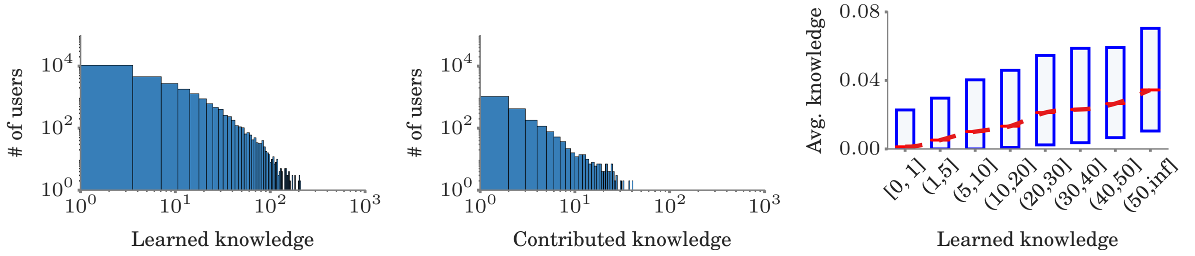
I. Are learners and contributors equally common?
II. Are more prolific learners better contributors?

To answer the first question, we compute the distribution of learned and contributed knowledge per user. Here, we estimate the knowledge value of each contribution (*i.e.*, answer) in a knowledge item by dividing the total knowledge item value across contributions proportionally to their quality scores (upvotes). Figures 3.21a and 3.21b summarize the results. Although, in absolute numbers, there are more learners than contributors in our dataset, the amount of knowledge fed into the site by the contributors shows higher variability than the knowledge learned by users – the distribution of contributed knowledge is fat tailed ($\alpha \approx 2.26$).

Next, we investigate the second question and assess whether more prolific learners are better contributors. To do so, we calculate the average knowledge value per contribution across users that have learned similar amount of knowledge over time, *i.e.*, sum of the knowledge value of all the knowledge items the user learned from, $\sum_{a\in\mathcal{A}}\sum_{q\in\mathcal{H}^{l}_{u}(T)} k_{qa}$. Figure 3.21c shows that the users that learn more knowledge are also more proficient at producing high knowledge contributions. In other words, our results suggest that "*by learning you will teach; by teaching you will learn.*"

3.2.4 Discussion

In this section, we take a step back and discuss the limitations of our model. First, we remark that, due to the large number of parameters in the model, it is necessary to have access to large amount of data for our model estimation method to be accurate. However, this limitation can be overcome, to some extent,



(a) Learned knowledge

(b) Contributed knowledge

(c) Avg. knowledge per contribution vs. learned knowledge

Figure 3.21: Learners vs contributors in Stack Overflow. Panels (a) and (b) show the distribution of overall learned and contributed knowledge per user. The former follows a log-normal distribution ($\mu \approx 1.39, \sigma \approx 1.09$), while the latter follows a power-law ($\alpha \approx 2.26, x_{min} \approx 1.84$). This shows that through the contributors are fewer in number than learners in absolute terms, they show much higher variability. Panel (c) shows a user's average knowledge value per contribution against overall learned knowledge in a box plot. The red dotted line shows the median values and the box limits are the 25%–75% percentiles. Interestingly, the users that learn more knowledge are also more proficient at producing high knowledge contributions.

by linking expertise of a user across different platforms or sites (*e.g.*, MOOCs), *i.e.*, our model can easily assimilate traces available for the same user from those sites.

In our model, it is also crucial that the score reflects the true assessment of the knowledge content of the item and not of, say, the popularity of the contributor. In the case of Stack Overflow, which is a strict and self-regulated community, upvotes are seldom granted to answers which do not address the question — cases of serial upvoting are caught and remedied quickly which (mostly) prevents users from voting as a *thank you* gesture. As a consequence, on Stack Overflow, upvotes on answers are a good assessment of the quality of the posts. However, a sensible choice for scores in platforms or sites with milder self-regulation may be challenging.

Finally, the learning events also need to be chosen such that they are not conflated with other objectives the user may have on the website. On Stack Overflow, if a user only upvotes a question, it indicates that she relates with the problem but none of the answers (if any) provide a solution. However, upvoting an answer is evidence that the Q&A pair *taught* the user something.

These unique features and mechanisms afforded by Stack Overflow allow us to easily identify learning events and assessments. Finding similar features in a different social network would require careful reasoning and justification.

3.2.5 Conclusion

In this chapter, we proposed a probabilistic model of crowdlearning, naturally designed to fit fine-grained learning and contributing event data. The key innovation of our model is modeling the evolution of users' expertise over time as a latent stochastic process, driven by both off-site and on-site learning. Then, we develop a scalable estimation method to fit the model parameters from millions of recorded learning events and contributions. Finally, we applied our model to a large set of learning and contributing events from Stack Overflow and found several interesting insights. For example, there is ample variation on the popularity of knowledge items (questions with their answers) with similar knowledge values and items with high knowledge value are rare. Newbies and experts acquire less knowledge than users in the middle range. Prolific learners tend to be also proficient contributors that share knowledge with high knowledge value.

Chapter 4

Competitive Knowledge and Information Dissemination

In the previous section, we proposed models for the process of knowledge and information dissemination in a collaborative setting. This section looks at the process of information and knowledge dissemination from the perspective of the content creators, where the users are *competing* with other users for the attention of their shared followers.

The process of dissemination of information on the internet changed in a fundamental manner with the advent of social-media sites like Twitter and Facebook. Unlike the more traditional setting studied in the previous sections, where the key focus of the sites is on the content, the new social-media sites make explicit follower-followee relationships their primary features. This allows users to *personalize* their corner of the internet by picking and choosing who they befriend in the virtual world. This, in turn, leads to a personalized *feed* of information which each person curates for themselves to the extent permitted by the increasingly algorithmic bent of the sites. This feed collects the content posted by the followees of the user (*i.e.*, the follower) and the user consumes them (or a part thereof) upon logging in, or in real time as push notifications. Recent empirical studies have suggested that the higher a post is on a follower's feed the more likely is the follower to interact with it and that users seldom reach information which is hidden *beyond a fold*, *i.e.*, requires scrolling before it becomes visible [Hodas and Lerman, 2012, Lerman and Hogg, 2014, Kang and Lerman, 2015, Spasojevic et al., 2015]. Hence, all broadcasters of information, *i.e.*, the followees of the follower, are truly competing with each other, vying for the attention of their respective followers. The nascent field of social-media marketing [Constantinides, 2014] is generally focused on creating viral content by means of adjusting the content of posts. However, the best content will not make an iota of difference if the posts are not visible on the feeds of the followers at all. This section looks at the problem of competitive dissemination at the granularity of a single broadcaster and device algorithmic solutions for the problem of estimating the visibility and controlling the time of posts.

In light of this competition, first, we explore the *when-to-post* problem in an idealized setting, *i.e.*, under the assumption that the broadcaster is notified of the change of visibility of his posts whenever the competitors make their posts in real time. This problem presents a perfect setting to further explore the competitive dissemination setting. Spasojevic et al. [2015] did the seminal empirical study of when was the "best" time to make posts on Twitter and Facebook, *i.e.*, time which elicits most likes/shares/comments/responses from other users. Since then, this problem has received some media attention from the popular media and has prompted research into algorithmic solutions to the problem under various assumptions and with different objectives [Karimi et al., 2016, Zarezade et al., 2017a]. Section 4.1 describes a general reinforcement learning based method which will be able to learn the best times to post. This work was done in collaboration with co-researchers at MPI-SWS presented at the Neural Information Processing Systems

(NeurIPS) conference [Upadhyay et al., 2018].

Section 4.2 considers the problem of keeping up with changing content online faced by broadcasters. In particular, I consider how to learn how a user's posts are performing on the feeds of the followers, *i.e.*, whether they are visible in the top-k or not, or what their position is on the feeds. However, the scale of the Internet, the restrictions placed on the user by the platform, *e.g.*, rate limited API, and the real-time dynamics of the feeds make this a challenging problem. I draw parallels between this challenge and the task which the web-crawlers face while indexing the entire the Internet and discuss approaches to solve this estimation problem in Section 4.2. This work will yield an algorithmic solution with theoretical guarantees which can help relax the idealized assumptions we had made while studying, *e.g.*, the when-to-post problem. This work was done with researchers from Yahoo! Research Labs and Poznan University. The results were presented at the 34th AAAI conference on Artificial Intelligence [Upadhyay et al., 2020].

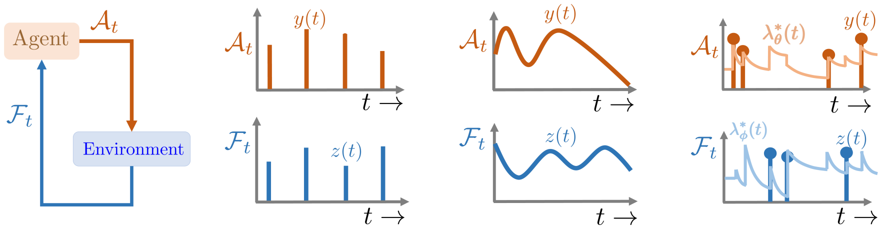
4.1 Deep Reinforcement Learning for *when-to-post* problem

Spasojevic et al. [2015] showed by means of an empirical study that certain times and week-days were better than others to make posts at when it came to receiving likes/shares/comments/responses from the other users. In this section, I will go a step further and frame this as a problem of competitive dissemination. This allows for a much more granular formulation which could be adapted for each broadcaster and will allow us to algorithmically determine the best posting times in an online fashion.

We use the framework of marked temporal point processes (MTPPs) [Aalen et al., 2008a] to model the posting behavior of our user (*i.e.*, our broadcaster) as well as the other users (*i.e.*, other broadcasters) on the network. In recent years, the framework of MTPPs has become increasingly popular for modeling asynchronous event data in continuous time in a wide range of application domains, from social and information networks to finance or health informatics. For example, in social and information networks, events may represent users' posts, clicks or likes; in finance, they may represent buying and selling orders; or, in health informatics, they may represent when a patient exhibits different symptoms or receives treatment. In most cases, the development of a new model reduces to the problem of designing an appropriate functional form for the conditional intensity (or intensities) of the events of interest as well as the distribution of the corresponding mark(s).

The *when-to-post* problem can be expressed with this formalism by representing the posting behavior of users as MTPPs. This representation was suggested by Karimi et al. [2016], though their formulation was closer to the setting explored by Spasojevic et al. [2015]. They came up with an offline method which determined the *intensity* of posting for each hour of the week which would have been optimal in the past for a post to appear at the top of the followers' feeds. However, one can do much more granular controlling of the posting rate by framing it as an online optimization problem. In this context, a recent line of work [Wang et al., 2018, 2017, Zarezade et al., 2018, 2017a] has exploited an alternative view of the dynamics of the MTPPs representing users' posting behavior as stochastic differential equations (SDEs) with jumps [Hanson, 2007] to design online, adaptive interventions using stochastic optimal control. These online approaches show significant improvement over the offline approach of Karimi et al. [2016]. While this line of work has shown promise at enhancing the functioning of social and information systems, their wide spread use and deployment is precluded mainly by two drawbacks. First, they make strong assumptions about the functional form of the conditional intensities and mark distributions of the MTPPs, which in turn prevent them from using state of the art MTPP models based on deep learning [Du et al., 2016, Mei and Eisner, 2017, Jing and Smola, 2017]. Second, the objective functions that the interventions optimize upon, need to be carefully chosen to ensure that the underlying stochastic optimal control problem remains tractable. As a consequence, the use of (more) meaningful objective functions with clear semantics is often off limits. In this work, these drawbacks are addressed by approaching the problem from the perspective of deep reinforcement learning of MTPPs.

More specifically, first a novel reinforcement learning problem is introduced where both the actions taken



Setup Discrete Continuous Point Process

Figure 4.1: Reinforcement learning setups. In the traditional discrete time setting [Sutton and Barto, 1998], actions and feedback occur in discrete time; in the continuous time setting [Doya, 2000], actions and feedback are real value functions in continuous time; and, in the marked temporal point process setting (our work), actions and feedback are asynchronous events localized in continuous time.

by an *agent* (*e.g.*, the posts made by the broadcaster we wish to help) and the feedback it receives from its *environment* (*e.g.*, posts of other broadcasters) are asynchronous stochastic events in continuous time, which are characterized using MTPPs. Here, the goal is finding the *optimal* intensity and mark distribution for the agent's actions—the optimal policy—that maximize an arbitrary reward function, which may depend on its actions and the feedback. Then, a novel policy gradient method is derived, specially designed to solve the above problem, which embeds the agent's actions and the feedback from the environment into real-valued vectors using deep recurrent neural networks (RNNs).

In contrast with the literature on stochastic optimal control of SDEs with jumps, this method does not make any assumptions on the functional form of the conditional intensity (or intensities) and mark distribution(s) characterizing the feedback, and it allows for arbitrarily complex reward functions.

Moreover, it departs from previous work in the reinforcement learning literature [Doya, 2000, Duan et al., 2016, Farajtabar et al., 2017, Frémaux et al., 2013, Lillicrap et al., 2015, Mnih et al., 2016, Sutton and Barto, 1998, Vasilaki et al., 2009, Wierstra et al., 2007] in two key aspects, which are also illustrated in Figure 4.1:

1. The broadcaster's actions (*i.e.*, posts) and environment's feedback (*i.e.*, the posts of the other broadcasters) are asynchronous stochastic events in continuous time. In contrast, previous work has considered synchronous actions and (potentially delayed) feedback in discrete time [Duan et al., 2016, Lillicrap et al., 2015, Mnih et al., 2016, Wierstra et al., 2007], with few notable exceptions [Doya, 2000, Frémaux et al., 2013, Vasilaki et al., 2009]. While these exceptions considered continuous time, they assumed actions and feedback to be continuous and deterministic and the dynamics of the environment to be known.[1]

2. The policy is a conditional intensity function (and a mark distribution), which is used to *sample* the times (and marks) of the agent's actions. Here, note that a sampled agent's action may need to be resampled due to the occurrence of new feedback events before the sampled time. In contrast, previous works considered the policy to be a probability distribution or, seldom, a deterministic function [Doya, 2000, Frémaux et al., 2013, Vasilaki et al., 2009].

The approach, named Temporal Point Process Reinforcement Learning (TPPRL), is general and can be applied to any setting where we can simulate or reply the actions of the environment and evaluate the reward.

4.1.1 Formulation

In this section, we first briefly revisit the theoretical framework of marked temporal point processes [Aalen et al., 2008a] and then use it to formally define our novel reinforcement learning problem, where an agent interacts with a complex environment by means of asynchronous stochastic discrete events in continuous time.

---

[1] This setting should not be confused with the *asynchronous* setting of Mnih et al. [2016], where the gradient descent is asynchronous but the action/observations are synchronous and the system evolves at discrete time steps.

Marked temporal point processes. A marked temporal point process (MTPP) is a random process whose realization consists of an ordered sequence of events localized in time, *i.e.*,

$$\mathcal{H} = \{e_0 = (t_0, z_0), e_1 = (t_1, z_1), \ldots, e_n = (t_n, z_n)\},$$

where $t_i \in \mathbb{R}^+$ is the time of occurrence of event $i \in \mathbb{Z}$ and $z_i \in \mathcal{Z}$ is the associated mark. The actual meaning of the events varies across applications, *e.g.* in social networks, t_i may represent the time when a message is posted, clicked or liked, z_i may represent the type of interaction, the message content, or its polarity, and the domain of the marks $\mathcal{Z}$ is application dependent. Here, we characterize the event times of a MTPP using a conditional intensity function $\lambda^*(t)$, which is the probability of observing an event in the time window $[t, t + dt)$ given the events history $\mathcal{H}_t = \{e_i = (t_i, z_i) \in \mathcal{H} \,|\, t_i < t\}$, *i.e.*,

$$\lambda^*(t) := \mathbb{P}\{\text{event in } [t, t + dt) \,|\, \mathcal{H}_t\}, \tag{4.1}$$

where the sign * means that the intensity may depend on the history $\mathcal{H}_t$. Moreover, we characterize the marks of the events using a distribution $m(z \,|\, \mathcal{H}_t) = m^*(z)$, which is the probability that mark z is selected, *if* an event has occurred at time t. Then, we can compute the likelihood of a history of events $\mathcal{A}_T \subseteq \mathcal{H}_T$ as:

$$\mathbb{P}(\mathcal{A}_T) := \left(\prod_{e_i \in \mathcal{A}_T} \overbrace{\lambda^*(t_i)}^{\text{Prob. of an action at } t_i} \underbrace{m^*(z_i)}_{\text{Prob. of mark } z_i} \right) \overbrace{\exp\left(- \int_0^T \lambda^*(s)\, ds \right)}^{\text{Prob. of no actions at } t \in [0,T] \setminus \{t_i\}} . \tag{4.2}$$

In the remainder of the paper, whenever an intensity function and mark distribution are parametrized by θ, we write $\lambda^*_\theta(\cdot)$, $m^*_\theta(\cdot)$, $\mathbb{P}_\theta(\mathcal{A}_T)$, and, for notational simplicity, use $p^*_\theta = (\lambda^*_\theta, m^*_\theta)$ as a short-hand to denote the joint probability density of the MTPP. Recent literature [Du et al., 2016, Farajtabar et al., 2017, Karimi et al., 2016, Kim et al., 2018, Mei and Eisner, 2017, Wang et al., 2017, Zarezade et al., 2018] has established that MTPPs outperform other models (*e.g.*, exponential law) in their ability to accurately predict online and off-line human actions.

Reinforcement learning of marked temporal point processes. Assume there is an agent who takes actions in a complex environment and the environment also provides feedback to the agent over time. Moreover, both the actions and the feedback are asynchronous stochastic events localized in time and thus we characterize them using marked temporal point processes (MTPPs), *i.e.*,

- *Action events*: $\mathcal{A} = \{e_i = (t_i, y_i)\}$, where $(t_i, y_i) \sim p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$
- *Feedback events*: $\mathcal{F} = \{f_i = (t_i, z_i)\}$, where $(t_i, z_i) \sim p^*_{\mathcal{F};\phi} = (\lambda^*_\phi, m^*_\phi)$

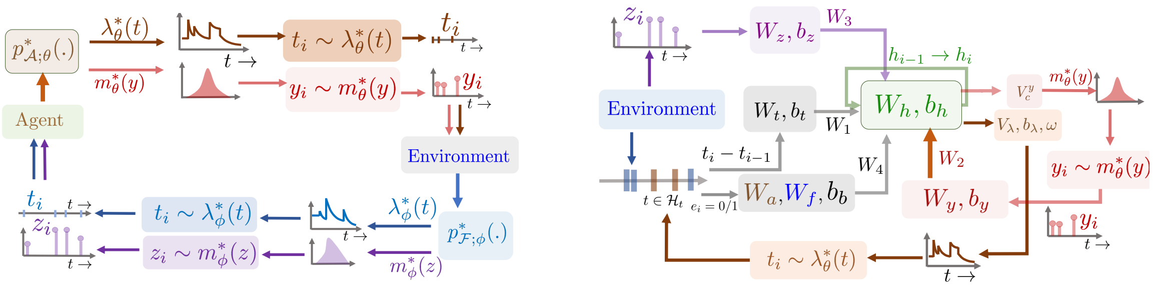
In the above characterization, we allow the joint probability densities $p^*_{\mathcal{A};\theta}$ and $p^*_{\mathcal{F};\phi}$ to depend on the joint history of events $\mathcal{H}_t := \mathcal{A}_t \cup \mathcal{F}_t$. Finally, after a *cut-off* time T, we assume that the agent receives an arbitrary (stochastic) reward $R^*(T)$, which may depend on the agent's actions $\mathcal{A}_T$ and the environment's feedback $\mathcal{F}_T$.

Given the above problem setting, we can formally define our reinforcement learning (RL) problem for marked temporal point processes as follows:

Problem definition. *Given an agent with $p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$, an environment with $p^*_{\mathcal{F};\phi} = (\lambda^*_\phi, m^*_\phi)$ and an arbitrary stochastic reward $R^*(T)$, the goal is to find the optimal action intensity and mark distribution—the optimal policy—that maximize the expected reward. Formally,*

$$\mathit{maximize}_{p^*_{\mathcal{A};\theta}(\cdot)} \quad \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) \right], \tag{4.3}$$

where the expectation is taken over all possible realizations of the marked temporal point processes associated to the agent's action events and the environment's feedback events. In the remainder of the paper, we will denote the optimal policy using $\pi^*(\theta) = \operatorname{argmax}_{p^*_{\mathcal{A};\theta}(\cdot)} \mathbb{E}\left[R^*(T)\right]$.



(a) Data and representation

(b) Policy parametrization

Figure 4.2: Reinforcement learning (RL) of of marked temporal point processes (MTPPs). Panel (a) shows the type of data and representation used in RL of MTPPs. Panel (b) shows the policy parametrization used by our policy gradient method.

Note that the above definition departs from previous work on reinforcement learning [Doya, 2000, Duan et al., 2016, Frémaux et al., 2013, Lillicrap et al., 2015, Mnih et al., 2016, Sutton and Barto, 1998, Vasilaki et al., 2009, Wierstra et al., 2007] in several ways. First, the agent's actions and environment's feedback are asynchronous stochastic events in continuous time. Moreover, note that the agent may receive feedback from the environment asynchronously at any time, not only after each of its actions. This is in contrast with previous work in the literature, which has only considered synchronous actions (and potentially delayed) feedback in discrete time (or, in some cases, continuous actions and feedback), as illustrated in Figure 4.1. Second, our policy is defined by a conditional intensity function (and a mark distribution), which is used to *sample* the times (and marks) of the agent's actions. Here, note that a sampled agent's action may need to be resampled due to the occurrence of new feedback events before the sampled time. In contrast, previous work has used probability distributions (or, in some cases, deterministic functions) as policies.

Remarkably, the above problem definition naturally fits numerous problems in a wide variety of application domains, particularly in the context of social and information online systems. For example, in personalized teaching in online learning platforms, the platform that shows content items to learners is the agent, the platform takes an action when it shows an item to a learner, the learners are the environment, and the probability that the learner recalls an item defines the reward. In viral marketing in social networks, a user who aims to increase the visibility of her posts is the agent, the user takes an action when she posts a message, her followers' feeds form the environment and the visibility (or attention) she receives defines the reward. In all these cases, the environment distribution $p^*_{\mathcal{F};\phi}$ may be highly complex and thus our policy gradient method will only assume that it can sample from $p^*_{\mathcal{F};\phi}$. In other words, the environment distribution will be considered a *black box*.

4.1.2 Control

In this section, we tackle the reinforcement learning problem defined by Eq. 4.3 using a novel policy gradient method for marked temporal point processes. More specifically, we first leverage recurrent neural networks (RNNs) to parametrize the policy $p^*_{\mathcal{A};\theta}$ and then use stochastic gradient descent (SGD) to find the policy parameters θ that maximizes the expected reward $\mathbb{E}[R^*]$.

Policy parametrization. In many application domains, at any time t, the (optimal) policy $p^*_{\mathcal{A};\theta}$ that maximizes the reward may depend on the previous history of the action events and the feedback events, $\mathcal{H}_t = \mathcal{A}_t \cup \mathcal{F}_t$, in an unknown and complex way. To capture such dependence, we parametrize the policy $p^*_{\mathcal{A};\theta}$ using a recurrent neural network (RNN), where we embed both the actions events and the feedback events into real-valued vectors $\boldsymbol{h}$, similarly as in several recent state of the art MTPP deep learning models [Du et al., 2016, Jing and Smola, 2017, Mei and Eisner, 2017][2]. Next, we elaborate further on our architecture[3], which we also summarize in Figure 4.2, and then discuss how to efficiently sample action events from the (optimal) policy.

— *Input layer.* After the i-th event occurs, be it an action event or a feedback event, the input layer converts the associated information, *i.e.*, the time t_i, the marker z_i (or y_i), and the type of event $e_i \in \{0, 1\}$, where $e_i = 0$ denotes action and $e_i = 1$ denotes feedback, into compact vectors. Specifically, it computes:

$$\begin{aligned} \boldsymbol{\tau}_i &= \boldsymbol{W}_t(t_i - t_{i-1}) + \boldsymbol{b}_t, & \boldsymbol{y}_i &= \boldsymbol{W}_y y_i + \boldsymbol{b}_y \ \text{ if } e_i = 0 \\ \mathbf{b}_i &= \boldsymbol{W}_a(1 - e_i) + \boldsymbol{W}_f e_i + \boldsymbol{b}_b, & \boldsymbol{z}_i &= \boldsymbol{W}_z z_i + \boldsymbol{b}_z \ \text{ if } e_i = 1 \end{aligned}$$

where $\boldsymbol{W}_\bullet$, $\boldsymbol{b}_t$, $\boldsymbol{b}_y$, $\boldsymbol{b}_z$ and $\boldsymbol{b}_b$ are trainable weights. Moreover, note that we encode the action marks y_i and feedback marks z_i separately since they may belong to different domains. To this aim, one of the inputs y_i and z_i will be marked as *absent* using sentinel values depending on whether $e_i = 0$ or $e_i = 1$, respectively. Finally, these signals are fed into the hidden layer, which we describe next.

— *Hidden layer.* This layer iteratively updates the latent embedding $\boldsymbol{h}_{i-1}$, by taking inputs of previous events from the input layer:

$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}_h \boldsymbol{h}_{i-1} + \boldsymbol{W}_1 \boldsymbol{\tau}_i + \boldsymbol{W}_2 \boldsymbol{y}_i + \boldsymbol{W}_3 \boldsymbol{z}_i + \boldsymbol{W}_4 \mathbf{b}_i + \boldsymbol{b}_h), \tag{4.4}$$

where $\boldsymbol{W}_\bullet$ and $\boldsymbol{b}_h$ are trainable weights.

— *Output layer.* The output layer computes the policy $p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$, *i.e.*, the intensity function λ^*_θ and the mark distribution m^*_θ. Assume the agent has generated i events by time t, then, the output layer computes the intensity as:

$$\lambda^*_\theta(t) = \exp\left(b_\lambda + w_t(t - t_i) + \boldsymbol{V}_\lambda \boldsymbol{h}_i\right) \tag{4.5}$$

where $\boldsymbol{V}_\lambda$, b_λ and w_t are trainable weights and t_i denotes the time of the i-th action event. Here, the b_λ encodes a base intensity level for the occurrence of the $(i+1)$-th action event, the term $w_t(t - t_i)$ encodes the influence of the i-th action event, and the term $\boldsymbol{V}_\lambda$ encodes the influence of previous events. The particular choice of mark distribution m^*_θ depends on the application domain. Here, we experiment with discrete marks and thus model the marks using a multinomial distribution, *i.e.*,

$$\mathbb{P}[y_{i+1} = c] = \frac{\exp(\boldsymbol{V}^y_{c,:} \boldsymbol{h}_i)}{\sum_{l \in \mathcal{Y}} \exp(\boldsymbol{V}^y_{l,:} \boldsymbol{h}_i)}, \tag{4.6}$$

where $\mathcal{Y}$ denote the domain of the marks and $\boldsymbol{V}^y$ are trainable weights.

Sampling action events from the policy. To implement the above policy $p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$, we need to be able to sample the action times t and marks y from the intensity function defined by Eq. 4.5 and the mark distribution defined by Eq. 4.6, respectively. While the latter reduces to sampling from a multinomial distribution, which is straightforward, the former requires developing a novel sampling algorithm leveraging inverse transform sampling, which we describe in Algorithm 3. The details of calculating $CDF(\bullet)$ and the related modifications are provided in Appendix A.3.

Maximizing the expected reward. In the following, we denote the expected reward as a function of the policy parameters θ as:

$$J(\theta) = \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T)\right] \tag{4.7}$$

---

[2] Note that previous MTPP deep learning models aims to provide event predictions. This is contrast with the current work, which aims to provide optimal event interventions.

[3] Depending on the application domains, action events or feedback events may not contain marks and, thus, the architecture may be slightly simpler.

Algorithm 3 Returns the next action time

```
Input:  Parameters $b_\lambda, w_t, \boldsymbol{V}_\lambda, \boldsymbol{h}_i$, last event time $t'$
Output:  Next action time $t$
$CDF(\bullet) \leftarrow$ Cumulative distribution of next arrival time
$u \leftarrow$ Unif$[0, 1]$
$t \leftarrow CDF^{-1}(u)$
while $t < T$ do
    $(s, z) \leftarrow$ WaitUntilNextFeedback$(t)$
    if feedback arrived before $t$ then
        $CDF(\bullet) \leftarrow$ Modify$(CDF(\bullet), s, z)$
        $t \leftarrow CDF^{-1}(u)$
    else
        return  t
    end if
end while
return  t
```

Then, we find the optimal policy $p^*_{\mathcal{A};\theta}$ that maximizes the expected reward function $J(\theta)$ using stochastic gradient descent (SGD) [Rumelhart et al., 1986], *i.e.*, $\theta_{l+1} = \theta_l + \alpha_l \nabla_\theta J(\theta)|_{\theta=\theta_l}$. To do so, we need to compute the gradient of the expected reward function $\nabla_\theta J(\theta)$, however, this may seem challenging at first especially since the expectation is taken over realizations of marked temporal point processes. Perhaps surprisingly, we can compute such gradient using the following proposition (proved in Appendix A.1).

Proposition 4. *Given an agent with* $p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$*, an environment with* $p^*_{\mathcal{F};\phi} = (\lambda^*_\phi, m^*_\phi)$*, the gradient of the expected reward function* $J(\theta)$ *with respect to* θ *is given by:*

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \right], \tag{4.8}$$

where $\log \mathbb{P}_\theta(\mathcal{A}_T) = \sum_{e_i \in \mathcal{A}_T} (\log \lambda^*_\theta(t_i) + \log m^*_\theta(z_i)) - \int_0^T \lambda^*_\theta(s)\, ds$.

In the above proposition, the gradient of the log-likelihood of the times and marks of a realization of the marked temporal point process associated to the agent's actions, $\nabla_\theta \log \mathbb{P}^*_{\mathcal{A};\theta}(\mathcal{H}_T)$, can be easily computed using the policy parametrization defined by Eqs. 4.5 and 4.6. Moreover, note that the proposition formally shows that the REINFORCE trick [Williams, 1992] is still valid if the expectation is taken over realizations of marked temporal point processes, which are a type of *random elements* [Daley and Vere-Jones, 2007] whose values are discrete events localized in continuous time.

Unfortunately, the above procedure does not limit the intensity of actions by the agent and this may be problematic in practice (*e.g.*, in viral marketing in social networks, a user who aims to increase the visibility of her posts may only be able to post a certain number of times). To overcome this, we consider instead a penalized expected reward function $J_r(\theta)$ with differentiable regularizers $g_\lambda(\lambda^*_\theta(t))$ and $g_m(m^*_\theta(t))$, which implicitly impose a budget on the number of action events and marks, respectively, *i.e.*,

$$J_r(\theta) = \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) - q_l \int_0^T g_\lambda(\lambda^*_\theta(t)) - q_m \int_0^T g_m(m^*_\theta(t)) dt \right]. \tag{4.9}$$

The gradient of the penalized reward can be readily computed using the following proposition (proved in Appendix A.2):

Proposition 5. *Given an agent with* $p^*_{\mathcal{A};\theta} = (\lambda^*_\theta, m^*_\theta)$*, an environment with* $p^*_{\mathcal{F};\phi} = (\lambda^*_\phi, m^*_\phi)$*, the gradient* *of $J_r(\theta)$ is given by,*

$$\nabla_\theta J_r(\theta) = \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \Bigg[\left(R^*(T) - q_l \int_0^T g_\lambda(\lambda^*_\theta(t)) - q_m \int_0^T g_m(m^*_\theta(t))\, dt \right) \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) - \left(q_l \int_0^T g'_\lambda(\lambda^*_\theta(t)) \nabla_\theta \lambda^*_\theta(t)\, dt + q_m \int_0^T g'_m(m^*_\theta(t)) \nabla_\theta m^*_\theta(t)\, dt \right) \Bigg], \quad (4.10)$$

where $g'_\lambda(\lambda^*_\theta(t)) = \frac{d\, g_\lambda(\lambda^*_\theta(t))}{d\, \lambda^*_\theta(t)}$ *and* $g'_m(m^*_\theta(t)) = \frac{d\, g_m(m^*_\theta(t))}{d\, m^*_\theta(t)}$.

In our experiments, we will approximate the expectation in Eq. (4.10) by first running a *batch* of realizations (or *episodes*) of the corresponding marked temporal point processes[4] and then calculating the mean of the resulting gradients for each batch.

4.1.3 Experiments on smart broadcasting

Problem definition. In the smart broadcasting problem, first introduced by Spasojevic et al. [2015], the goal is to help a social media user decide when to post to achieve high visibility in her followers' feeds, *i.e.*, to elicit attention from her followers. Under our problem definition, the user is the agent, she generates action events $\mathcal{A}$ when she posts, her followers' feeds forms the environment, the environment generates feedback events $\mathcal{F}$ when any of the other users her followers follow post, and the visibility she receives defines the reward. Then, the problem reduces to finding the (optimal) policy $p^*_{\mathcal{A};\theta}$ that maximizes the reward.

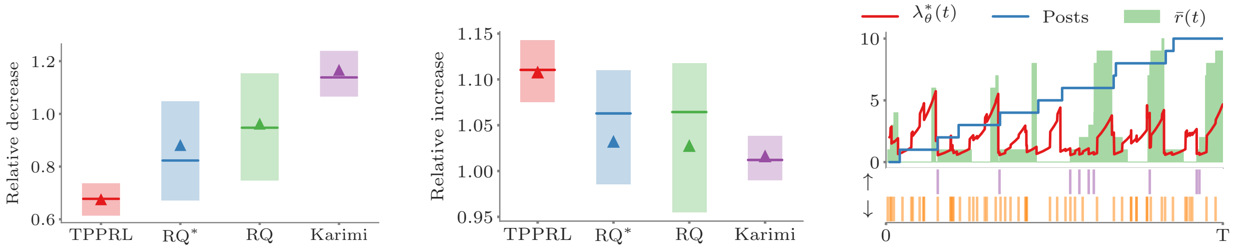
Following previous work [Wang et al., 2018, Zarezade et al., 2018, 2017a], we measure visibility a user achieves, *i.e.*, the reward, using two different metrics: (i) the position of her most recent post on her followers' feeds over time, or *rank*, *i.e.*, $R^*(T) = \int_0^T r(t)dt$, where the position zero, $r(t) = 0$, corresponds to top and thus lower is better; (ii) the (amount of) time that her most recent post is at the top of her followers' feeds, or *time at the top*, *i.e.*, $R^*(T) = \int_0^T \mathbb{I}(r(t) < 1)dt$, and thus higher is better. If the followers' feeds are sorted in reverse chronological order, previous work has derived optimal offline [Karimi et al., 2016] and online [Zarezade et al., 2017a] algorithms for (i) and (ii), respectively, under the additional assumption that the posting intensity of other users her followers follow adopts certain functional form. However, as pointed out by previous work, feeds are typically algorithmically sorted, the posting intensity of other users may be highly complex, and thus the derived algorithms may be of limited use in practice. Here, we use our reinforcement learning method to derive (optimal) policies for algorithmically sorted feeds and, by doing so, we are able to help users achieve higher visibility than the above algorithms. Appendix A.7 contains additional experiments for feeds sorted in reverse chronological order.

Experimental setup. We use data gathered from Twitter as reported in previous work [Cha et al., 2010], which comprises profiles of 52 million users, 1.9 billion directed follow links among these users, and 1.7 billion public tweets posted by the collected users. The follow link information is based on a snapshot taken at the time of data collection, in September 2009. Here, we focus on the tweets published during a two month period, from July 1, 2009 to September 1, 2009, and sample 1000 users uniformly at random. For each of these users, we retrieve five of her followers (chosen at random), select five other *followees* of each follower (chosen at random), and collect all the (re)tweets they published. Each follower represents a wall and our broadcaster is *competing* with the other followees of follower for attention. Since we do not have access to the feed sorting algorithm used by Twitter, we experiment with a relatively simple sorting algorithm based on a priority queue[5] (refer to Appendix A.6). Here, since our feed sorting algorithm does only depends on the time of the post and the identity of the user who posts, not marks (*e.g.*, content of the post), the optimal

---

[4] In some applications, we may be able to play back historical data from the environment against our policy and, in other domains, we may need to resort to a (complex) environment simulator.

[5] We expect that, the more complex the sorting algorithm, the larger the competitive advantage our algorithm will offer in comparison with competing methods designed for feeds sorted in reverse chronological order.



(a) Average rank (b) Time at top (c) Example

Figure 4.3: Smart broadcasting. Performance of our policy gradient method against RedQueen [Zarezade et al., 2017a] (RQ), a variant of RedQueen which has access to true ranks (RQ*), and Karimi's method [Karimi et al., 2016] on feeds using a sorting algorithm based on a priority queue (refer to Appendix A.6). Panels (a) and (b) show the average rank and time at the top, where the solid horizontal line shows the median value across users, normalized with respect to the value achieved by a user who follows a uniform Poisson intensity, and the box limits correspond to the 25%-75% percentiles. For the average rank, lower is better and, for time at the top, higher is better. In both cases, the number of messages posted by each method is the same. Panel (c) shows a user's intensity $\lambda^*_\theta(\cdot)$ (in blue), as provided by our method, the counts of the user's posts (in green), the average rank (in red), the posting times of a competing user with higher priority (in purple), and the posting times of another competing user with lower priority (in yellow).

policy only comprises an intensity function, *i.e.*, $p^*_{\mathcal{A};\theta} = \lambda^*_\theta(t)$. Then, we train and test our policy gradient method as follows.

For each user, we divide her feedback events, *i.e.*, the posts by other users her followers follow, into a training set and a test set. The latter contains all feedback events generated in a time window of length T at the end of the recording period and the former contains all other feedback events. Here, we set the length T such that the overall expected number of events in the test set is ∼200. Then, we train each user's policy $\lambda^*_\theta(t)$ by using stochastic gradient descent (SGD) with a quadratic regularizer $g(\lambda^*(t)) = (\lambda^*(t))^2$. More specifically, on each iteration i, we build a batch of b sequences of length T, taken uniformly at random from the training set, we *replay* the feedback events from these sequences while interleaving the posts generated by our policy $\lambda^*_{\theta_i}$, and compute the reward at the end of each sequence. To test the trained policy $\lambda^*_\theta(t)$, we just replay the feedback events from the test set while interleaving the posts generated by the policy and compute the reward at the end of the sequence. Appendix A.4 contain additional implementation details.

In the above, we experiment both with rank and time at the top as rewards and compare our method with two state of the art methods, RedQueen [Zarezade et al., 2017a] and the method by Karimi et al. [2016]. The former is an online algorithm specially designed to minimize the average rank in feeds sorted in reverse chronological order and the latter is an offline algorithm specially designed to maximize the time at the top in feeds sorted in reverse chronological order. However, because RedQueen assumes that the feed is inverse chronologically sorted and posts with intensity $\propto \text{rank}_{\text{chrono}}(t)$, we also compare our method TPPRL against a stronger heuristic RQ*, which posts with intensity $\propto \text{rank}_{\text{priority}}(t)$.

Results. Figures 4.3(a-b) summarize the results, where the number of messages posted by each method is the same and all rewards are normalized by the reward achieved by a baseline user who follows a uniform Poisson intensity. The results show that, by not making any assumption about the feed sorting algorithm, our method is able to outperform both RedQueen and Karimi's method, which were specially designed to minimize the average rank and time at the top in feeds sorted in reverse chronological order, respectively. Moreover, our method provides solutions with smaller variance in performance than RedQueen. Finally, in Figure 4.3(c), we give some intuition on the type of policy our method learns using a toy example, where a user competes for attention with two other users in a follower's feed, one with higher priority and another with lower priority. Our method learns to avoid posting whenever the user with higher priority posts.

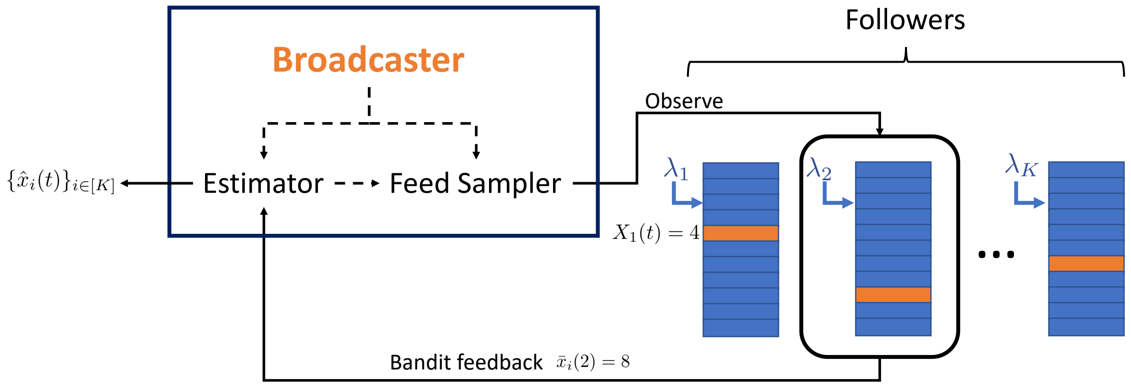


Figure 4.4: The problem setup: The broadcaster's aim is to calculate her rank on the feeds of her followers at each time point t. However, the broadcaster can only observe 1 follower's feed at each time step and has to estimate her rank on the other feeds. In the meanwhile, all feeds are receiving posts from other broadcasters at the rate $\{\lambda_i\}_{i\in[K]}$. Our broadcaster will have two modules: (i) an estimator, which produces the estimates $\{\hat{x}_i(t)\}_{i\in[K]}$ for each time point t, and (ii) a feed sampler, which picks the next feed to *observe* at time t. The estimator's internal state (*i.e.*, estimates of the rate for different feeds $\{\hat{\lambda}_i(t)\}_{i\in[K]}$) may be used by the sampler to pick the next feed.

4.1.4 Summary

In this thesis, TPPRL is be applied to the *when-to-post* problem [Karimi et al., 2016, Spasojevic et al., 2015, Wang et al., 2018, Zarezade et al., 2018, 2017a]. For *simple* dynamics and objective functions, which allow for stochastic optimal control approaches, TPPRL achieves a comparable performance even though it does not have access to the true underlying dynamics. For *complex* dynamics and/or objective functions, which do not allow for stochastic optimal control approaches, TPPRL is able to successfully find interventions that optimize the corresponding objective function and beat several competitive baselines (see Figure 4.3). To facilitate research in temporal point processes within the reinforcement learning community at large, an open-source implementation of TPPRL written in TensorFlow as well as synthetic and real-world data used in the experiments has been released.[6]

4.2 Learning to schedule

In today's fast paced world, if a Twitter or Facebook post is missed, it may as well have never existed, irrespective of how good the content was. As each broadcaster is competing with all others on the social network for attention of the followers, it leads to a real time competition for the visibility of each post. The analytics associated with keeping track of the performance of one's post (*e.g.*, likes, reshares, retweets, *etc.*) are contingent on the post being visible to the followers and being liked by the followers, which is a problem of independent interest [Berger and Milkman, 2012]. Hence, likes and reshares do not give a clear feedback to the broadcaster about whether the followers can see the posts at all or not. However, the platforms like Facebook and Twitter usually rate-limit the API which the broadcaster can use to fetch the feeds of the followers to determining the visibility of one's posts. Hence, it is of paramount importance to carefully choose which follower's feed to observe before to get a accurate an estimate as possible.

The problem of determining the visibility of one's posts on the feeds of the followers can be formulated as an online sequential decision making problem with bandit feedback, as shown in Figure 4.4, where the

---

[6] https://github.com/Networks-Learning/tpprl

broadcaster attempts to learn the rate of change of each feed while making API calls to the platform to learn the performance of her own posts. This formulation instantly suggests that parallels can be drawn between this problem and the problem of web-indexing studied by Cho and Garcia-Molina [2003a,b]. Recently, Azar et al. [2018] have uncovered tractable algorithms for the web-crawling setting when the rate of updates of each source (*i.e.*, web-page or feed) is known. The problem is under active investigation and my aim is to come up with algorithms to learn the visibility of one's post which have theoretical guarantees on their accuracy/performance.

4.2.1 Problem Formulation

In this section, we consider the problem of keeping a cache of m webpages up-to-date by modelling the changes to those webpages, the requests for the pages, and the bandwidth constraints placed on a standard web-crawler. We assume that the cache is empty when all processes start at time 0.

We model the changes to each webpage as Poisson processes with constant rates. The parameters of these *change processes* are denoted by $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_m]$, where $\xi_i > 0$ denotes the rate of changes made to webpage i. We will assume that $\boldsymbol{\xi}$ are not known to us but we know only an upper bound $\xi_{\max}$ and lower bound $\xi_{\min}$ on the change rates. The crawler will learn $\boldsymbol{\xi}$ by refreshing the pages and observing the single-bit feedback described below. We denote the time webpage i changes for the nth time as $x_{i,n}$. We model the incoming requests for each webpage also as Poisson processes with constant rates and denote these rates as $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \ldots, \zeta_m]$. We will assume that these rates, which can also be interpreted as the *importance* of each webpage in our cache, are known to the crawler. We will denote the time webpage i is requested for the nth time as $z_{i,n}$. The change process and the request process, given their parameters, are assumed to be independent of each other.

We denote time points when page i is refreshed by the crawler using $(y_{i,n})_{n=1}^{\infty}$. The feedback which the crawler gets after refreshing a webpage i at time $y_{i,n}$ consists of a single bit which indicates whether the webpage has changed or not since the last observation, if any, that was made at time $y_{i,n-1}$. Let $E_i^{\emptyset}[t_0, t]$ indicate the event that neither a change nor a refresh of the page has happened between time t_0 and t for webpage i. Define $\textsc{Fresh}(i, t)$ as the event that the webpage is *fresh* in the cache at time t. Defining the maximum of an empty set to be $-\infty$, we have:

$$\textsc{Fresh}(i,t) = \begin{cases} 0 & \text{if } E_i^{\emptyset}[0,t] \\ 1_{(\max\{x_{i,j}: x_{i,j} < t\} < \max\{y_{i,j}: y_{i,j} < t\})} & \text{if } \neg E_i^{\emptyset}[0,t] \end{cases}$$

where the indicator function $1_{(\bullet)}$ takes value 1 on the event in its argument and value 0 on its complement. Hence, we can describe the feedback we receive upon refreshing a page i at time $y_{i,n}$ as:

$$o_{i,n} = \textsc{Fresh}(i, y_{i,n}). \tag{4.11}$$

We call this a *partial* observation of the change process to contrast it with *full* observability of the process, *i.e.*, when a refresh at $y_{i,n}$ provides the number of changes to the webpage in the period $(y_{i,n}, y_{i,n-1})$. For example, the crawler will have full observability of the incoming request processes.

The policy space Π consists of all measurable functions which, at any time t, decide when the crawler should refresh which page in its cache based on the observations up to time t that includes $\{(o_{i,n})_{n=1}^{N} \mid N = \operatorname{argmax}_n y_{i,n} < t; i \in [m]\}$.

The objective of the web-crawling problem is to refresh webpages such that it maximizes the number of requests which are served a fresh version. So the utility of a policy $\pi \in \Pi$ followed from time t_1 to t_2 can be written as:

$$U([t_1, t_2], \pi; \boldsymbol{\xi}) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t_1 \le z_{i,n} \le t_2} \textsc{Fresh}(i, z_{i,n}). \tag{4.12}$$

Our goal is to find a policy that maximizes this utility (4.12).[7] However, if the class of policies is unconstrained, the utility can be maximized by a trivial policy which continuously refreshes all webpages in the cache. This is not a practical policy since it will overburden the web servers and the crawler. Therefore, we would like to impose a bandwidth constraint on the crawler. Such a constraint can take various forms and a natural way of framing it is that the expected number of webpages that are refreshed in *any* time interval with width w cannot exceed $w \times R$. This constraint defines a class of stochastic policies $\Delta_R = \{(\rho_1, \ldots, \rho_m) \in (\mathbb{R}^+)^m : \sum_{i=1}^m \rho_i = R\} \subset \Pi$, where each webpage's refresh time is drawn by the crawler from a Poisson process with rate ρ_i.

This problem setup was studied by Azar et al. [2018] and shown to be tractable. Recently, Kolobov et al. [2019a] have studied a different objective function which penalises the *harmonic staleness* of pages and is similarly tractable. Another example of such an objective (albeit with full observability) is proposed by Sia et al. [2007]. We show later that our analysis extends to their objective as well.

We define the regret of policy $\pi \in \Delta_R$ as follows

$$R(T, \pi; \boldsymbol{\xi}) = \max_{\pi' \in \Delta_R} \mathbb{E}\left[U([0,T], \pi'; \boldsymbol{\xi})\right] - \mathbb{E}\left[U([0,T], \pi; \boldsymbol{\xi})\right].$$

It is worth reiterating that the parameters $\boldsymbol{\xi}$ will not be known to the crawler. The crawler will need to determine when and which page to refresh given only the single bits of information $o_{i,n}$ corresponding to each refresh the policy makes.

4.2.2 Learning with Poisson Processes and Partial Observability

In this section, we will derive an analytical form of the utility function which is amenable to analysis, describe how to uncover the optimal policy in Δ_R if all parameters (*i.e.*, $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$) are known, and consider the problem of learning the parameters $\boldsymbol{\xi}$ with partial observability. We will use the insight gained to determine some properties a learning algorithm should have so that it can be analysed tractably.

Utility and the Optimal policy. Consider the expected value of the utility of a policy $\boldsymbol{\rho} \in \Delta_R$ which the crawler follows from time t_0 till time T. Assume that the cache at time t_0 is given by $\boldsymbol{S}(t_0) = [s_1, s_2, \ldots, s_m] \in \{0,1\}^m$, where $s_i = \textsc{Fresh}(i, t_0)$. Then, using (4.12), we have:

$$\begin{aligned}
&\mathbb{E}[U([t_0,T], \boldsymbol{\rho}; \boldsymbol{\xi}) \mid \boldsymbol{S}(t_0)] \\
&= \frac{1}{m} \sum_{i=1}^{m} \mathbb{E}\left[\sum_{t_0 < z_{i,n} < T} \textsc{Fresh}(i, z_{i,n}) \,\middle|\, \textsc{Fresh}(i, t_0) = s_i \right] \\
&= \frac{1}{m} \sum_{i=1}^{m} \int_{t_0}^{T} \underbrace{\zeta_i \mathbb{P}_{\boldsymbol{\rho}}\left(\textsc{Fresh}(i,t) = 1 \mid \textsc{Fresh}(i, t_0) = s_i\right)}_{F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi})} \, dt \qquad (4.13)
\end{aligned}$$

where (4.13) follows from Campbell's formula for Poisson Process [Kingman, 1993] (expectation of a sum over the point process equals the integral over time with process' intensity measure) as well as the fact that the request process and change/refresh processes are independent.[8] In the next lemma, we show that the differential utility function $F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi})$, defined implicitly in (4.13), can be made time-independent if the policy is allowed to run for *long-enough*.

Lemma 1 (Adapted from [Azar et al., 2018]). *For any given $\varepsilon > 0$, let $\boldsymbol{\rho} \in \Delta_R$ be a policy which the crawler adopts at time t_0 and let the initial state of the cache be $\boldsymbol{S}(t_0) = [s_1, s_2, \ldots, s_m] \in \{0,1\}^m$, where $s_i = \textsc{Fresh}(i, t_0)$. Then if $t - t_0 \geq \frac{1}{\xi_{\min}} \log\left(\frac{2\sum_{j=1}^m \zeta_i}{\varepsilon}\right)$, then $\sum_{i=1}^m \frac{\zeta_i \xi_i}{\xi_i + \rho_i} < \sum_{i=1}^m F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi}) < \sum_{i=1}^m \frac{\zeta_i \xi_i}{\xi_i + \rho_i} + \varepsilon$.*

---

[7] The freshness of the webpages does depend on the policy π which is hidden by function $\textsc{Fresh}$.

[8] Note that the *rates* of the processes can be correlated; only the events need to be drawn independently.

Hence, as long as condition described by Lemma 1 holds, the differential utility function for a policy $\boldsymbol{\rho} \in \Delta_R$ is time independent and can be written as just $F(\boldsymbol{\rho}; \boldsymbol{\xi}) = \sum_{i=1}^{m} \frac{\zeta_i \xi_i}{\xi_i + \rho_i}$. Substituting this into (4.13), we get:

$$\mathbb{E}[U([t_0, T], \boldsymbol{\rho}; \boldsymbol{\xi})] \approx \frac{1}{m} \sum_{i=1}^{m} \int_{t_0}^{T} \frac{\rho_i}{\rho_i + \xi_i} \zeta_i \, dt = \frac{T - t_0}{m} \sum_{i=1}^{m} \frac{\rho_i \zeta_i}{\rho_i + \xi_i} \tag{4.14}$$

This leads to the following time-horizon independent optimisation problem for the optimal policy:

$$\underset{\boldsymbol{\rho} \in \Delta_R}{\text{maximize}} \; F(\boldsymbol{\rho}; \boldsymbol{\xi}) = \sum_{i=1}^{m} \frac{\rho_i \zeta_i}{\rho_i + \xi_i} \tag{4.15}$$

Azar et al. [2018] have considered the approximate utility function given by (4.15) to derive the optimal refresh rates $\boldsymbol{\rho}^{\star}$ for known $\boldsymbol{\xi}$ in $\mathcal{O}(m \log m)$ time (See Algorithm 2 in [Azar et al., 2018]).[9]

This approximation has bearing upon the kind of learning algorithms we could use while keeping the analysis of the algorithm and computation of the optimal policy tractable. The learning algorithm we employ must follow a policy $\boldsymbol{\rho} \in \Delta_R$ for a certain amount of *burn-in* time before we can use (4.14) to approximate the performance of the policy. If the learning algorithm changes the policy *too quickly*, then we may see large deviations between the actual utility and the approximation. However, if the learning algorithm changes the policy *slowly*, where Lemma 1 can serve as a guide to the appropriate period, then we can use (4.14) to easily calculate its performance between t_0 and T. Similar conditions can be formulated for the approximate objectives proposed by Kolobov et al. [2019a] and Sia et al. [2007] as well.

Now that we know how to uncover the optimal policy when $\boldsymbol{\xi}$ are known, we turn our attention to the task of learning it with partial observations.

Learnability with Partial Observations. In this section, we address the problem of partial information of Poisson process and investigate under what condition the rate of the Poisson process can be estimated. In our setting, for an arbitrary webpage, we only observe binary outcomes $(o_n)_{n=1}^{\infty}$, defined by (4.11). The refresh times $(y_n)_{n=1}^{\infty}$ and the Poisson process of changes with rate ξ induce a distribution over $\{0,1\}^{\mathbb{N}}$ which is denoted by μ_{ξ}. If the observations happen at regular time intervals, i.e. $y_n - y_{n-1} = c$ for some constant c, then the support $\mathcal{S}_{\xi}$ of μ_{ξ} is:

$$\mathcal{S}_{\xi} = \left\{ (o_n)_{n=1}^{\infty} \in \{0,1\}^{\mathbb{N}} : \lim_{n \to \infty} \frac{\sum_{j=1}^{n} o_j}{n} = 1 - e^{-c\,\xi} \right\}.$$

This means that we can have a consistent estimator, based on the strong law of large numbers, if the crawler refreshes the cache at fixed intervals.

However, we can characterise the necessary property of the set of partial observations which allows parameter estimation of Poisson processes under the general class of policies Π. This result may be of independent interest.

Lemma 2. *Let* $\{y_0 := 0\} \cup (y_n)_{n=1}^{\infty}$ *be a sequence of times, such that* $\forall n. \, y_n > 0$*, at which observations* $(o_n)_{n=1}^{\infty} \in \{0,1\}^{\mathbb{N}}$ *are made of a Poisson process with rate* ξ*, such that* $o_n := 1$ *iff there was an event of the process in* $(y_{n-1}, y_n]$*, define* $w_n = y_n - y_{n-1}$*,* $I = \{n : w_n < 1\}$ *and* $J = \{n : w_n \geq 1\}$*. Then:*

1. *If* $\sum_{n \in I} w_n < \infty$ *and* $\sum_{n \in J} e^{-\xi w_n} < \infty$*, then any statistic for estimating* ξ *has non-vanishing bias.*
2. *If* $\sum_{n \in I} w_n = \infty$*, then there exist disjoint subsets* $I_1, I_2, \ldots$ *of* I *such that* $\left(\sum_{n \in I_k} w_n\right)_{k=1}^{\infty}$ *is monotone and* $\sum_{n \in I_k} w_n \in (1,2)$ *for* $k = 1, 2, \ldots$ *For any such sequence* $\mathcal{I} = (I_k)_{k=1}^{\infty}$*, the mapping* $c_{\mathcal{I}}(\xi) = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \exp\left(-\xi \sum_{n \in I_k} w_n\right)$ *is strictly monotone and*

$$\left[\frac{1}{K} \sum_{k=1}^{K} \mathbb{I}\left(\sum_{n \in I_k} o_n \geq 1 \right) \right] \xrightarrow{a.s.} 1 - c_{\mathcal{I}}(\xi).$$

---

[9] The optimal policy can be obtained in $\mathcal{O}(m)$ time by using the method proposed by Duchi et al. [2008].

3. *If* $\sum_{n\in J} e^{-w_n\xi} = \infty$ *then, there exists a sequence* $\mathcal{J} = (J_k)_{k=1}^{\infty}$ *of disjoint subsets of* J *such that* $\left(\sum_{n\in J_k} e^{-w_n\xi}\right)_{k=1}^{\infty}$ *is monotone and* $\sum_{n\in J_k} e^{-w_n\xi} \in [1/e, 2/e)$ *for* $k = 1, 2, \ldots$ *For any such* $\mathcal{J}$*, the mapping* $c_{\mathcal{J}}(\xi) = \lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K}\prod_{n\in J_k}\left(1-e^{-\xi w_n}\right)\right]$ *is strictly monotone and*

$$\lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K}\mathbb{I}\left(o_n\geq 1,\ \forall n\in J_k\right)\right] \xrightarrow{a.s.} c_{\mathcal{J}}(\xi),$$

Note that it is possible that, for some ξ, the statistics almost surely converge to a value that is unique to ξ, but for some other one they do not. Indeed, when $w_n = \ln n$, then $\sum_{n\in I} w_n < \infty$ and $\sum_{n\in J} e^{-\xi w_n} < \infty$ for $\xi = 2$, but $\sum_{n\in J} e^{-\xi w_n} = \infty$ for $\xi = 1$. More concretely, assuming that the respective limits exist, we have:

$$\liminf_{n\in J}\frac{w_n}{\ln n} > \frac{1}{\xi} \implies \sum_n e^{-\xi w_n} < \infty \quad \text{and}$$
$$\limsup_{n\in J}\frac{w_n}{\ln n} \leq \frac{1}{\xi} \implies \sum_n e^{-\xi w_n} = \infty.$$

In particular, if $\limsup_{n\in J}\frac{w_n}{\ln n} = 0$, it implies that $\sum_{n\in J} e^{-\xi w_n} = \infty$ for all $\xi > 0$, which, in turn, implies that it will be possible to learn the true value for any parameter $\xi > 0$.

Lemma 2 has important implications on the learning algorithms we can use to learn $\boldsymbol{\xi}$. It suggests that if the learning algorithm decreases the refresh rate ρ_i for a webpage too quickly, such that $\mathbb{P}\left(\liminf_{n\to\infty}\frac{w_{i,n}}{\ln n} > \frac{1}{\xi_i}\right) > 0$ (assuming the limit exists), then the estimate of each parameter ξ_i has non-vanishing error.

In summary, in this section, we have made two important observations about the learning algorithm we can employ to solve the web-crawling problem. Firstly, given an error tolerance of $\varepsilon > 0$, the learning algorithm should change the policy only after $\sim \frac{1}{\xi_{\min}}\log\left(2\sum_i\zeta_i/\varepsilon\right)$ steps to allow for time invariant differential utility approximations to be valid. Secondly, in order to obtain consistent estimates for $\boldsymbol{\xi}$ from partial observations, the learning algorithm should not change the policy so drastically that it violates the conditions in Lemma 2. These observations strongly suggest that to obtain theoretical guarantees on the regret, one should use *phased* learning algorithms where each phase of the algorithm is of duration $\sim \frac{1}{\xi_{\min}}\log\left(2\sum_i\zeta_i/\varepsilon\right)$, the policy is only changed when moving from one phase to the other, and the changes made to the policy are such that constraints presented in Lemma 2 are not violated. Parallels can be drawn between such learning algorithms and the algorithms used for online learning of Markov Decision Processes which rely on bounds on mixing times [Neu et al., 2010]. In Section 4.2.4, we present the simplest of such algorithms, *i.e.*, the explore-and-commit algorithm, for the problem and provide theoretical guarantees on the regret. Additionally, we also empirically compare the performance of ETC to the phased ε-greedy learning algorithm.

In the next section, we investigate practical estimators for the parameters $\widehat{\boldsymbol{\xi}}$ and the formal guarantees they provide for the web-crawling problem.

4.2.3 Parameter Estimation and Sensitivity Analysis with Partial Observations

In this section, we address the problem of parameter estimation of Poisson processes under partial observability and investigate the relationship between the utility of the optimal policy $\boldsymbol{\rho}^\star$ (obtained using true parameters) and policy $\widehat{\boldsymbol{\rho}}$ (obtained using the estimates).

Assume the same setup as for Lemma 2, *i.e.*, we are given a finite sequence of observation times $\{y_0 := 0\} \cup (y_n)_{n=1}^{N}$ in advance, and we observe $(o_n)_{n=1}^{N}$, defined as in (4.11), based on a Poisson process with rate ξ. Define $w_n = y_n - y_{n-1}$. Then log-likelihood of $(o_n)_{n=1}^{N}$ is:

$$\mathcal{L}(\xi) = \sum_{n:o_n=1}\ln(1-e^{-\xi w_n}) - \sum_{n:o_n=0}\xi w_n \tag{4.16}$$

which is a concave function. Taking the derivative and solving for ξ yields the maximum likelihood estimator [Cho and Garcia-Molina, 2003b]. However, as the MLE estimator lacks a closed form, coming up with a non-asymptotic confidence interval is a very challenging task. Hence, we consider a simpler estimator.

Let us define an intermediate statistic $\widehat{p}$ as the fraction of times we observed that the underlying Poisson process produced no events, $\widehat{p} = \frac{1}{N}\sum_{n=1}^{N}(1 - o_n)$. Since $\mathbb{P}(o_n = 0) = e^{-\xi w_n}$ we get $\mathbb{E}[\widehat{p}] = \frac{1}{N}\sum_{n=1}^{N} e^{-\xi w_n}$. Motivated by this, we can estimate ξ by the following moment matching method: choose $\widetilde{\xi}$ to be the unique solution of the equation

$$\widehat{p} = \frac{1}{N}\sum_{n=1}^{N} e^{-\widetilde{\xi} w_n}, \tag{4.17}$$

and then obtain estimator $\widehat{\xi}$ of ξ by clipping $\widetilde{\xi}$ to range $[\xi_{\min}, \xi_{\max}]$, $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$. The RHS in (4.17) is monotonically decreasing in $\widetilde{\xi}$, therefore finding the solution of (4.17) with error γ can be done in $O(\log(1/\gamma))$ time based on binary search. Additionally, if the intervals are of fixed size, *i.e.*, $\forall n.\, w_n = c$, then $\widetilde{\xi}$ reduces to the maximum likelihood estimator. Such an estimator was proposed by Cho and Garcia-Molina [2003b] and was shown to have good empirical performance. Here, instead of smoothing the estimator, a subsequent clipping of $\widetilde{\xi}$ resolves the issue of its instability for the extreme values of $\widehat{p} = 0$ and $\widehat{p} = 1$ (when the solution to (4.17) becomes $\widetilde{\xi} = \infty$ and $\widetilde{\xi} = 0$, respectively). In the following lemma, we will show that this estimator $\widehat{\xi}$ is also amenable to non-asymptotic analysis by providing a high probability confidence interval for it.

Lemma 3. *Under the condition of Lemma 2, for any $\delta \in (0, 1)$, and N observations it holds that*

$$\mathbb{P}\left(|\widehat{\xi} - \xi| \geq \left(\frac{1}{N}\sum_{n=1}^{N} w_n e^{-\xi_{\max} w_n}\right)^{-1} \sqrt{\frac{\log\frac{2}{\delta}}{2N}}\right) \leq \delta$$

where $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$ and $\widetilde{\xi}$ is obtained by solving (4.17).

Similar analysis is done for the setting when one has full observability in Appendix B.11. With the following lemma we bound the sensitivity of the expected utility to the accuracy of our parameter estimates $\widehat{\boldsymbol{\xi}}$.

Lemma 4. *For the expected utility $F(\boldsymbol{\rho}; \boldsymbol{\xi})$ defined in* (4.15)*, let $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \boldsymbol{\xi})$, $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$ and define the suboptimality of $\widehat{\boldsymbol{\rho}}$ as $\operatorname{err}(\widehat{\boldsymbol{\rho}}) := F(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi})$. Then $\operatorname{err}(\widehat{\boldsymbol{\rho}})$ can be bounded by:*

$$\operatorname{err}(\widehat{\boldsymbol{\rho}}) \leq \sum_i \frac{1}{\widehat{\xi}_i \min\{\widehat{\xi}_i, \xi_i\}} \zeta_i (\widehat{\xi}_i - \xi_i)^2.$$

This lemma gives us hope that if we can learn $\widehat{\boldsymbol{\xi}}$ well enough such that $|\widehat{\xi}_i - \xi_i| \sim \mathcal{O}\left({}^1/\sqrt{T}\right)$ for all i, then we can obtain sub-linear regret by following the policy $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$. This indeed is possible and, in the next section, we show that an explore-and-commit algorithm can yield $\mathcal{O}(\sqrt{T})$ regret. We would like to also bring to the notice of the reader that a similar result up to constants can be shown for the Harmonic staleness objective proposed by Kolobov et al. [2019a] (See Appendix B.5) and the accumulating delay objective by Sia et al. [2007] (See Appendix B.6).

4.2.4 Explore-Then-Commit Algorithm

In this section, we will analyse a version of the explore-and-commit (ETC) algorithm for solving the web-crawling problem. The algorithm will first learn $\boldsymbol{\xi}$ by sampling all pages till time τ and then commit to the policy of observing the pages from time τ till T at the rates $\widehat{\boldsymbol{\rho}}$ as given by the Algorithm 2 in [Azar et al., 2018], obtained by passing it the estimated rates $\widehat{\boldsymbol{\xi}}$ instead of the true rates $\boldsymbol{\xi}$.

Revisiting the policy space. The constraint we had used to define the policy space Δ_R was that given any interval of width w, the expected number of refreshes in that interval should not exceed wR, which limited us to the class of Poisson policies. However, an alternative way to impose the constraint is to bound the time-averaged number of requests made per unit of time asymptotically. It can be shown that given our modelling assumptions that request and change processes are memory-less, the policy which maximizes the utility in (4.12) given a fixed number of observations per page will space them equally. This motivates a policy class $\mathcal{K}_R = \{\boldsymbol{\kappa} = (\kappa_1, \dots, \kappa_m) : \sum_{i=1}^{m} {}^{1}\!/\!{}_{\kappa_i} = R\} \subset \Pi$ as the set of deterministic policies which refresh webpage i at regular time intervals of length κ_i. Policies from $\mathcal{K}_R$ allow us to obtain tight confidence intervals for $\widehat{\boldsymbol{\xi}}$ by a straight-forward application of Lemma 3. However, the sensitivity of the utility function for this policy space to the quality of the estimated parameters is difficult to bound tightly. In particular, the differential utility function for this class of policies (defined in (4.13)) is not strongly concave, which is a basic building block of Lemma 4. This precludes performance bounds which are quadratic in the error of estimates $\widehat{\boldsymbol{\xi}}$, which lead to worse bounds on the regret of the ETC algorithm. These reasons are further expounded in the Appendix B.9. Nevertheless, we can show that using the uniform-intervals policy $\boldsymbol{\kappa}^{\text{UI}}$ incurs *lower* regret than the uniform-rates policy $\boldsymbol{\rho}^{\text{UR}}$, while still making on average R requests per unit time as shown in Appendix B.9.

Hence, to arrive at regret bounds, we will perform the exploration using *Uniform-interval exploration* policy $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$ which refreshes webpages at regular intervals $\forall i.\, \kappa_i = \frac{m}{R}$, which will allow us to use Lemma 3 to bound the error of the estimated $\widehat{\boldsymbol{\xi}}$ with high probability.

Lemma 5. *For a given $\delta \in (0,1)$, after following the uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ for time τ, which is assumed to be a multiplicity of ${}^{m}\!/\!{}_{R}$, we can claim the following for the error in the estimates $\widehat{\boldsymbol{\xi}}$ produced using the estimator proposed in Lemma 3:*

$$\mathbb{P}\left(\forall i \in [m] \colon |\widehat{\xi}_i - \xi_i| \le e^{\frac{\xi_{\max} m}{R}} \sqrt{\frac{R \log \frac{2m}{\delta}}{2\tau m}}\right) \ge 1 - \delta.$$

With these lemmas, we can bound the regret suffered by the ETC algorithm using the following Theorem.

Theorem 1. *Let π^{EC} denote the explore-and-commit algorithm which explores using the uniform-interval exploration policy for time τ (assumed to be a multiplicity of $\frac{m}{R}$), estimates $\widehat{\boldsymbol{\xi}}$ using the estimator proposed in (4.17), and then uses the policy $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho} \in \Delta_R} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$ till time T. Then for a given $\delta \in (0,1)$, with probability $1-\delta$, the expected regret of the explore and commit policy π^{EC} is bounded by:*

$$R(T, \pi^{\text{EC}}; \boldsymbol{\xi}) \le \left(\frac{\tau}{m} + \frac{(T-\tau)}{\tau} \frac{e^{2\frac{\xi_{\max} m}{R}} R \log\left({}^{2m}\!/\!{}_{\delta}\right)}{2m^2 \xi_{\min}^2}\right) \sum_{i=1}^{m} \zeta_i.$$

Further, we can choose an exploration horizon $\tau^\star \sim \mathcal{O}(\sqrt{T})$ such that, with probability $1-\delta$, the expected regret is $\mathcal{O}(\sqrt{T})$.

Proof. Since the utility of any policy is non-negative, we can upper-bound the regret of the algorithm in the exploration phase by the expected utility of the best stationary policy $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho} \in \Delta_R} F(\boldsymbol{\rho}; \boldsymbol{\xi})$, which is $\frac{\tau}{m} \sum_{i=1}^{m} \zeta_i \frac{\rho_i^\star}{\rho_i^\star + \xi_i} < \frac{\tau}{m} \sum_{i=1}^{m} \zeta_i$. In the exploitation phase, the regret is given by $\frac{T-\tau}{m}(F(\boldsymbol{\rho}^*, \boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}}, \boldsymbol{\xi}))$ (see (4.14)), which we bound using Lemma 4. Hence, we see that (with a slight abuse of notation to allow us to write $R(T, \boldsymbol{\kappa}^{\text{UI}}; \boldsymbol{\xi})$ for $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$):

$$\begin{aligned} R(T, \pi^{\text{EC}}; \boldsymbol{\xi}) &= R(\tau, \boldsymbol{\kappa}^{\text{UI}}; \boldsymbol{\xi}) + R(T - \tau, \widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) \\ &\le \frac{\tau}{m} \sum_{i=1}^{m} \zeta_i + \frac{(T-\tau)}{m} \sum_{i=1}^{m} \frac{\zeta_i (\widehat{\xi}_i - \xi_i)^2}{\widehat{\xi}_i \min\{\widehat{\xi}_i, \xi_i\}} \qquad (4.18) \end{aligned}$$

As we are using the estimator from Lemma 3, we have $\widehat{\xi}_i \min\{\widehat{\xi}_i, \xi_i\} \geq \xi_{\min}^2$. Using this and Lemma 5 with (4.18), we get with probability $1 - \delta$:

$$\begin{aligned}
R(T, \pi^{\mathrm{EC}}; \boldsymbol{\xi}) &\leq \tau \overbrace{\frac{1}{m} \sum_{i=1}^{m} \zeta_i}^{A} + (T - \tau) \frac{\sum_{i=1}^{m} \zeta_i}{m \xi_{\min}^2} (\widehat{\xi}_i - \xi_i)^2 \\
&= A\tau + (T - \tau) \frac{\sum_{i=1}^{m} \zeta_i}{m \xi_{\min}^2} \left(e^{\frac{\xi_{\max} m}{R}} \sqrt{\frac{R \log\left({}^{2m}/_{\delta}\right)}{2m\tau}} \right)^2 \\
&= A\tau + \frac{(T - \tau)}{\tau} \underbrace{\frac{\sum_{i=1}^{m} \zeta_i}{2m^2 \xi_{\min}^2} e^{2 \frac{\xi_{\max} m}{R}} R \log\left({}^{2m}/_{\delta}\right)}_{B} \\
&= A\tau + \frac{B\,T}{\tau} - B
\end{aligned} \tag{4.19}$$

This proves the first claim.

The bound in (4.19) takes the minimum value when $\tau^{\star} = \sqrt{\frac{B}{A}} \sqrt{T} = \sqrt{\frac{e^{2 \frac{\xi_{\max} m}{R}} R \log\left({}^{2m}/_{\delta}\right)}{2m\xi_{\min}^2}} \sqrt{T}$, giving with probability $1 - \delta$, the worst-case regret bound of:

$$R(T, \boldsymbol{\rho}^{\mathrm{EC}}; \boldsymbol{\xi}) < 2\sqrt{ABT}$$

This proves the second part of the theorem. $\square$

This theorem bounds the expected regret conditioned on the event that the crawler learns $\widehat{\boldsymbol{\xi}}$ such that $\forall i.\, |\widehat{\xi}_i - \xi_i| < \sqrt{\frac{\log {}^{2m}/_{\delta}}{{}^{2\tau R}/_{m}}}$. These kinds of guarantees have been seen in recent works [Rosenski et al., 2016, Avner and Mannor, 2014].

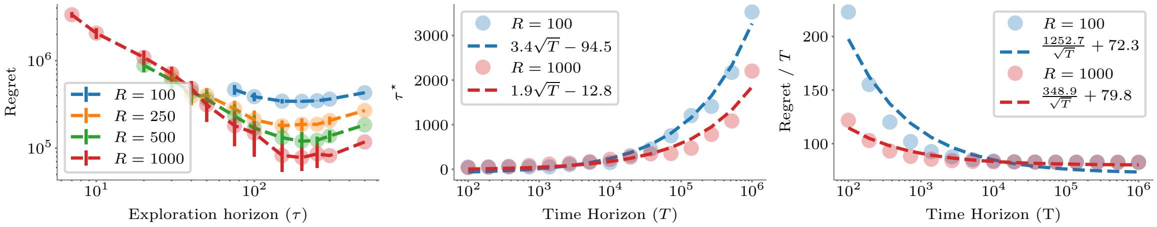
Note that the proof of the regret bounds can be easily generalised to the full-observation setting (See Appendix B.11) and for other objective functions (See Appendix B.5 and Appendix B.6).

Finally, note that using the doubling trick the regret bound can be made horizon independent at no extra cost. The policy can be de-randomized to either yield a fixed interval policy in $\mathcal{K}_R$ or, to a carousel like policy with similar performance guarantees [Azar et al., 2018, See Algorithm 3]. With this upper-bound on the regret of the ETC algorithm, in the next section we explore the empirical performance of the strategy.

4.2.5 Experimental Evaluation

We start with the analysis of the ETC algorithm, which shows empirically that the bounds that we have proven in Theorem 1 are tight up to constants. Next, we compare the ETC algorithm with phased ε-greedy algorithm and show that phased strategies can out-perform a well-tuned ETC algorithm, if given sufficient number of phases to learn. We leave the detailed analysis of this class of algorithms for later work. An empirical evaluation of the MLE estimator and the moment matching estimator for partial observations, and the associated confidence intervals proposed in Lemma 3 is done in Appendix B.10. These show that, for a variety of different parameters, the performance of the MLE estimator and the moment matching estimator is close to each other.

Evaluation of ETC Algorithm. For the experimental setup, we make user of the MSMACRO dataset [Kolobov et al., 2019b]. The dataset was collected over a period of 14 weeks by the production crawler for Bing. The crawler visited webpages approximately once per day. The crawl time and a binary indicator of whether the webpage had changed since the last crawl or not are included in the dataset along with an importance score for the various URLs. We sample 5000 webpages from the dataset while taking care to exclude pages which



(a) Regret / τ (with $T = 10^4$) (b) $\tau^\star$ / T (c) Normalized Regret / T

Figure 4.5: Performance of the ETC algorithm. Panel (a) shows the regret suffered with different exploration horizons (keeping $T = 10^4$ fixed) showing that a minima exists. Panel (b) shows that the optimal value of the horizon scales as $\mathcal{O}(\sqrt{T})$ while panel (c) shows that the time-horizon normalized regret of the ETC algorithm decreases as $\mathcal{O}(1/\sqrt{T})$.

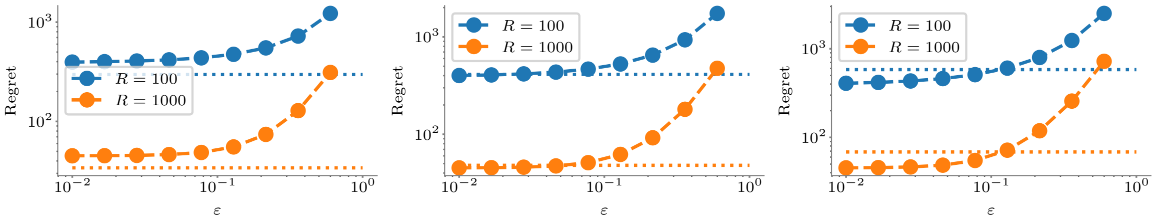
did not change at all or which changed upon every crawl so as to not constraint the estimates of the change rates artificially to $\xi_{\min}$ or $\xi_{\max}$. We calculate their rate of change ($\boldsymbol{\xi}$) using the MLE estimator (4.16). The corresponding importance values $\boldsymbol{\zeta}$ are also sampled from the importance value employed by the production web-crawler. We set $\xi_{\min} = 10^{-9}$ and $\xi_{\max} = 25$. The experiments simulate the change times $((x_{i,n})_{n=1}^{\infty})_{i \in [m]}$ for webpages 50 times with different random seeds and report quantities with standard deviation error bars, unless otherwise stated.

We first empirically determine the regret for different exploration horizon τ and bandwidth parameter R. To this end, we run a grid search for different values of τ (starting from the minimum time required to sample each webpage at least once), simulate the exploration phase using uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ and simulated change times to determine the parameters $\widehat{\boldsymbol{\xi}}$ and the regret suffered during the exploration phase. We calculate $\widehat{\boldsymbol{\rho}}$ using Algorithm 2 of Azar et al. [2018], similarly calculate $\boldsymbol{\rho}^\star$ using the true parameters $\boldsymbol{\xi}$, calculate their respective utility after the commit phase from τ till time horizon $T = 10^4$ using (4.14), and use it to determine the total regret suffered. We report the mean regret with the error bars indicating the standard deviations in Figure 4.5a. We see that there indeed is an optimal exploration horizon, as expected, and the value of both the horizon and the regret accumulated depends on R. We explore the relationship between the optimal exploration horizon $\tau^\star$ and the time horizon T next by varying T from 10^2 to 10^6 and calculating the optimal horizon $\tau^\star$ (using ternary search to minimize the empirical mean of the utility) for $R \in \{10^2, 10^3\}$; plots for other values of R are qualitatively similar. Figure 4.5b shows that the optimal exploration horizon $\tau^\star$ scales as $\mathcal{O}(\sqrt{T})$.

Finally, we plot the time-horizon normalized regret suffered by π^{EC} when using the optimal exploration horizon $\tau^\star$ in Figure 4.5c. We see that the normalized regret decreases as $\frac{1}{\sqrt{T}}$, as postulated by Theorem 1. Plots for different values of $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ are qualitatively similar. It can also be seen in all plots that if the allocated bandwidth R is high, then the regret suffered is lower but the dependence of the optimal exploration threshold $\tau^\star$ on the available bandwidth is non-trivial: in Figure 4.5b, we see that the $\tau^\star_{R=100} < \tau^\star_{R=1000}$ if $T < 10^3$ and $\tau^\star_{R=100} > \tau^\star_{R=1000}$ if $T > 10^4$.

It is noteworthy that Figures 4.5b and 4.5c suggest that the bounds we have proven in Theorem 1 are tight up to constants.

Phased ε-greedy Algorithm. In Section 4.2.4, we have shown guarantees on the performance of explore-then-commit algorithm which is the simplest form of policies which adheres to the properties we mention in Section 4.2.2. However, it is not difficult to imagine other strategies which conform to the same recommendations. For example, consider a phased ε-greedy algorithm which runs with $\boldsymbol{\rho}^{\text{UR}}$ for duration given in Lemma 1, estimates $\widehat{\boldsymbol{\xi}}$, calculates $\widehat{\boldsymbol{\rho}}$, and then follows the policy $\boldsymbol{\rho}^{\varepsilon}$, where $\rho_i^{\varepsilon} = (1 - \varepsilon)\widehat{\rho}_i + \varepsilon \frac{R}{m}$, and then starts another phase, improving its policy with improving estimates of $\widehat{\boldsymbol{\xi}}$. Since $\forall i \in [m].\, \rho_i^{\varepsilon} > \varepsilon \frac{R}{m}$, the policy will continue exploring all the webpages, ensuring eventual consistency in the estimates of $\widehat{\boldsymbol{\xi}}$.



(a) $T = 10^{3.67}$ / 3 phases (b) $T = 10^{3.83}$ / 6 phases (c) $T = 10^4$ / 9 phases

Figure 4.6: Performance of the phased ε greedy algorithm. The dotted lines show the regret of the ETC algorithm, with optimal exploration horizon, same bandwidth R and time horizon T. While the ETC algorithm performs well when the number of phases is small, with increasing number of phases, the phased ε-greedy algorithm is able to obtain lower regret than ETC.

We performed simulations with the $\boldsymbol{\rho}^{\varepsilon}$ algorithm and found that though it performed worse than ETC for small time horizons (Figures 4.6a and 4.6b), it performed better when given sufficient number of phases (see Figure 4.6c). Exploring the regret bounds of such policies is part of our future work.

4.2.6 Conclusion

In this section, we have taken the first step towards solving the problem of learning changing rates of web-pages while solving the web-crawling problem, while also providing a guiding framework for analysing online learning problems where the agent's policy can be described using a Poisson process. We have shown that the learning algorithms should be *phased* and there are restrictions on how much they can change the policy from step to step while keeping learning feasible. Further, by bounding the performance of a Poisson policy using a deterministic policy, we have proved that a simple explore-and-commit policy has $\mathcal{O}(\sqrt{T})$ regret under mild assumptions about the parameters.

Chapter 5

Conclusions and Future work

The first part of the thesis proposes two latent variable based models for, on the one hand, capturing the opinions and expertise of users and, on the other hand, explaining the complexity of online discussions and the *value* of knowledge. They both leverage signals from detailed user activity history left on online platforms and help us gain insights about the process of collaborative dissemination of information and knowledge. These insights, in turn, can help the platforms improving the dissemination process, *e.g.*, by improving the incentive design (*e.g.*, importance of encouraging downvotes on Q&A forums), or by quantifying complexity of discourse (*e.g.*, by highlighting different axes of disagreement).

In the second part, the thesis explores the problems faced during competitive dissemination on social media and describes methods for addressing those problems. The novel reinforcement learning based method for determining when a broadcaster should make posts is a general method which could be applied to any other problem which requires controlling a marked temporal process. Examples of such settings are the problem of scheduling reviews to optimize learning and for trading on the stock market. The problem of measuring the visibility of one's posts on the feeds of the followers leads to exploration of refreshing policies in a general setting which can be used for web-caching or for caching databases.

With this, there are several interesting future directions left to explore.

For example, the proposed measure of complexity of discussions—the dimension of the latent space of opinions—may be a good starting point to develop theoretically grounded measures of polarization [Choi et al., 2010, Mejova et al., 2014, Conover et al., 2011] and controversy [Guerra et al., 2013, Garimella et al., 2018], which have been lacking in the literature. Moreover, it would be very interesting to augment the modeling framework to also incorporate, in addition to the voting data, the textual information in the comments, the identity of commenters, and their trust-worthiness. Besides increasing the accuracy of our method, these may aid the interpretability of the dimensions as well. Our algorithm for determining the minimum dimension under which the opinion space is able to explain the voting data exhibits weak theoretical guarantees though it performs well on real-data. It would be interesting to develop exact algorithm by adapting recent advances in exact sign-rank estimation [Bhangale and Kopparty, 2015, Basri et al., 2009].

Natural follow-ups to potentially improve the expressiveness of our Crowdlearning modeling framework include:

- Consider more complex off-site learning trends, *e.g.*, isotonic regression Kakade et al. [2011] or exponential/power-law Heathcote et al. [2000].
- Allow for a knowledge item to have different knowledge values per user by considering a knowledge distribution per item, and use Bayesian inference Murphy [2012b] to learn the model parameters.

- Consider linguistic changes in users' contributions according to their expertise Danescu-Niculescu-Mizil et al. [2013].
- Perform a non-parametric estimation of the kernels that model the users' forgetting process.
- Incorporate incentives mechanisms such as badges or flairs, which are often used in crowdlearning sites Anderson et al. [2013] and MOOCs Anderson et al. [2014].
- Allow for knowledge overlaps between knowledge items, as observed in real data Babaei et al. [2015], by learning submodular functions Balcan and Harvey [2011].

One of the key modeling ideas behind the framework is realizing that users' contributions can be viewed as noisy discrete samples of the users' expertise at points localized (non-uniformly) in time. It would be very interesting to generalize this idea to any type of event data and derive sampling theorems and conditions under which an underlying general continuous signal of interest (be it user's expertise, opinion, or wealth) can be recovered from event data with provable guarantees. Finally, we experimented with data gathered exclusively from Stack Overflow, however, it would be interesting to apply our model (or augmented version of our model) to Stack Exchange at large, to other questions and answers websites (*e.g.*, AskReddit), microblogging platforms (*e.g.*, Twitter), social networking sites (*e.g.*, Pinterest), or even MOOC platforms (*e.g.*, Coursera).

The work on using reinforcement learning algorithms for MTPPs is but the first step towards opening the road to deriving more sophisticated reinforcement learning algorithms, *e.g.*, actor-critic algorithms, for the novel problem setting. We have evaluated in two real-world applications in personalized teaching and viral marketing, however, there are many other (high impact) applications fitting our novel problem setting, *e.g.*, quantitative trading. Finally, it would be very interesting to develop multiple agent reinforcement learning algorithms for MTPPs which will be able to model competitions and collaborations among multiple agents on social media.

With consideration to our work on online estimation of visibility of one's post, though we have proved a theoretical upper bound on regret of $\mathcal{O}(\sqrt{T})$ and have empirically seen that bound is tight up to constants for the explore-and-commit algorithm, it is not clear whether this bound is tight for the class of all *phased* strategies. Exploring the class of such strategies is a planned extension. Lastly, we believe there are rich connections worth exploring between the proposed algorithm and the recent work on the Recharging Bandits paradigm [Immorlica and Kleinberg, 2018].

Bibliography

O. Aalen, O. Borgan, and H. Gjessing. *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008a.

Odd Aalen, Ornulf Borgan, and Hakon Gjessing. *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008b.

Noga Alon, Peter Frankl, and V Rodl. Geometrical realization of set systems and probabilistic communication complexity. In *FOCS*, 1985.

Noga Alon, Shay Moran, and Amir Yehudayoff. Sign rank versus vc dimension. In *Conference on Learning Theory*, 2016.

Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. Steering user behavior with badges. In *22nd International Conference on World Wide Web*, 2013.

Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. Engaging with massive online courses. In *23rd International Conference on World Wide Web*, 2014.

Lee Averell and Andrew Heathcote. The form of the forgetting curve and the fate of memories. *Journal of Mathematical Psychology*, 55(1), 2011.

Orly Avner and Shie Mannor. Concurrent bandits and cognitive radio networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 66–81. Springer, 2014.

R. Axelrod. The dissemination of culture a model with local convergence and global polarization. *Journal of conflict resolution*, 41(2):203–226, 1997.

Yossi Azar, Eric Horvitz, Eyal Lubetzky, Yuval Peres, and Dafna Shahaf. Tractable near-optimal policies for crawling. *Proceedings of the National Academy of Sciences*, 115(32):8099–8103, 2018.

Mahmoudreza Babaei, Przemyslaw Grabowicz, Isabel Valera, and Manuel Gomez-Rodriguez. Quantifying information overload in social media and its impact on social contagions. In *9th AAAI Conference on Weblogs and Social Media*, 2015.

Ryan SJD Baker, Zachary A Pardos, Sujith M Gowda, Bahador B Nooraei, and Neil T Heffernan. Ensembling predictions of student knowledge within intelligent tutoring systems. In *User Modeling, Adaption and Personalization*. Springer, 2011.

Maria-Florina Balcan and Nicholas JA Harvey. Learning submodular functions. In *forty-third annual ACM symposium on Theory of computing*. ACM, 2011.

Pablo Barberá. Birds of the same feather tweet together: Bayesian ideal point estimation using twitter data. *Political Analysis*, 23(1):76–91, 2015.

Ronen Basri, Pedro F Felzenszwalb, Ross B Girshick, David W Jacobs, and Caroline J Klivans. Visibility constraints on features of 3d objects. In *CVPR*, 2009.

Joseph E Beck and Jack Mostow. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. In *Intelligent Tutoring Systems*, 2008.

James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA, 2007.

Jonah Berger and Katherine L Milkman. What makes online content viral? *Journal of marketing research*, 49(2):192–205, 2012.

Amey Bhangale and Swastik Kopparty. The complexity of computing the minimum rank of a sign pattern matrix. *arXiv preprint arXiv:1503.04486*, 2015.

Sonia A Bhaskar and Adel Javanmard. 1-bit matrix completion under exact low-rank constraint. In *CISS*, 2015.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 2003.

Alexandre Bovet and Hernán A Makse. Influence of fake news in twitter during the 2016 us presidential election. *Nature communications*, 10(1):7, 2019.

Jürgen Broß. Aspect-oriented sentiment analysis of customer reviews using distant supervision techniques. *Doctoral dissertation*, 2013.

Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and P Krishna Gummadi. Measuring user influence in twitter: The million follower fallacy. *ICWSM*, 10(10-17):30, 2010.

Junghoo Cho and Hector Garcia-Molina. Effective page refresh policies for web crawlers. *ACM Transactions on Database Systems (TODS)*, 28(4):390–426, 2003a.

Junghoo Cho and Hector Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology (TOIT)*, 3(3):256–290, 2003b.

Yoonjung Choi, Yuchul Jung, and Sung-Hyon Myaeng. Identifying controversial issues and their sub-topics in news articles. In *Pacific-Asia Workshop on Intelligence and Security Informatics*, pages 140–153. Springer, 2010.

Michael Conover, Jacob Ratkiewicz, Matthew R Francisco, Bruno Gonçalves, Filippo Menczer, and Alessandro Flammini. Political polarization on Twitter. *ICWSM*, 2011.

Efthymios Constantinides. Foundations of social media marketing. *Procedia-Social and behavioral sciences*, 148:40–57, 2014.

Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 1994.

Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. No country for old members: User lifecycle and linguistic change in online communities. In *22nd International conference on World Wide Web*, 2013.

Felicia Day. *You're Never Weird on the Internet (almost): A Memoir*. Simon and Schuster, 2015.

Abir De, Sourangshu Bhattacharya, Parantapa Bhattacharya, Niloy Ganguly, and Soumen Chakrabarti. Learning a linear influence model from transient opinion dynamics. In *CIKM*, 2014.

Abir De, Adish Singla, Utkarsh Upadhyay, and Manuel Gomez-Rodriguez. Can a User Guess What Her Followers Want? *ACM Workshop on Behavioral EC*, 2019.

Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340, 2008.

Morris H DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

Anthony Downs. An economic theory of political action in a democracy. *Journal of political economy*, 65(2): 135–150, 1957.

Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1):219–245, 2000.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*, 2016.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *ICML*, 2016.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279. ACM, 2008.

H. Ebbinghaus. *Memory: a contribution to experimental psychology.* Teachers College, Columbia University, 1885.

Carsten Eickhoff, Jaime Teevan, Ryen White, and Susan Dumais. Lessons from the journey: A query log analysis of within-session learning. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining.* ACM, 2014. ISBN 978-1-4503-2351-2. doi: 10.1145/2556195.2556217. URL http://doi.acm.org/10.1145/2556195.2556217.

Susan E Embretson and Steven P Reise. *Item response theory.* Psychology Press, 2013.

M. Farajtabar, N. Du, M. Gomez-Rodriguez, I. Valera, L. Song, and H. Zha. Shaping social activity by incentivizing users. In *Advances in Neural Information Processing Systems*, 2014.

Mehrdad Farajtabar. *Point Process Modeling and Optimization of Social Networks.* PhD thesis, Georgia Institute of Technology, 2018.

Mehrdad Farajtabar, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Khalil, Shuang Li, Le Song, and Hongyuan Zha. Fake news mitigation via point process based intervention. In *ICML*, 2017.

Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS computational biology*, 9(4):e1003024, 2013.

Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. Quantifying controversy on social media. *ACM Transactions on Social Computing*, 1(1):3, 2018.

Saptarshi Ghosh, Naveen Sharma, Fabricio Benevenuto, Niloy Ganguly, and Krishna Gummadi. Cognos: crowdsourcing search for topic experts in microblogs. In *35th international ACM SIGIR conference on Research and development in information retrieval.* ACM, 2012.

Jose Gonzalez-Brenes and Jack Mostow. What and when do students learn? fully data-driven joint estimation of cognitive and student models. In *6th International Conference on Educational Data Mining*, 2013.

Pedro Henrique Calais Guerra, Wagner Meira Jr, Claire Cardie, and Robert Kleinberg. A measure of polarization on social media networks based on community boundaries. In *ICWSM*, 2013.

Benjamin V. Hanrahan, Gregorio Convertino, and Les Nelson. Modeling problem difficulty and expertise in Stack Overflow. In *ACM 2012 Conference on Computer Supported Cooperative Work Companion*. ACM, 2012. ISBN 978-1-4503-1051-2. doi: 10.1145/2141512.2141550. URL http://doi.acm.org/10.1145/2141512.2141550.

Floyd B Hanson. *Applied stochastic processes and control for Jump-diffusions: modeling, analysis, and computation*, volume 13. Siam, 2007.

Andrew Heathcote, Scott Brown, and DJK Mewhort. The power law repealed: The case for an exponential law of practice. *Psychonomic bulletin & review*, 7(2), 2000.

R. Hegselmann and U. Krause. Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3), 2002.

Nathan Hodas and Kristina Lerman. How visibility and divided attention constrain social contagion. *SocialCom*, 2012.

P. Holme and M. E. Newman. Nonequilibrium phase transition in the coevolution of networks and opinions. *Physical Review E*, 74(5):056108, 2006.

Cho-Jui Hsieh, Kai-Yang Chiang, and Inderjit S Dhillon. Low rank modeling of signed networks. In *KDD*, 2012.

Nicole Immorlica and Robert D Kleinberg. Recharging bandits. *IEEE 59th Annual Symposium on Foundations of Computer Science*, 2018.

How Jing and Alexander J Smola. Neural survival recommender. In *WSDM*, 2017.

Dejan Jovanović and Leonardo de Moura. Solving non-linear arithmetic. In *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*, volume 7364 of *Lecture Notes in Computer Science*, pages 339–354. Springer, 2012.

Pawel Jurczyk and Eugene Agichtein. Discovering authorities in question answer communities by using link analysis. In *16th ACM Conference on Conference on Information and Knowledge Management*. ACM, 2007. ISBN 978-1-59593-803-9. doi: 10.1145/1321440.1321575. URL http://doi.acm.org/10.1145/1321440.1321575.

Sham M. Kakade, Varun Kanade, Ohad Shamir, and Adam Kalai. Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems*, 2011.

Jeon-Hyung Kang and Kristina Lerman. Vip: Incorporating human cognitive biases in a probabilistic model of retweeting. In *Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 101–110. Springer, 2015.

Wei-Chen Kao, Duen-Ren Liu, and Shiu-Wen Wang. Expert finding in question-answering websites. In *ACM Symposium on Applied Computing*. ACM, 2010. ISBN 9781605586397. doi: 10.1145/1774088.1774266. URL http://dl.acm.org/citation.cfm?id=1774088.1774266.

Mohammad Reza Karimi, Erfan Tavakoli, Mehrdad Farajtabar, Le Song, and Manuel Gomez Rodriguez. Smart broadcasting: Do you want to be seen? In *KDD*, 2016.

Jooyeon Kim, Behzad Tabibian, Alice Oh, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Leveraging the crowd to detect and reduce the spread of fake news and misinformation. In *WSDM*, 2018.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014.

John Kingman. *Poisson Processes*. Oxford Science Publications, 1993.

Robert Kleinberg and Nicole Immorlica. Recharging bandits. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 309–319, 2018. doi: 10.1109/FOCS.2018.00037. URL https://doi.org/10.1109/FOCS.2018.00037.

Andrey Kolobov, Yuval Peres, Cheng Lu, and Eric Horvitz. Staying up to date with online content changes using reinforcement learning for scheduling. May 2019a. URL https://www.microsoft.com/en-us/research/publication/staying-up-to-date-with-online-content-changes-using-reinforcement-learning-for-scheduling/.

Andrey Kolobov, Yuval Peres, Eyal Lubetzky, and Eric Horvitz. Optimal freshness crawl under politeness constraints. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 495–504. ACM, 2019b.

Yehuda Koren and Robert Bell. Advances in collaborative filtering. In *Recommender systems handbook*, pages 77–118. Springer, 2015.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

Stevens Le Blond, Cédric Gilbert, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and David R Choffnes. A broad view of the ecosystem of socially engineered exploit documents. In *NDSS*, 2017.

Nathaniel Leibowitz, Barak Baum, Giora Enden, and Amir Karniel. The exponential learning equation as a function of successful trials results in sigmoid performance. *Journal of Mathematical Psychology*, 54(3), 2010. ISSN 00222496. doi: 10.1016/j.jmp.2010.01.006. URL http://linkinghub.elsevier.com/retrieve/pii/S0022249610000179.

Kristina Lerman and Tad Hogg. Leveraging position bias to improve peer recommendation. *PloS one*, 9(6), 2014.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, (1):76–80, 2003.

Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1): 1–167, 2012.

Geoffrey R Loftus. Evaluating forgetting curves. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 11(2), 1985.

Lars Lorch, Abir De, Samir Bhatt, William Trouleau, Utkarsh Upadhyay, and Manuel Gomez-Rodriguez. Stochastic optimal control of epidemic processes in networks. *arXiv preprint arXiv:1810.13043*, 2018.

Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010.

Barbara Means, Yukie Toyama, Robert Murphy, Marianne Bakia, and Karla Jones. Evaluation of evidence-based practices in online learning: A meta-analysis and review of online learning studies. *US Department of Education*, 2009.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *NIPS*, 2017.

Yelena Mejova, Amy X Zhang, Nicholas Diakopoulos, and Carlos Castillo. Controversy and sentiment in online news. *arXiv preprint arXiv:1409.8152*, 2014.

Samuel Merrill and Bernard Grofman. *A unified theory of voting: Directional and proximity spatial models*. Cambridge University Press, 1999.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, 2016.

Kevin Murphy. *Machine learning: a probabilistic perspective*. MIT Press, Cambridge, UK, 2012a.

Kevin P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012b.

Gergely Neu, Andras Antos, András György, and Csaba Szepesvári. Online markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems*, pages 1804–1812, 2010.

Geoffrey R Norman and Henk G Schmidt. The psychological basis of problem-based learning: A review of the evidence. *Academic Medicine*, 67(9), 1992.

Aditya Pal and Scott Counts. Identifying topical authorities in microblogs. In *Fourth ACM International Conference on Web Search and Data Mining*. ACM, 2011. ISBN 978-1-4503-0493-1. doi: 10.1145/1935826.1935843. URL http://doi.acm.org/10.1145/1935826.1935843.

Aditya Pal and Joseph A. Konstan. Expert identification in community question answering: Exploring question selection bias. In *19th ACM International Conference on Information and Knowledge Management*, 2010. ISBN 978-1-4503-0099-5. doi: 10.1145/1871437.1871658. URL http://doi.acm.org/10.1145/1871437.1871658.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24 (4):694–707, 2016.

Bo Pang, Lillian Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

Harold Pashler, Nicholas Cepeda, Robert V Lindsey, Ed Vul, and Michael C Mozer. Predicting the optimal spacing of study: A multiscale context model of memory. In *NIPS*, 2009.

Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. In *FOCS*, 1984.

Adamska Paulina and Juzwin Marta. Study of the Temporal-Statistics-Based Reputation Models for Q&A Systems. *Computer Science*, 16(3), 2015. ISSN 1508-2806. doi: 10.7494/csci.2015.16.3.253. URL http://journals.agh.edu.pl/csci/article/view/1342.

Philip I Pavlik, Hao Cen, and Kenneth R Koedinger. Performance factors analysis–a new alternative to knowledge tracing. In *2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*. IOS Press, 2009.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, 2015.

David Pollard. A few good inequalities. Technical report, 2015. http://www.stat.yale.edu/ pollard/Books/Mini/BasicMG.pdf, accessed on 29.10.2019.

Daryl Posnett, Eric Warburg, Premkumar T. Devanbu, and Vladimir Filkov. Mining stack exchange: Expertise is evident from initial contributions. In *International Conference on Social Informatics*, 2012. doi: 10.1109/SocialInformatics.2012.67. URL http://dx.doi.org/10.1109/SocialInformatics.2012.67.

Jiezhong Qiu, Jie Tang, Tracy Xiao Liu, Jie Gong, Chenhui Zhang, Qian Zhang, and Yufei Xue. Modeling and predicting learning behavior in moocs. In *Fourth ACM International Conference on Web Search and Data Mining*. ACM, 2016.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, pages 1–20, 2010.

Jonathan Rosenski, Ohad Shamir, and Liran Szlak. Multi-player bandits–a musical chairs approach. In *International Conference on Machine Learning*, pages 155–163, 2016.

Charles K Rowley. The relevance of the median voter theorem. *Zeitschrift für die gesamte Staatswissenschaft/Journal of Institutional and Theoretical Economics*, (H. 1):104–126, 1984.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

Burr Settles and Brendan Meeder. A trainable spaced repetition model for language learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1848–1858, 2016.

Ka Cheung Sia, Junghoo Cho, and Hyun-Kyu Cho. Efficient monitoring algorithm for fast news alerts. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):950–961, 2007.

Christopher H Skinner. Applied comparative effectiveness researchers must measure learning rates: A commentary on efficiency articles. *Psychology in the Schools*, 47(2), 2010.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Nemanja Spasojevic, Zhisheng Li, Adithya Rao, and Prantik Bhattacharyya. When-to-post on social networks. In *KDD*, 2015.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Behzad Tabibian, Utkarsh Upadhyay, Abir De, Ali Zarezade, Bernhard Schoelkopf, and Manuel Gomez-Rodriguez. Enhancing human learning via spaced repetition optimization. In *Proceedings of the National Academy of Sciences*, 2019a.

Behzad Tabibian, Utkarsh Upadhyay, Abir De, Ali Zarezade, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences*, page 201815156, 2019b.

Alfred Tarski. A decision method for elementary algebra and geometry. In *Quantifier elimination and cylindrical algebraic decomposition*, pages 24–84. Springer, 1998.

Utkarsh Upadhyay, Isabel Valera, and Manuel Gomez-Rodriguez. Uncovering the dynamics of crowdlearning and the value of knowledge. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 61–70. ACM, 2017.

Utkarsh Upadhyay, Abir De, and Manuel Gomez-Rodriguez. Deep reinforcement learning of marked temporal point processes. In *Advances in Neural Information Processing Systems*, 2018.

Utkarsh Upadhyay, De Abir, Pappu Aasish, and Manuel Gomez-Rodriguez. On the complexity of opinions and online discussions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019.

Utkarsh Upadhyay, Robert Busa-Fekete, Wojciech Kotlowski, David Pal, and Balazs Szorenyi. Learning to Crawl. In *34th AAAI Conference on Artificial Intelligence*, Feb 2020.

I. Valera and M. Gomez-Rodriguez. Modeling adoption and usage of competing products. In *2015 IEEE International Conference on Data Mining*, 2015.

Eleni Vasilaki, Nicolas Frémaux, Robert Urbanczik, Walter Senn, and Wulfram Gerstner. Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail. *PLoS computational biology*, 5(12):e1000586, 2009.

Yichen Wang, Grady Williams, Evangelos Theodorou, and Le Song. Variational policy for guiding point processes. In *ICML*, 2017.

Yichen Wang, Evangelos Theodorou, Apurv Verma, and Le Song. A stochastic differential equation framework for guiding online user activities in closed loop. In *AISTATS*, 2018.

Emo Welzl. Partition trees for triangle counting and other range searching problems. In *Proceedings of the fourth annual symposium on Computational geometry*, pages 23–33. ACM, 1988.

Ryen W. White, Susan T. Dumais, and Jaime Teevan. Characterizing the influence of domain expertise on web search behavior. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. ACM, 2009. ISBN 978-1-60558-390-7. doi: 10.1145/1498759.1498819. URL http://doi.acm.org/10.1145/1498759.1498819.

Daan Wierstra, Alexander Foerster, Jan Peters, and Juergen Schmidhuber. Solving deep memory POMDPs with recurrent policy gradients. In *ICANN*, 2007.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

E. Yildiz, A. Ozdaglar, D. Acemoglu, A. Saberi, and A. Scaglione. Binary opinion dynamics with stubborn agents. *ACM Transactions on Economics and Computation*, 1(4):19, 2013.

M. E. Yildiz, R. Pagliari, A. Ozdaglar, and A. Scaglione. Voting models in random networks. In *Information Theory and Applications Workshop*, pages 1–7, 2010.

Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. Individualized bayesian knowledge tracing models. In *Artificial Intelligence in Education*. Springer, 2013.

A. Zarezade, U. Upadhyay, H. Rabiee, and M. Gomez-Rodriguez. Redqueen: An online algorithm for smart broadcasting in social networks. In *WSDM*, 2017a.

A. Zarezade, A. De, U. Upadhyay, H. Rabiee, and M. Gomez-Rodriguez. Steering social activity: A stochastic optimal control point of view. *JMLR*, 2018.

Ali Zarezade, Abir De, Utkarsh Upadhyay, Hamid R Rabiee, and Manuel Gomez-Rodriguez. Steering social activity: A stochastic optimal control point of view. *Journal of Machine Learning Research*, 18:205–1, 2017b.

Jun Zhang, Mark S. Ackerman, and Lada Adamic. Expertise networks in online communities: Structure and algorithms. In *16th International Conference on World Wide Web*, 2007. ISBN 978-1-59593-654-7. doi: 10.1145/1242572.1242603. URL http://doi.acm.org/10.1145/1242572.1242603.

Ke Zhou, Hongyuan Zha, and Le Song. Learning triggering kernels for multi-dimensional hawkes processes. In *30th International Conference on Machine Learning*, 2013.

Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23(4), 1997. ISSN 0098-3500. doi: 10.1145/279232.279236. URL http://doi.acm.org/10.1145/279232.279236.

Appendix A

Appendix: Deep reinforcement Learning for *when-to-post* problem

A.1 Proof of Proposition 4

We first start by rewriting the expected reward function $J(\theta)$ as:

$$
\begin{aligned}
J(\theta) &= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)}\left[R^*(T)\right] = \mathbb{E}_{|\mathcal{A}_T|,|\mathcal{F}_T|}\left[\mathbb{E}_{\mathcal{A}_T,\mathcal{F}_T \,|\, |\mathcal{A}_T|,|\mathcal{F}_T|}\left[R(T) \,|\, |\mathcal{A}_T|, |\mathcal{F}_T|\right]\right] \\
&= \sum_{m,k} \mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda^*_\theta(t_i) m^*_\theta(y_i) \right) \exp\left(- \int_0^T \lambda^*_\theta(s)\, ds \right) \\
&\times \mathbb{P}(|\mathcal{F}_T| = k) \left(\prod_{j \in \mathcal{F}_T} \int_{t_j, z_j} \lambda^*_\phi(t_j) m^*_\phi(z_j) \right) \exp\left(- \int_0^T \lambda^*_\phi(s)\, ds \right) R^*(T) \\
&\times \prod_{i \in \mathcal{A}_T} dt_i dy_i \prod_{j \in \mathcal{F}_T} dt_j dz_j,
\end{aligned}
$$

where we have first taken the expectation with respect to all histories conditioned on a given number of events and then taken the expectation with respect to the number of events. Then, we can compute the gradient $\nabla_\theta J(\theta)$ as follows:

$$\begin{aligned}
\nabla_\theta J(\theta) = &\sum_{m,k} \nabla_\theta \left\{ \mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right) \right\} \\
&\times \mathbb{P}(|\mathcal{F}_T| = k) \left(\prod_{j \in \mathcal{F}_T} \int_{t_j, z_j} \lambda_\phi^*(t_j) m_\phi^*(z_j) \right) \exp\left(- \int_0^T \lambda_\phi^*(s)\, ds \right) R^*(T) \\
&\times \prod_{i \in \mathcal{A}_T} d\,t_i d\,y_i \prod_{j \in \mathcal{F}_T} d\,t_j d\,z_j \\
&= \sum_{m,k} \frac{\nabla_\theta \left\{ \mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right) \right\}}{\mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right)} \\
&\times \mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right) \\
&\times \mathbb{P}(|\mathcal{F}_T| = k) \left(\prod_{j \in \mathcal{F}_T} \int_{t_j, z_j} \lambda_\phi^*(t_j) m_\phi^*(z_j) \right) \exp\left(- \int_0^T \lambda_\phi^*(s)\, ds \right) R^*(T) \\
&\times \prod_{i \in \mathcal{A}_T} d\,t_i d\,y_i \prod_{j \in \mathcal{F}_T} d\,t_j d\,z_j \\
&= \sum_{m,k} \nabla_\theta \left\{ \log \left(\mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right) \right) \right\} \\
&\times \mathbb{P}(|\mathcal{A}_T| = m) \left(\prod_{i \in \mathcal{A}_T} \int_{t_i, y_i} \lambda_\theta^*(t_i) m_\theta^*(y_i) \right) \exp\left(- \int_0^T \lambda_\theta^*(s)\, ds \right) \\
&\times \mathbb{P}(|\mathcal{F}_T| = k) \left(\prod_{j \in \mathcal{F}_T} \int_{t_j, z_j} \lambda_\phi^*(t_j) m_\phi^*(z_j) \right) \exp\left(- \int_0^T \lambda_\phi^*(s)\, ds \right) R^*(T) \\
&\times \prod_{i \in \mathcal{A}_T} d\,t_i d\,y_i \prod_{j \in \mathcal{F}_T} d\,t_j d\,z_j \\
&= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \right]
\end{aligned}$$

where we have used that $\frac{\nabla_\theta f(\theta)}{f(\theta)} = \nabla_\theta \log f(\theta)$ and

$$\log \mathbb{P}_\theta(\mathcal{A}_T) = \sum_{e_i \in \mathcal{A}_T} \left(\log \lambda_\theta^*(t_i) + \log m_\theta^*(z_i) \right) - \int_0^T \lambda_\theta^*(s)\, ds.$$

A.2 Proof of Proposition 5

We first start by rewriting the penalized expected reward function $J_r(\theta)$ as:

$$\begin{aligned}
J_r(\theta) &= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) - q_l \int_0^T g_\lambda(\lambda^*_\theta(t))dt - q_m \int_0^T g_m(m^*_\theta(t))dt \right] \\
&= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) \right] - q_l \, \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g_\lambda(\lambda^*_\theta(t))dt \right] \\
&\quad - q_m \, \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g_m(m^*_\theta(t))dt \right],
\end{aligned}$$

where we have just used the linearity of the expectation. Then, we can use Proposition 4 and the chain rule to compute the gradient $\nabla_\theta J_r(\theta)$:

$$\begin{aligned}
\nabla_\theta J_r(\theta) &= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[R^*(T) \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \right] \\
&\quad - q_l \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g_\lambda(\lambda^*_\theta(t))dt \, \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \right] \\
&\quad - q_l \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g'_\lambda(\lambda^*_\theta(t)) \nabla_\theta \lambda^*_\theta(t) dt \right] \\
&\quad - q_m \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g_m(m^*_\theta(t))dt \, \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \right] \\
&\quad - q_m \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \left[\int_0^T g'_m(m^*_\theta(t)) \nabla_\theta m^*_\theta(t) dt \right] \\
&= \mathbb{E}_{\mathcal{A}_T \sim p^*_{\mathcal{A};\theta}(\cdot), \mathcal{F}_T \sim p^*_{\mathcal{F};\phi}(\cdot)} \Bigg[\\
&\qquad \left(R^*(T) - q_l \int_0^T g_\lambda(\lambda^*_\theta(t)) \, dt - q_m \int_0^T g_m(m^*_\theta(t)) \, dt \right) \nabla_\theta \log \mathbb{P}_\theta(\mathcal{A}_T) \\
&\qquad - \left(q_l \int_0^T g'_\lambda(\lambda^*_\theta(t)) \nabla_\theta \lambda^*_\theta(t) \, dt + q_m \int_0^T g'_m(m^*_\theta(t)) \nabla_\theta m^*_\theta(t) \, dt \right) \Bigg]
\end{aligned}$$

where $g'_\lambda(\lambda^*_\theta(t)) = \frac{d\, g_\lambda(\lambda^*_\theta(t))}{d\, \lambda^*_\theta(t)}$ and $g'_m(m^*_\theta(t)) = \frac{d\, g_m(m^*_\theta(t))}{d\, m^*_\theta(t)}$.

A.3 Sampling event times from the intensity $\lambda^*_\theta(t)$

Immediately after taking an action at time t_i, the agent has to determine the time of the next action t_{i+1} by sampling from the intensity function $\lambda^*_\theta(t)$ given by Eq. 4.5. However, if a feedback event arrives at time $s < t_{i+1}$, *i.e.*, the feedback event arrives *before* the agent has performed her next action, then the intensity function $\lambda^*_\theta(t)$ will need to be updated and the time t_{i+1} will not be a valid sample from the updated intensity. To overcome this difficulty, we design the following procedure, which to the best of our knowledge, is novel in the context of temporal point processes. Recall that the intensity function of the action events was

$$\lambda^*_\theta(t) = \exp(b_\lambda + \boldsymbol{V}_h \boldsymbol{h}_i) \exp(\omega_t (t - t_i)) \tag{A.1}$$

In other words, we write $\lambda^*_\theta(t) = c.e^{\omega_t(t-t_i)}$ and c changes due to an arrival of an event. So, we can state our problem as the following more general problem of sampling from a partially known intensity function:

$$\lambda(t) = \begin{cases} c_1 e^{-\omega(t-t_i)} & \text{if } t < s \\ c_2 e^{-\omega(t-t_i)} & \text{otherwise,} \end{cases} \tag{A.2}$$

where the parameters c_1 is known to us at time t_i but s, c_2 are revealed to us only at time s, *i.e.*, if our sampled time is greater than s. Due to this, we cannot sample from the above intensity using simple rejection sampling or the superposition property of Poisson processes, as previous work Tabibian et al. [2019a], Zarezade et al. [2017a]. Instead, at a high level, we solve the problem by first sampling a uniform random variable $u \sim U[0,1]$ and then using it to calculate $t_{i+1} = CDF_1^{-1}(u \,|\, c_1, t_i)$, where $CDF_1(t \,|\, c_1, t_i)$ denotes the cumulative distribution function of the next event time. Here, we are using inverse transform sampling under the assumption that the intensity function is defined completely using c_1 only. Then, we wait until the earlier of either t_{i+1}, when we *accept* the sample, or s, in which case the parameters c_2 are revealed to us. With the full knowledge of the intensity function, we can now *refine* our sample $t_{i+1} \leftarrow CDF_2^{-1}(t \,|\, c_1, t_i, c_2, s)$ re-using the same u that we had originally sampled.

To be able to perform the above procedure in an efficient manner, we should be able to express $CDF_1^{-1}(t \,|\, c_1, t_i)$ and $CDF_2^{-1}(t \,|\, c_1, t_i, c_2, s)$ analytically. Perhaps surprisingly, we can indeed express both functions analytically for our parametrized intensity function, given by Eq. A.2, *i.e.*,

$$
\begin{aligned}
CDF_1(t \,|\, c_1, t_i) &= \Pr\left[\text{An event happens before } t\right] \\
&= 1 - \Pr\left[\text{No event in } (t_i, t]\right] \\
&= 1 - \exp\left(-\int_{t_i}^{t} \lambda(\tau)\, d\tau\right) \\
&= 1 - \exp\left(-\int_{t_i}^{t} c_1 e^{-\omega(\tau - t_i)}\, d\tau\right) \\
&= 1 - \exp\left(\frac{c_1}{\omega}\left(e^{-\omega(t-t_i)} - 1\right)\right) \\
\implies CDF_1^{-1}(u \,|\, c_1, t_i) &= t_i - \frac{1}{\omega}\log\left(1 + \frac{\omega}{c_1}\log\left(1-u\right)\right) \qquad \text{(A.3)} \\
\text{Similarly, } CDF_2^{-1}(u \,|\, c_1, t_i, c_2, s) &= s - \frac{1}{\omega}\log\left(1 + \frac{\omega}{c_2}\log\left(\frac{1-u}{Q}\right)\right) \qquad \text{(A.4)} \\
\text{where } Q &= \exp\left(-\frac{c_1}{\omega}\left(1 - \exp\left(-\omega(s - t_i)\right)\right)\right).
\end{aligned}
$$

Notice that Eq. A.4 is the same as Eq. A.3, if our uniform sample had been $u' = 1 - \frac{1-u}{Q}$, and we had started the sampling process at time s instead of time t_i with parameters c_2, ω. Using this insight, we can easily generalize this sampling mechanism to account for an arbitrary number of feedback events occurring between two actions of the agent. Algorithm 4 summarizes our sampling algorithm, where ComputeC1 and ComputeC2 compute the current values of c_1 and c_2, respectively, WaitUntilNextFeedback(t) sets a flag e to True if a feedback event (s, z) happens before time t. Remarkably, given a cut-off time T, the algorithm only needs to sample $|\mathcal{A}_T|$ times from a uniform distribution and perform $O(|\mathcal{H}_T|)$ computations.

Finally, note that, in the above procedure, there is a possibility that the inverse CDF functions may not be completely defined on the domain $[0,1]$. This would mean that the agent's MTPP may go *extinct*, *i.e.*, there may be a finite probability of the agent not taking an action after time t_i at all. In such cases, we assume that the next action time is beyond our episode horizon T, but we will save the original u and will keep calculating the inverse CDF using it as, due to the non-linear dependence of the parameters on the history, the samples may become finite again.

A.4 Experimental details

We carried out all our experiments using TensorFlow 1.4.1 with DynamicRNN API and we implemented stochastic gradient descent (SGD) using the Adam optimizer, which achieved good performance in practice, as shown in Figure A.1. Therein, we had to specify eight hyperparameters: (i) N_b – the number of batches, (ii) N_e – the number of episodes in each batch, (iii) T – the time length of each episode, (iv) l_r– the learning

Algorithm 4 It returns the next action time

```
Input: Time of previous action $t'$, history $\mathcal{H}_{t'}$ up to $t'$, cut-off time $T$
Output: Next action time $t$
$c_1 \leftarrow$ ComputeC1($\mathcal{H}_{t'}$)
$t \leftarrow CDF_1^{-1}(u \,|\, c_1, t')$
while $t < T$ do
    $(e, s, z) \leftarrow$WaitUntilNextFeedback($t$)
    if $e ==$ True then
        $\mathcal{H}_{t'} \leftarrow \mathcal{H}_{t'} \cup \{(s, z)\}$
        $c_1 \leftarrow$ ComputeC1($\mathcal{H}_{t'}$), $c_2 \leftarrow$ ComputeC2($\mathcal{H}_{t'}$)
        $t \leftarrow CDF_2^{-1}(u \,|\, c_1, t', c_2, s)$
    else
        return t
        break
    end if
end while
return t
```

Application	N_b	N_e	T	l_r	D_i	D_h	q_l	q_m
Spaced repetition	5000	32	14 days	$\frac{0.02}{1+2i\cdot10^{-3}}$	8	8	10^{-2}	$5 \cdot 10^{-3}$
Smart broadcasting	1000	16	It varies across users	$\frac{10^{-2}}{1+i\cdot10^{-4}}$	8	8	0.33 (100)	–

Table A.1: Hyperparameter values used in the implementation of our method for smart broadcasting and spaced repetition. In smart broadcasting, $q_l = 0.33$ for top-1 inverse chronological ordering and $q_l = 100$ for average rank inverse chronological ordering.

rate, (v) D_i – the dimension of vectors $\boldsymbol{W}_{\bullet}, \boldsymbol{b}_{\bullet}$'s in the input layer, (vi) D_h – the dimension of the hidden state $\boldsymbol{h}_i$, (vii) q_l – the value of the regularizer coefficient for intensity function, (viii) q_m – the value of the regularizer coefficient for mark distribution. Note that, the dimensions of the other trainable parameters $\boldsymbol{W}_h, \boldsymbol{W}_1, .., \boldsymbol{W}_4$ and $\boldsymbol{b}_h$ in the hidden layer depend on D_i and $\boldsymbol{V}_\lambda$ and $\boldsymbol{V}_c^y$ in the output layer depend on D_h, which we selected using cross validation. The values for both applications—spaced repetition and smart broadcasting —are given in Table A.1.

We run the spaced repetition experiments using a Tesla K80 GPU on a machine with 32 cores and 500GB RAM. With this configuration, for episodes with up to ∼2000 events, the training process takes ∼5 seconds in average to run one iteration of SGD with batch size $N_e = 32$. We run the smart broadcasting experiments on 2 CPU cores of an Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz and 20GB RAM. With this configuration, for feeds sorted algorithmically and episodes with up to ∼250 events, the training process takes ∼30 seconds to run one iteration of SGD with batch size $N_e = 16$.

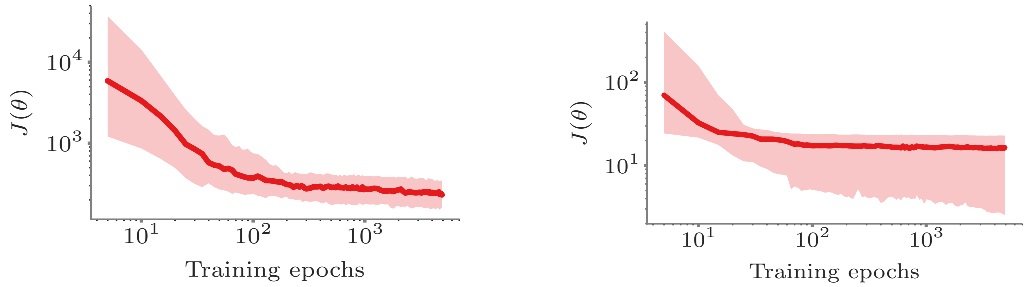
A.5 Student model

We use the student model proposed by Tabibian *et al.* Tabibian et al. [2019a], which is an improved version of the student model proposed by Settles *et al.* Settles and Meeder [2016]. To accurately predict the student's ability to recall an item, the model accounts for the item difficulty, the history of reviews (and recalls) by the student, and the time since the last review.

More formally, the probability $m_i(t)$ that an item i, which was last reviewed at time η, will be successfully recalled at time t is given by:

$$m_i(t) = e^{-n_i(t)\times(t-\eta)} \tag{A.5}$$

where $n_i(t)$ denotes the forgetting rate for the item i. The rate of forgetting an item depends on the inherent difficulty of the item, denoted by $n_i(0)$, but also on whether the user was able to recall the item successfully in the past or not. More specifically, the model has two additional parameters α and β, which determine by



(a) $J(\theta)$ for quadratic loss

(b) $J(\theta)$ for time spent at top

Figure A.1: The cost-to-go $J(\theta)$ calculated on the held-out test-set for different loss functions during training falls quickly with the number of epochs.

how much the forgetting rate ought to change if the student recall, or fails to recall, the item on a review at time t, *i.e.*,

$$n_i(t) = \begin{cases} (1-\alpha) \times n_i(t^-) & \text{if recalled} \\ (1+\beta) \times n_i(t^-) & \text{if forgotten} \end{cases} \tag{A.6}$$

In our work, the parameters α and β, as well as the initial item difficulty $n_i(0)$, are learned using historical learning data from Duolingo as in Tabibian *et al.* Tabibian et al. [2019a].

Note that we have picked this student model for its simplicity but relatively good predictive power, as shown by previous work. Several other student models have also been proposed in literature, ranging from exponential Ebbinghaus [1885] to more recent multi-scale context models (MCM) Pashler et al. [2009], which are biologically inspired and can explain a wider variety of learning phenomenon. Since our methodology is agnostic to the choice of student model, it would be very interesting to experiment with other student models.

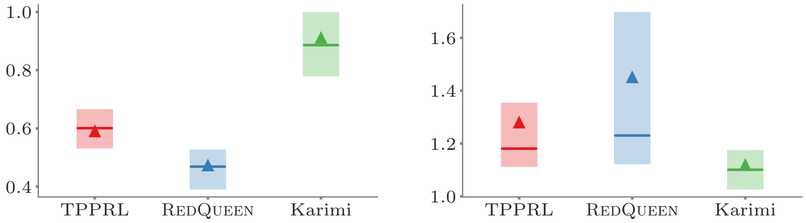
A.6 Feed sorting algorithm

We use a feed sorting algorithm inspired by the *in-case-you-missed-it* feature, which is now prevalent in a variety of social media sites, notably Twitter at the time of writing. Our sorting algorithm divides each user's feed in two sections: (i) a prioritized section at the top of the user's feed, where messages are sorted according to the *priority* of the user who posted the message, and (ii) a bulk section, where messages are sorted in reverse chronological order. In the above, each post stays for a fixed time τ in the prioritized section and then it moves to the inverse chronological section. Moreover, note that if the prioritized section contains several messages from the same user, they are sorted chronologically.

In our experiments, for each user's feed, we set the priority of the users she follows inversely proportional to her level of activity, as more active users will naturally appear on the feed while users with sporadic posting activity may need more promotion, we set the priority of the user under our control to be at the median priority among all users posting in the feed, and set τ to be approximately 10% of the prioritized lifetime of posts $\tau = 0.1T$, where T is the time length of each sequence.

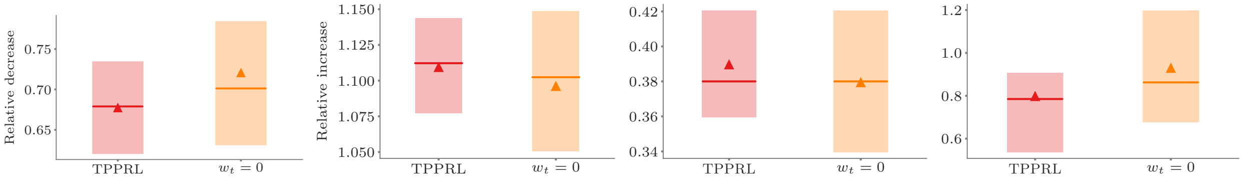
A.7 Experiments on feeds sorted in reverse chronological order

We follow the same experimental setup as in Section 4.1.3, however, feeds are sorted in reverse chronological order. Figure A.2 summarizes the results, where the number of messages posted by each method is the same and all rewards are normalized by the reward achieved by a baseline user who follows a uniform Poisson intensity. The results show that our method is able to achieve competitive results in comparison with



(a) Average rank (b) Time at top

Figure A.2: Performance of our policy gradient method against RedQueen Zarezade et al. [2017a] and Karimi's method Karimi et al. [2016] on feeds sorted in reverse chronological order. Panels (a) and (b) show the average rank and time at the top, where the solid horizontal line shows the median value across users, normalized with respect to the value achieved by a user who follows a uniform Poisson intensity, and the box limits correspond to the 25%-75% percentiles. For the average rank, lower is better and, for time at the top, higher is better. In both cases, the number of messages posted by each method is the same.



(a) Average rank (b) Time at top (c) Recall (d) Items' difficulty

Figure A.3: Comparing against piece-wise constant ($w_t = 0$) baseline. In all figures, the solid horizontal line shows the median value across users and the box limits correspond to the 25%-75% percentiles. Panels (a) and (b) show the average rank and time at the top for the smart broadcasting experiments, respectively. The values are normalized with respect to the value achieved by a user who follows a uniform Poisson intensity. For the average rank, lower is better and, for time at the top, higher is better. In both cases, the number of messages posted by each method is the same within a 10% tolerance. Panel (c) shows the empirical recall probability at test time and Panel (d) shows the distribution of the difficulty of items chosen by our method and the baseline version for the space repetition experiments. The total number of learning events (across all items) are within 5% of each other in the two settings.

RedQueen, which is an online algorithm specially designed to minimize the average rank in feeds sorted in reverse chronological order, and it outperforms Karimi's method, which is an offline algorithm specially designed to maximize the time at the top in feeds sorted in reverse chronological order.

A.8 Baseline with $w_t = 0$

We also explored how our algorithm performs when we force the w_t parameter to be zero, *i.e.*, we force the policy to be piece-wise constant between feedback and action events. To this end, we retrained the neural networks by doing a parameter sweep over q_l (and q_m for the spaced repetition experiments) and picked those values which arrived to roughly the same number of events as produced by the policy learned by the network where we do not constraint $w_t = 0$.

The resulting baseline is shown in Figure A.3 for both the smart broadcasting (Figures A.3a and A.3b) and spaced repetition experiments (Figures A.3c and A.3d). We see that forcing the policy to be piecewise constant degrades performance and increases the variance in both settings, as expected. In the smart broadcasting experiments, the mean (median) relative decrease in average rank is 33% (33%) for our method TPPRL, while it is 28% (30%) for the $w_t = 0$ baseline. Similarly, the increase in mean time spent at the top is about 11% for our method (TPPRL), while it is 9% for the $w_t = 0$ baseline. In the spaced repetition experiment, we see that the mean recall falls from 38.9% to 37.9%. The difference in policy learned is especially notable in Figure A.3d where we see that the agent, when constrained to $w_t = 0$, learns to spread its attempts over a wider set of items, which have higher difficulty than the items selected by the unconstrained policy.

Appendix B

Appendix: Learning to Schedule

B.1 Proof of Lemma 1

Lemma 1 (Adapted from [Azar et al., 2018])**.** *For any given* $\varepsilon > 0$*, let* $\boldsymbol{\rho} \in \Delta_R$ *be a policy which the crawler adopts at time* t_0 *and let the initial state of the cache be* $\boldsymbol{S}(t_0) = [s_1, s_2, \ldots, s_m] \in \{0,1\}^m$*, where* $s_i = \textsc{Fresh}(i, t_0)$*. Then if* $t - t_0 \geq \frac{1}{\xi_{\min}} \log\left(\frac{2\sum_{j=1}^m \zeta_i}{\varepsilon}\right)$*, then* $\sum_{i=1}^m \frac{\zeta_i \xi_i}{\xi_i + \rho_i} < \sum_{i=1}^m F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi}) < \sum_{i=1}^m \frac{\zeta_i \xi_i}{\xi_i + \rho_i} + \varepsilon$*.*

Proof. Let $E^{\emptyset}_i[t_0, t]$ denote an event that neither a change nor a refresh has happened for webpage i in time interval $[t_0, t]$. Note that under event $E^{\emptyset}_i[t_0, t]$, we have $\textsc{Fresh}(i, t) = s_i$. Otherwise (*i.e.*, under event $\neg E^{\emptyset}_i[t_0, t]$), as we have assumed that the change and the refresh processes are independent Poisson processes for all webpages, the probability that the last event which happened for webpage i between t_0 and t was an update event is $\frac{\rho_i}{\rho_i + \xi_i}$. Hence, we can write the differential utility function as:

$$
\begin{aligned}
&F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi}) = \zeta_i \mathbb{P}_{\boldsymbol{\rho}}(\textsc{Fresh}(i, t)) \\
&= \frac{\zeta_i \rho_i}{\rho_i + \xi_i} \mathbb{P}(\neg E^{\emptyset}_i[t_0, t]) + \zeta_i s_i \mathbb{P}(E^{\emptyset}_i[t_0, t]) \\
&= \frac{\zeta_i \rho_i}{\rho_i + \xi_i} \left(1 - e^{-(\rho_i + \xi_i)(t - t_0)}\right) + \zeta_i s_i e^{-(\rho_i + \xi_i)(t - t_0)} \\
&= \frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \left(\frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \zeta_i s_i\right) e^{-(\rho_i + \xi_i)(t - t_0)} \\
&\implies \sum_{i=1}^m F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi}) \\
&= \sum_{i=1}^m \frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \sum_{i=1}^m \left(\frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \zeta_i s_i\right) e^{-(\rho_i + \xi_i)(t - t_0)} \qquad \text{(B.1)}
\end{aligned}
$$

This proves the first part of the inequality that $\sum_{i=1}^m \frac{\zeta_i \xi_i}{\xi_i + \rho_i} < \sum_{i=1}^m F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi})$.

Now substituting $t - t_0 = \frac{1}{\xi_{\min}} \log\left(\frac{2\sum_{j=1}^{m} \zeta_j}{\varepsilon}\right)$ into (B.1):

$$
\begin{aligned}
&\sum_{i=1}^{m} F^{(i)}_{[t_0,t]}(\boldsymbol{\rho}; \boldsymbol{\xi}) = \\
&\quad \sum_{i=1}^{m} \frac{\zeta_i \rho_i}{\rho_i + \xi_i} \\
&\qquad + \sum_{i=1}^{m} \left(\frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \zeta_i s_i\right) e^{-(\rho_i+\xi_i)\frac{1}{\xi_{\min}} \log\left(\frac{2\sum_{j=1}^{m} \zeta_j}{\varepsilon}\right)} \\
&\leq \sum_{i=1}^{m} \frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \sum_{i=1}^{m} \zeta_i \left(\frac{\rho_i}{\rho_i + \xi_i} + s_i\right) \left(\frac{\varepsilon}{2\sum_{j=1}^{m} \zeta_j}\right) \\
&\leq \sum_{i=1}^{m} \frac{\zeta_i \rho_i}{\rho_i + \xi_i} + \varepsilon
\end{aligned}
$$

where we have used $\frac{\rho_i + \xi_i}{\xi_{\min}} \geq 1$ in the first inequality and $\frac{\rho_i}{\rho_i + \xi_i} + s_i \leq 2$ in the second inequality. ☐

B.2 Proof of Lemma 3

Lemma 3. *Under the condition of Lemma 2, for any* $\delta \in (0, 1)$*, and* N *observations it holds that*

$$
\mathbb{P}\left(|\widehat{\xi} - \xi| \geq \left(\frac{1}{N} \sum_{n=1}^{N} w_n e^{-\xi_{\max} w_n} \right)^{-1} \sqrt{\frac{\log \frac{2}{\delta}}{2N}} \right) \leq \delta
$$

where $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$ *and* $\widetilde{\xi}$ *is obtained by solving* (4.17)*.*

Proof. Recall that $\widehat{p} = \frac{1}{N} \sum_{n=1}^{N} (1 - o_n)$ is the empirical frequency of no-event counts, and denote $\mathbb{E}[\widehat{p}]$ by p. In this notation, we have:

$$
\widehat{p} = \frac{1}{N} \sum_{n=1}^{N} e^{-\widetilde{\xi} w_n}, \qquad \text{and} \quad p = \frac{1}{N} \sum_{n=1}^{N} e^{-\xi w_n},
$$

where $\widetilde{\xi}$ is monotonically decreasing in $\widehat{p}$ (and similarly for ξ as a function of p).

First, assume that $\widehat{p} \leq p$, which implies $\widetilde{\xi} \geq \xi$ by the monotonicity property mentioned above, and by the property of clipping (and the fact that $\xi \in [\xi_{\min}, \xi_{\max}]$) we also have $\widetilde{\xi} \geq \widehat{\xi} \geq \xi$. By the convexity of the exponential function:

$$
e^{-\xi w_n} \;\geq\; e^{-\widehat{\xi} w_n} - w_n e^{-w_n \widehat{\xi}} (\xi - \widehat{\xi}) \;\geq\; e^{-\widetilde{\xi} w_n} - w_n e^{-w_n \widehat{\xi}} (\xi - \widehat{\xi}),
$$

which implies after summing over n:

$$
p - \widehat{p} \;\geq\; (\widehat{\xi} - \xi) \frac{1}{N} \sum_{n=1}^{N} w_n e^{-w_n \widehat{\xi}} \;\geq\; (\widehat{\xi} - \xi) \frac{1}{N} \sum_{n=1}^{N} w_n e^{-w_n \xi_{\max}}. \tag{B.2}
$$

Similarly for $p \leq \widehat{p}$ we have $\widetilde{\xi} \leq \widehat{\xi} \leq \xi$ and therefore:

$$
e^{-\widetilde{\xi} w_n} \;\geq\; e^{-\widehat{\xi} w_n} \;\geq\; e^{-\xi w_n} - w_n e^{-w_n \xi} (\widehat{\xi} - \xi),
$$

which implies:

$$\widehat{p}-p \;\geq\; (\xi-\widehat{\xi})\frac{1}{N}\sum_{n=1}^{N} w_n e^{-w_n\xi} \;\geq\; (\xi-\widehat{\xi})\frac{1}{N}\sum_{n=1}^{N} w_n e^{-w_n\xi_{\max}} \tag{B.3}$$

By combining (B.2) and (B.3), we get:

$$|\widehat{p}-p| \geq |\xi-\widehat{\xi}|\frac{1}{N}\sum_{n=1}^{n} w_n e^{-w_n\xi_{\max}} \tag{B.4}$$

For $\widehat{p}$ being a frequency of counts, Hoeffding's inequality for independent Bernoulli variables implies that for $\delta \in (0,1)$:

$$\mathbb{P}\left(|\widehat{p}-p| < \underbrace{\sqrt{\frac{\log {}^{2}/_{\delta}}{2N}}}_{\varepsilon}\right) \geq 1-\delta$$

Hence, with probability at least $1-\delta$, we have $|\widehat{p}-p| \leq \varepsilon$ and combining it with (B.4), with probability $1-\delta$:

$$|\xi-\widehat{\xi}| \leq \left(\frac{1}{N}\sum_{n=1}^{N} w_n e^{-w_n\xi_{\max}}\right)^{-1}\varepsilon,$$

which finishes the proof. □

B.3 Proof of Lemma 2

Lemma 2. *Let* $\{y_0 := 0\} \cup (y_n)_{n=1}^{\infty}$ *be a sequence of times, such that* $\forall n.\, y_n > 0$*, at which observations* $(o_n)_{n=1}^{\infty} \in \{0,1\}^{\mathbb{N}}$ *are made of a Poisson process with rate* ξ*, such that* $o_n := 1$ *iff there was an event of the process in* $(y_{n-1}, y_n]$*, define* $w_n = y_n - y_{n-1}$*,* $I = \{n : w_n < 1\}$ *and* $J = \{n : w_n \geq 1\}$*. Then:*

1. *If* $\sum_{n\in I} w_n < \infty$ *and* $\sum_{n\in J} e^{-\xi w_n} < \infty$*, then any statistic for estimating* ξ *has non-vanishing bias.*
2. *If* $\sum_{n\in I} w_n = \infty$*, then there exist disjoint subsets* $I_1, I_2, \ldots$ *of* I *such that* $\left(\sum_{n\in I_k} w_n\right)_{k=1}^{\infty}$ *is monotone and* $\sum_{n\in I_k} w_n \in (1,2)$ *for* $k = 1, 2, \ldots$ *For any such sequence* $\mathcal{I} = (I_k)_{k=1}^{\infty}$*, the mapping* $c_{\mathcal{I}}(\xi) = \lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} \exp\left(-\xi \sum_{n\in I_k} w_n\right)$ *is strictly monotone and*

$$\left[\frac{1}{K}\sum_{k=1}^{K} \mathbb{I}\left(\sum_{n\in I_k} o_n \geq 1\right)\right] \xrightarrow{a.s.} 1 - c_{\mathcal{I}}(\xi).$$

3. *If* $\sum_{n\in J} e^{-w_n\xi} = \infty$ *then, there exists a sequence* $\mathcal{J} = (J_k)_{k=1}^{\infty}$ *of disjoint subsets of* J *such that* $\left(\sum_{n\in J_k} e^{-w_n\xi}\right)_{k=1}^{\infty}$ *is monotone and* $\sum_{n\in J_k} e^{-w_n\xi} \in [1/e, 2/e)$ *for* $k = 1, 2, \ldots$ *For any such* $\mathcal{J}$*, the mapping* $c_{\mathcal{J}}(\xi) = \lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K}\prod_{n\in J_k}\left(1-e^{-\xi w_n}\right)\right]$ *is strictly monotone and*

$$\lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K} \mathbb{I}\left(o_n \geq 1,\ \forall n \in J_k\right)\right] \xrightarrow{a.s.} c_{\mathcal{J}}(\xi),$$

Proof. Let $\alpha = -e^{\xi}\log\left(1-e^{-\xi}\right)$ for which it holds that

$$1 - e^{-\xi} = e^{-\alpha e^{-\xi}}. \tag{B.5}$$

Using the convexity of the exponential function for any $x_0 \in \mathbb{R}$ and any $\beta \in [0,1]$ we have:

$$e^{\beta x_0} \;=\; e^{\beta x_0 + (1-\beta)\cdot 0} \;\leq\; \beta e^{x_0} + (1-\beta).$$

Take any $x \in [0, e^{-\xi}]$ and apply the above to $x_0 = -\alpha e^{-\xi}$ and $\beta = xe^{\xi} \in [0,1]$ to get:

$$\begin{aligned} e^{-\alpha x} \;&\leq\; xe^{\xi} e^{-\alpha e^{-\xi}} + (1 - xe^{\xi}) \\ &=\; xe^{\xi}(1 - e^{-\xi}) + (1 - xe^{\xi}) \;=\; 1 - x, \end{aligned} \tag{B.6}$$

where we used (B.5) in the first equality.

Consider the probability p that $o_n = 0$ for all $i \in I$ and $o_n = 1$ for all $i \in J$:

$$\begin{aligned} p := \prod_{n \in \mathbb{N}} \mathbb{P}\left[o_n = \begin{cases} 1 & \text{if } n \in J \\ 0 & \text{if } n \in I \end{cases} \right] \\ = \left(\prod_{n \in I} e^{-w_n \xi} \right) \times \left(\prod_{n \in J} (1 - e^{-w_n \xi}) \right) \\ > \exp\left(-\xi \sum_{n \in I} w_n \right) \times \exp\left(-\alpha \sum_{n \in J} e^{-w_n \xi} \right) \quad , \end{aligned}$$

where the last inequality follows by (B.6). This probability is bounded away from 0 if $\sum_{n \in I} w_i < \infty$ and $\sum_{n \in J} e^{-w_n \xi} < \infty$.

Now consider another Poisson process with rate $\xi' < \xi$ which we also observe at times $(y_n)_{n=1}^{\infty}$ to produce observations $(o'_n)_{n=1}^{\infty}$. Then define, analogous to p above:

$$p' := \left(\prod_{n \in I} e^{-w_n \xi'} \right) \times \left(\prod_{n \in J} \left[1 - e^{-w_n \xi'} \right] \right),$$

which is also positive (since $\xi' < \xi$). Thus, the two series $(o_n)_{n=1}^{\infty}$ and $(o'_n)_{n=1}^{\infty}$ are identical with probability at least $p'p > 0$. This proves the first claim as there is no way to distinguish between ξ and ξ' when $(o_n)_{n=1}^{\infty} = (o'_n)_{n=1}^{\infty}$.

Regarding the second claim, the existence of the sequence $I_1, I_2, \ldots$ satisfying the constraints follows from the monotone subsequence theorem (*i.e.*, constructing $I'_1, I'_2, \ldots$ that satisfies all the constraints except for the one on monotonicity, the theorem guarantees the existence of a subset that also satisfies the constraint on monotonicity). Note now that, for all k:

$$\begin{aligned} \mathbb{E}\left[\mathbb{I}\left(\sum_{n \in I_k} o_n \geq 1 \right) \right] &= 1 - \mathbb{P}\left(o_n = 0, \forall n \in I_k \right) \\ &= 1 - e^{-\xi \sum_{n \in I_k} w_n}, \end{aligned} \tag{B.7}$$

thus, due to the law of large numbers,

$$\frac{1}{K} \sum_{k=1}^{K} \left[\mathbb{I}\left(\sum_{n \in I_k} o_n \geq 1 \right) - \left(1 - e^{-\xi \sum_{n \in I_k} w_n} \right) \right] \xrightarrow{a.s.} 0.$$

Also notice that $\lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \left[\left(1 - e^{-\xi \sum_{n \in I_k} w_n} \right) \right]$ exists due to the monotonicity constraint and the fact that ξ and each w_n are positive. These together imply

$$\begin{aligned}\frac{1}{K}\sum_{k=1}^{K}\left[\mathbb{I}\left(\sum_{n\in I_k} o_n \geq 1\right)\right] &\xrightarrow{a.s.} \\ &\lim_{K\to\infty}\frac{1}{K}\sum_{k=1}^{K}\left[1-e^{-\xi\sum_{n\in I_k} w_n}\right] \\ &= 1-\lim_{K\to\infty}\frac{\sum_{k=1}^{K} e^{-\xi\sum_{n\in I_k} w_n}}{K}. \end{aligned} \tag{B.8}$$

Due to the constraint $\sum_{n\in I_k} w_n < 2$,

$$\lim_{K\to\infty}\frac{\sum_{k=1}^{K} e^{-\xi\sum_{n\in I_k} w_n}}{K} > 0. \tag{B.9}$$

Finally, the constraint $\sum_{n\in I_k} w_n > 1$ implies that, for any $\xi > \xi' > 0$,

$$\xi\sum_{n\in I_k} w_n - \xi'\sum_{n\in I_k} w_n = (\xi-\xi')\sum_{n\in I_k} w_n > \xi-\xi',$$

further implying

$$\frac{\sum_{k=1}^{K} e^{-\xi'\sum_{n\in I_k} w_n}}{K} > e^{\xi-\xi'}\frac{\sum_{k=1}^{K} e^{-\xi\sum_{n\in I_k} w_n}}{K}. \tag{B.10}$$

Strict monotonicity, and thus also uniqueness, now follows from (B.10) and (B.9).

The last claim follows similarly. □

B.4 Proof of Lemma 4

Lemma 4. *For the expected utility $F(\boldsymbol{\rho};\boldsymbol{\xi})$ defined in* (4.15)*, let $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho};\boldsymbol{\xi})$, $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho};\widehat{\boldsymbol{\xi}})$ and define the suboptimality of $\widehat{\boldsymbol{\rho}}$ as $\operatorname{err}(\widehat{\boldsymbol{\rho}}) := F(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}};\boldsymbol{\xi})$. Then $\operatorname{err}(\widehat{\boldsymbol{\rho}})$ can be bounded by:*

$$\operatorname{err}(\widehat{\boldsymbol{\rho}}) \leq \sum_i \frac{1}{\widehat{\xi}_i\min\{\widehat{\xi}_i,\xi_i\}}\zeta_i(\widehat{\xi}_i-\xi_i)^2.$$

Proof. Since $\boldsymbol{\rho}^\star$ minimizes $F(\boldsymbol{\rho},\boldsymbol{\xi})$, it follows from the first-order optimality condition that:

$$\begin{aligned}&\left(\nabla_{\boldsymbol{\rho}^\star} F(\boldsymbol{\rho}^\star,\boldsymbol{\xi})\right)^\top(\widehat{\boldsymbol{\rho}}-\boldsymbol{\rho}^\star) \leq 0 \\ \Longrightarrow &\sum_i \frac{\zeta_i\xi_i}{(\xi_i+\rho_i^\star)^2}(\widehat{\rho}_i-\rho_i^\star) \leq 0.\end{aligned}$$

We thus lower-bound the suboptimality by:

$$\begin{aligned}\operatorname{err}(\widehat{\boldsymbol{\rho}}) &= F(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}};\boldsymbol{\xi}) \\ &\geq F(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}};\boldsymbol{\xi}) + \left(\nabla_{\boldsymbol{\rho}^\star} F(\boldsymbol{\rho}^\star,\boldsymbol{\xi})\right)^\top(\widehat{\boldsymbol{\rho}}-\boldsymbol{\rho}^\star) \\ &= \sum_i\left(\frac{\zeta_i\rho_i^\star}{\xi_i+\rho_i^\star} - \frac{\zeta_i\widehat{\rho}_i}{\xi_i+\widehat{\rho}_i} + \frac{\zeta_i\xi_i}{(\xi_i+\rho_i^\star)^2}(\widehat{\rho}_i-\rho_i^\star)\right) \\ &= \sum_i\zeta_i\left(\frac{\xi_i(\rho_i^\star-\widehat{\rho}_i)}{(\xi_i+\rho_i^\star)(\xi_i+\widehat{\rho}_i)} - \frac{\xi_i(\rho_i^\star-\widehat{\rho}_i)}{(\xi_i+\rho_i^\star)^2}\right) \\ &= \sum_i\frac{\zeta_i\xi_i(\rho_i^\star-\widehat{\rho}_i)^2}{(\xi_i+\rho_i^\star)^2(\xi_i+\widehat{\rho}_i)}. \end{aligned} \tag{B.11}$$

On the other hand, because $\widehat{\boldsymbol{\rho}}$ minimizes $F(\boldsymbol{\rho}, \widehat{\boldsymbol{\xi}})$, we can upper-bound the suboptimality by:

$$\begin{aligned}
\operatorname{err}(\widehat{\boldsymbol{\rho}}) = F(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - F(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) + \underbrace{F(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) - F(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}})}_{\leq 0} \\
+ F(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}}) - F(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) \\
\leq F(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - F(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) \; + \; F(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}}) - F(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}).
\end{aligned} \tag{B.12}$$

We explicitly calculate both differences on the right hand side:

$$\begin{aligned}
F(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - F(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) &= \sum_i \zeta_i \left(\frac{\rho_i^\star}{\xi_i + \rho_i^\star} - \frac{\rho_i^\star}{\widehat{\xi}_i + \rho_i^\star} \right) \\
&= \sum_i \frac{\zeta_i \rho_i^\star (\widehat{\xi}_i - \xi_i)}{(\xi_i + \rho_i^\star)(\widehat{\xi}_i + \rho_i^\star)}, \\
F(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}}) - F(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) &= \sum_i \zeta_i \left(\frac{\widehat{\rho}_i}{\widehat{\xi}_i + \widehat{\rho}_i} - \frac{\widehat{\rho}_i}{\xi_i + \widehat{\rho}_i} \right) \\
&= \sum_i \frac{\zeta_i \widehat{\rho}_i (\xi_i - \widehat{\xi}_i)}{(\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i + \widehat{\rho}_i)},
\end{aligned}$$

so that (B.12) becomes

$$\begin{aligned}
&\operatorname{err}(\widehat{\boldsymbol{\rho}}) \\
&\leq \sum_i \zeta_i (\widehat{\xi}_i - \xi_i) \left(\frac{\rho_i^\star}{(\xi_i + \rho_i^\star)(\widehat{\xi}_i + \rho_i^\star)} - \frac{\widehat{\rho}_i}{(\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i + \widehat{\rho}_i)} \right) \\
&= \sum_i \frac{\zeta_i (\xi_i \widehat{\xi}_i - \rho_i^\star \widehat{\rho}_i)(\widehat{\xi}_i - \xi_i)(\rho_i^\star - \widehat{\rho}_i)}{(\xi_i + \rho_i^\star)(\widehat{\xi}_i + \rho_i^\star)(\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i + \widehat{\rho}_i)} \\
&\leq \sum_i \frac{\zeta_i |\xi_i \widehat{\xi}_i - \rho_i^\star \widehat{\rho}_i| |\widehat{\xi}_i - \xi_i| |\rho_i^\star - \widehat{\rho}_i|}{(\xi_i + \rho_i^\star)(\widehat{\xi}_i + \rho_i^\star)(\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i + \widehat{\rho}_i)}.
\end{aligned}$$

Further bound:

$$|\xi_i \widehat{\xi}_i - \rho_i^\star \widehat{\rho}_i| \leq \xi_i \widehat{\xi}_i + \rho_i^\star \widehat{\rho}_i \leq (\widehat{\xi}_i + \rho_i^\star)(\xi_i + \widehat{\rho}_i),$$

to obtain:

$$\begin{aligned}
\operatorname{err}(\widehat{\boldsymbol{\rho}}) &\leq \sum_i \frac{\zeta_i |\widehat{\xi}_i - \xi_i| |\rho_i^\star - \widehat{\rho}_i|}{(\xi_i + \rho_i^\star)(\widehat{\xi}_i + \widehat{\rho}_i)} \\
&= \sum_i \sqrt{\frac{\zeta_i (\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i - \xi_i)^2}{\xi_i (\widehat{\xi}_i + \widehat{\rho}_i)^2}} \sqrt{\frac{\zeta_i \xi_i (\rho_i^\star - \widehat{\rho}_i)^2}{(\xi_i + \rho_i^\star)^2 (\xi_i + \widehat{\rho}_i)}} \\
&\leq \sqrt{\sum_i \frac{\zeta_i (\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i - \xi_i)^2}{\xi_i (\widehat{\xi}_i + \widehat{\rho}_i)^2}} \sqrt{\sum_i \frac{\zeta_i \xi_i (\rho_i^\star - \widehat{\rho}_i)^2}{(\xi_i + \rho_i^\star)^2 (\xi_i + \widehat{\rho}_i)}} \\
&\leq \sqrt{\sum_i \frac{\zeta_i (\xi_i + \widehat{\rho}_i)(\widehat{\xi}_i - \xi_i)^2}{\xi_i (\widehat{\xi}_i + \widehat{\rho}_i)^2}} \sqrt{\operatorname{err}(\widehat{\boldsymbol{\rho}})},
\end{aligned}$$

where in the last but one inequality we used Cauchy-Schwarz inequality $\sum_i x_i y_i \leq \sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}$, while in the last inequality we used (B.11). Solving for $\mathrm{err}(\widehat{\boldsymbol{\rho}})$ gives:

$$\begin{aligned}\mathrm{err}(\widehat{\boldsymbol{\rho}}) &\leq \sum_i \frac{\zeta_i(\xi_i+\widehat{\rho}_i)(\widehat{\xi}_i-\xi_i)^2}{\xi_i(\widehat{\xi}_i+\widehat{\rho}_i)^2} \\ &\leq \sum_i \frac{1}{\widehat{\xi}_i \min\{\widehat{\xi}_i, \xi_i\}} \zeta_i(\widehat{\xi}_i-\xi_i)^2.\end{aligned}$$

The last inequality follows from: if $\xi_i \leq \widehat{\xi}_i$, then $\xi_i + \widehat{\rho}_i \leq \widehat{\xi}_i + \widehat{\rho}_i$ and:

$$\frac{\xi_i+\widehat{\rho}_i}{\xi_i(\widehat{\xi}_i+\widehat{\rho}_i)^2} \leq \frac{\widehat{\xi}_i+\widehat{\rho}_i}{\xi_i(\widehat{\xi}_i+\widehat{\rho}_i)^2} \leq \frac{1}{\xi_i(\widehat{\xi}_i+\widehat{\rho}_i)} \leq \frac{1}{\xi\widehat{\xi}_i},$$

whereas if $\xi_i \geq \widehat{\xi}_i$ then

$$\frac{\xi_i+\widehat{\rho}_i}{\xi_i(\widehat{\xi}_i+\widehat{\rho}_i)^2} \leq \frac{\xi_i+\widehat{\rho}_i}{\xi_i\widehat{\xi}_i(\widehat{\xi}_i+\widehat{\rho}_i)} \leq \frac{\xi_i}{\xi_i\widehat{\xi}_i^2} = \frac{1}{\widehat{\xi}_i^2},$$

and the last inequality follows from the fact that the function $f(x) = \frac{a+x}{b+x}$ with $a \geq b$ is maximized at $x = 0$. ☐

B.5 Sensitivity of Harmonic Staleness to Accuracy of Parameter Estimates

In this section, we consider the Harmonic staleness objective suggested in Kolobov et al. [2019a]. We will show that a result similar to Lemma 4 can be arrived at for that objective as well using similar techniques.

To that end, for this section define (from Problem 1 in [Kolobov et al., 2019a]):

$$H(\boldsymbol{\rho}, \boldsymbol{\xi}) = \sum_{i=1}^{m} \zeta_i \log \frac{\rho_i}{\rho_i + \xi_i}. \tag{B.13}$$

Lemma 6. *For the expected utility* $H(\boldsymbol{\rho}; \boldsymbol{\xi})$ *defined in* (B.13)*, let* $\boldsymbol{\rho}^\star = \mathrm{argmax}_{\boldsymbol{\rho}} H(\boldsymbol{\rho}; \boldsymbol{\xi})$, $\widehat{\boldsymbol{\rho}} = \mathrm{argmax}_{\boldsymbol{\rho}} H(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$ *and define the suboptimality of* $\widehat{\boldsymbol{\rho}}$ *as* $\mathrm{err}(\widehat{\boldsymbol{\rho}}) := H(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - H(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi})$*. Then* $\mathrm{err}(\widehat{\boldsymbol{\rho}})$ *can be bounded by:*

$$\mathrm{err}(\widehat{\boldsymbol{\rho}}) \leq \frac{(\xi_{\min} + R)R^2}{\xi_{\min}^5} \sum_{i=1}^{m} \zeta_i(\xi_i - \widehat{\xi}_i)^2.$$

Proof. It is easy to show by directly computing second derivatives that $H(\boldsymbol{\rho}, \boldsymbol{\xi})$ is concave in $\boldsymbol{\rho}$ and convex in $\boldsymbol{\xi}$.

As before, let $\boldsymbol{\rho}^\star = \mathrm{argmax}_{\boldsymbol{\rho} \in \Delta_R} H(\boldsymbol{\rho}, \boldsymbol{\xi})$ and $\widehat{\boldsymbol{\rho}} = \mathrm{argmax}_{\boldsymbol{\rho} \in \Delta_R} H(\boldsymbol{\rho}, \widehat{\boldsymbol{\xi}})$. By the optimality conditions, $\nabla_{\boldsymbol{\rho}} H(\boldsymbol{\rho}^\star, \boldsymbol{\xi})^\top (\boldsymbol{\rho} - \boldsymbol{\rho}^\star) \leq 0$ for all $\boldsymbol{\rho} \in \Delta_R$. We also bound the second derivative:

$$\begin{aligned}\frac{\partial^2 H(\boldsymbol{\rho}, \boldsymbol{\xi})}{\partial \rho_i \partial \rho_j} &= \delta_{ij}\zeta_i \left(\frac{1}{(\rho_i + \xi_i)^2} - \frac{1}{\rho_i^2} \right) \\ &\leq -\delta_{ij}\zeta_i \frac{\rho_i\xi_i + \xi_i^2}{(\xi_i + \rho_i)^2 \rho_i^2} \\ &= -\delta_{ij}\zeta_i \frac{\xi_i}{(\xi_i + \rho_i)\rho_i^2} \\ &\leq -\delta_{ij}\zeta_i \underbrace{\frac{\xi_{\min}}{(\xi_{\min} + R)R^2}}_{C}\end{aligned}$$

where in the last inequality we used the fact that $\frac{\xi_i}{\xi_i+R}$ is increasing in ξ_i. Taylor expanding $H(\boldsymbol{\rho}, \boldsymbol{\xi})$ around $\boldsymbol{\rho}^\star$ gives for some $\bar{\boldsymbol{\rho}}$ on the interval between $\boldsymbol{\rho}$ and $\boldsymbol{\rho}^\star$:

$$\begin{aligned} H(\boldsymbol{\rho}, \boldsymbol{\xi}) &= H(\boldsymbol{\rho}^\star, \boldsymbol{\xi}) + \underbrace{\nabla_{\boldsymbol{\rho}} H(\boldsymbol{\rho}^\star, \boldsymbol{\xi})^\top (\boldsymbol{\rho} - \boldsymbol{\rho}^\star)}_{\leq 0} \\ &\quad + \frac{1}{2} (\boldsymbol{\rho} - \boldsymbol{\rho}^\star)^\top \nabla^2_{\boldsymbol{\rho}} H(\bar{\boldsymbol{\rho}}, \boldsymbol{\xi}) (\boldsymbol{\rho} - \boldsymbol{\rho}^\star) \\ &\leq H(\boldsymbol{\rho}^\star, \boldsymbol{\xi}) - \frac{C}{2} \sum_{i=1}^m \zeta_i (\rho_i - \rho_i^\star)^2. \end{aligned}$$

Taking $\boldsymbol{\rho} = \widehat{\boldsymbol{\rho}}$, and rearranging, results in:

$$\mathrm{err}(\widehat{\boldsymbol{\rho}}) = H(\boldsymbol{\rho}^\star, \boldsymbol{\xi}) - H(\widehat{\boldsymbol{\rho}}, \boldsymbol{\xi}) \geq \frac{C}{2} \sum_{i=1}^m \zeta_i (\rho_i - \rho_i^\star)^2.$$

For the lower bound, note that:

$$\begin{aligned} \mathrm{err}(\widehat{\boldsymbol{\rho}}) &= H(\boldsymbol{\rho}^\star, \boldsymbol{\xi}) - H(\widehat{\boldsymbol{\rho}}, \boldsymbol{\xi}) \\ &= H(\boldsymbol{\rho}^\star, \boldsymbol{\xi}) - H(\boldsymbol{\rho}^\star, \widehat{\boldsymbol{\xi}}) + \underbrace{H(\boldsymbol{\rho}^\star, \widehat{\boldsymbol{\xi}}) - H(\widehat{\boldsymbol{\rho}}, \widehat{\boldsymbol{\xi}})}_{\leq 0} \\ &\qquad + H(\widehat{\boldsymbol{\rho}}, \widehat{\boldsymbol{\xi}}) - H(\widehat{\boldsymbol{\rho}}, \boldsymbol{\xi}) \\ &\leq \sum_{i=1}^m \zeta_i \log \frac{(\rho_i^\star + \widehat{\xi}_i)(\widehat{\rho}_i + \xi_i)}{(\rho_i^\star + \xi_i)(\widehat{\rho}_i + \widehat{\xi}_i)} \\ &= \sum_{i=1}^m \zeta_i \log \left(1 + \frac{(\rho_i^\star - \widehat{\rho}_i)(\xi_i - \widehat{\xi}_i)}{(\rho_i^\star + \xi_i)(\widehat{\rho}_i + \widehat{\xi}_i)} \right) \\ &\leq \sum_{i=1}^m \zeta_i \frac{(\rho_i^\star - \widehat{\rho}_i)(\xi_i - \widehat{\xi}_i)}{(\rho_i^\star + \xi_i)(\widehat{\rho}_i + \widehat{\xi}_i)} \qquad \text{(B.14)} \\ &\leq \sum_{i=1}^m \zeta_i \frac{|\rho_i^\star - \widehat{\rho}_i||\xi_i - \widehat{\xi}_i|}{(\rho_i^\star + \xi_i)(\widehat{\rho}_i + \widehat{\xi}_i)} \\ &\leq \frac{1}{\xi^2_{\min}} \sum_{i=1}^m \zeta_i |\rho_i^\star - \widehat{\rho}_i||\xi_i - \widehat{\xi}_i| \\ &\leq \frac{1}{\xi^2_{\min}} \sqrt{\sum_{i=1}^m \zeta_i (\rho_i^\star - \widehat{\rho}_i)^2} \sqrt{\sum_{i=1}^m \zeta_i (\xi_i - \widehat{\xi}_i)^2} \qquad \text{(B.15)} \\ &\leq \frac{1}{\xi^2_{\min}} \sqrt{\frac{2}{C}} \sqrt{\mathrm{err}(\widehat{\boldsymbol{\rho}})} \sqrt{\sum_{i=1}^m \zeta_i (\xi_i - \widehat{\xi}_i)^2}, \end{aligned}$$

where in (B.14) we used $\log(1+x) \leq x$, while in (B.15) we used the Cauchy-Schwarz inequality $\sum_i x_i y_i \leq \sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}$ with $x_i = \sqrt{\zeta_i}|\rho_i^\star - \widehat{\rho}_i|$ and $y_i = \sqrt{\zeta_i}|\xi_i - \widehat{\xi}_i|$.

Solving for $\mathrm{err}(\widehat{\boldsymbol{\rho}})$ gives:

$$\begin{aligned} \mathrm{err}(\widehat{\boldsymbol{\rho}}) &\leq \frac{2}{C \xi^4_{\min}} \sum_{i=1}^m \zeta_i (\xi_i - \widehat{\xi}_i)^2 \\ &= \frac{(\xi_{\min} + R) R^2}{\xi^5_{\min}} \sum_{i=1}^m \zeta_i (\xi_i - \widehat{\xi}_i)^2. \end{aligned}$$

Hence, proved. $\square$

B.6 Sensitivity of Accumulated Delay objective to Accuracy of Parameter Estimates

In this section, we consider the Accumulated Delay objective suggested by Sia et al. [2007]. The same objective can be arrived at if one formulates the online learning version of the Smart Broadcasting problem as formulated and solved by Zarezade et al. [2017a].

The objective's differential expected utility for the accumulated delay can be expressed as:

$$J(\boldsymbol{\rho};\boldsymbol{\xi}) = -\sum_{i=1}^{m} \frac{\zeta_i \xi_i}{\rho_i} \tag{B.16}$$

Sia et al. [2007] also show that the optimal solution which maximizes the utility (B.16) under the bandwidth constraint is given by

$$\rho_i^\star = \frac{R\sqrt{\zeta_i \xi_i}}{\sum_{j=1}^{m} \sqrt{\zeta_j \xi_j}} \tag{B.17}$$

$$\implies J(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) = -\frac{1}{R}\left(\sum_{i=1}^{m} \sqrt{\zeta_i \xi_i}\right)^2. \tag{B.18}$$

Lemma 7. *For the expected utility* $J(\boldsymbol{\rho};\boldsymbol{\xi})$ *defined in* (B.16), *let* $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}} J(\boldsymbol{\rho};\boldsymbol{\xi})$, $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} J(\boldsymbol{\rho};\widehat{\boldsymbol{\xi}})$ *and define the suboptimality of* $\widehat{\boldsymbol{\rho}}$ *as* $\operatorname{err}(\widehat{\boldsymbol{\rho}}) := J(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) - J(\widehat{\boldsymbol{\rho}};\boldsymbol{\xi})$. *Then* $\operatorname{err}(\widehat{\boldsymbol{\rho}})$ *can be bounded by:*

$$\operatorname{err}(\widehat{\boldsymbol{\rho}}) \le \frac{\xi_{\max}^2 \left(\sum_{i=1}^{m} \sqrt{\zeta_i}\right)^4}{R\zeta_{\min}\xi_{\min}^3} \sum_{i=1}^{m} (\xi_i - \widehat{\xi}_i)^2.$$

Proof. We will follow largely the same steps as in the proof of Lemma 6 to prove this result as well. Let $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}\in\Delta_R} J(\boldsymbol{\rho};\boldsymbol{\xi})$ and $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}\in\Delta_R} J(\boldsymbol{\rho};\widehat{\boldsymbol{\xi}})$. By the optimality conditions, $\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho}^\star)^T(\boldsymbol{\rho}-\boldsymbol{\rho}^\star) \le 0$ for all $\boldsymbol{\rho} \in \Delta_R$. We can bound the second derivative as well:

$$\begin{aligned}\frac{\partial^2 J(\boldsymbol{\rho};\boldsymbol{\xi})}{\partial\rho_i \partial\rho_j} &= -2\delta_{ij}\frac{\zeta_i\xi_i}{\rho_i^3} \\ &\le -2\delta_{ij}\frac{\zeta_{\min}\xi_{\min}}{R^3}.\end{aligned}$$

Taylor expanding $J(\boldsymbol{\rho};\boldsymbol{\xi})$ around $\boldsymbol{\rho}^\star$ gives for some point $\bar{\boldsymbol{\rho}}$ between $\boldsymbol{\rho}$ and $\boldsymbol{\rho}^\star$

$$\begin{aligned}J(\boldsymbol{\rho};\boldsymbol{\xi}) = J(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) &+ \underbrace{\nabla_{\boldsymbol{\rho}^\star} J(\boldsymbol{\rho};\boldsymbol{\xi})^T(\boldsymbol{\rho}-\boldsymbol{\rho}^\star)}_{\le 0} \\ &+ \frac{1}{2}(\boldsymbol{\rho}-\boldsymbol{\rho}^\star)^T \nabla_{\boldsymbol{\rho}}^2 J(\bar{\boldsymbol{\rho}};\boldsymbol{\xi})(\boldsymbol{\rho}-\boldsymbol{\rho}^\star) \\ \le J(\boldsymbol{\rho}^\star;\boldsymbol{\xi}) &- \frac{\zeta_{\min}\xi_{\min}}{R^3}\sum_{i=1}^{m}(\rho_i - \rho_i^\star)^2\end{aligned}$$

Take $\boldsymbol{\rho} = \widehat{\boldsymbol{\rho}}$, using the closed form optimal solutions for $\boldsymbol{\rho}^\star$ and $\widehat{\boldsymbol{\rho}}$ from (B.17), using $\boldsymbol{\xi}$ and $\widehat{\boldsymbol{\xi}}$ respectively, and rearranging terms, we have:

$$
\begin{aligned}
\operatorname{err}(\widehat{\boldsymbol{\rho}}) &= J(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - J(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) \\
&\geq \frac{\zeta_{\min}\xi_{\min}}{R^3} \sum_{i=1}^{m} (\widehat{\rho}_i - \rho_i^\star)^2 \\
&= \frac{\zeta_{\min}\xi_{\min}}{R^3} \sum_{i=1}^{m} \left(\frac{R\sqrt{\zeta_i \xi_i}}{\sum_{j=1}^m \sqrt{\zeta_i \xi_j}} - \frac{R\sqrt{\zeta_i \widehat{\xi}_i}}{\sum_{j=1}^m \sqrt{\zeta_j \widehat{\xi}_j}} \right)^2 \\
&= \frac{\zeta_{\min}\xi_{\min}}{R} \sum_{i=1}^{m} \left(\frac{\sqrt{\zeta_i \xi_i}}{\sum_{j=1}^m \sqrt{\zeta_i \xi_j}} - \frac{\sqrt{\zeta_i \widehat{\xi}_i}}{\sum_{j=1}^m \sqrt{\zeta_j \widehat{\xi}_j}} \right)^2
\end{aligned}
\tag{B.19}
$$

Next, for the upper bound on the error, consider

$$
\begin{aligned}
\operatorname{err}(\widehat{\boldsymbol{\rho}}) &= J(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - J(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) + \underbrace{J(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) - J(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}})}_{\leq 0} \\
&\qquad + J(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}}) - J(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) \\
&\leq J(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - J(\boldsymbol{\rho}^\star; \widehat{\boldsymbol{\xi}}) + J(\widehat{\boldsymbol{\rho}}; \widehat{\boldsymbol{\xi}}) - J(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi}) \\
&= -\frac{1}{R} \left(\sum_{i=0}^{m} \sqrt{\zeta_i \xi_i} \right)^2 - \frac{1}{R} \left(\sum_{i=0}^{m} \sqrt{\zeta_i \widehat{\xi}_i} \right)^2 \\
&\qquad + \frac{1}{R} \left(\sum_{i=1}^{m} \frac{\zeta_i \widehat{\xi}_i}{\sqrt{\zeta_i \xi_i}} \right) \left(\sum_{j=1}^{m} \sqrt{\zeta_j \xi_j} \right) \\
&\qquad + \frac{1}{R} \left(\sum_{i=1}^{m} \frac{\zeta_i \xi_i}{\sqrt{\zeta_i \widehat{\xi}_i}} \right) \left(\sum_{j=1}^{m} \sqrt{\zeta_j \widehat{\xi}_j} \right) \\
&= \frac{\left(\sum_{j=0}^m \sqrt{\zeta_j \xi_j} \right) \left(\sum_{j=0}^m \sqrt{\zeta_j \widehat{\xi}_j} \right)}{R} \\
&\qquad \times \left[\sum_{i=0}^{m} \left(\frac{\sqrt{\zeta_i \xi_i}}{\sum_{j=1}^m \sqrt{\zeta_i \xi_j}} - \frac{\sqrt{\zeta_i \widehat{\xi}_i}}{\sum_{j=1}^m \sqrt{\zeta_j \widehat{\xi}_j}} \right) \right. \\
&\qquad\qquad \left. \times \left(\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}} \right) \right]
\end{aligned}
$$

Denote $S = \sum_{j=0}^m \sqrt{\zeta_j \xi_j}$, $\widehat{S} = \sum_{j=0}^m \sqrt{\zeta_j \widehat{\xi}_j}$ and use Cauchy-Schwarz inequality $\sum_i x_i y_i \leq \sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}$

with $x_i = \left|\frac{\sqrt{\zeta_i\xi_i}}{\sum_{j=1}^{m}\sqrt{\zeta_i\xi_j}} - \frac{\sqrt{\zeta_i\widehat{\xi}_i}}{\sum_{j=1}^{m}\sqrt{\zeta_j\widehat{\xi}_j}}\right|$ and $y_i = \left|\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}}\right|$ to get:

$$
\begin{aligned}
\operatorname{err}(\widehat{\boldsymbol{\rho}}) &\le \frac{S\widehat{S}}{R}\left[\sum_{i=0}^{m}\left(\frac{\sqrt{\zeta_i\xi_i}}{S} - \frac{\sqrt{\zeta_i\widehat{\xi}_i}}{\widehat{S}}\right)\left(\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}}\right)\right] \\
&\le \frac{S\widehat{S}}{R}\left[\sum_{i=0}^{m}\left|\frac{\sqrt{\zeta_i\xi_i}}{S} - \frac{\sqrt{\zeta_i\widehat{\xi}_i}}{\widehat{S}}\right|\left|\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}}\right|\right] \\
&\le \frac{S\widehat{S}}{R}\sqrt{\sum_{i=0}^{m}\left(\frac{\sqrt{\zeta_i\xi_i}}{S} - \frac{\sqrt{\zeta_i\widehat{\xi}_i}}{\widehat{S}}\right)^2}\sqrt{\sum_{i=0}^{m}\left(\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}}\right)^2} \\
&\le \frac{S\widehat{S}}{R}\sqrt{\frac{R\operatorname{err}(\widehat{\boldsymbol{\rho}})}{\zeta_{\min}\xi_{\min}}}\sqrt{\sum_{i=0}^{m}\left(\sqrt{\frac{\widehat{\xi}_i}{\xi}} - \sqrt{\frac{\xi_i}{\widehat{\xi}_i}}\right)^2}
\end{aligned}
$$

where we have used (B.19) in the last inequality. Because we know $\forall i \in [m].\ \xi_{\min} \le \xi_i, \widehat{\xi}_i \le \xi_{\max}$, we have $S, \widehat{S} \le \sqrt{\xi_{\max}}\left(\sum_{i=0}^{m}\sqrt{\zeta_i}\right)$.

$$
\begin{aligned}
\sqrt{\operatorname{err}(\widehat{\boldsymbol{\rho}})} &\le \frac{S\widehat{S}}{\sqrt{R\zeta_{\min}\xi_{\min}}}\sqrt{\sum_{i=0}^{m}\frac{(\widehat{\xi}_i - \xi_i)^2}{\xi_i\widehat{\xi}_i}} \\
\operatorname{err}(\widehat{\boldsymbol{\rho}}) &\le \frac{S^2\widehat{S}^2}{R\zeta_{\min}\xi_{\min}}\sum_{i=0}^{m}\frac{(\widehat{\xi}_i - \xi_i)^2}{\xi_i\widehat{\xi}_i} \\
&\le \frac{\xi_{\max}^2\left(\sum_{i=1}^{m}\sqrt{\zeta_i}\right)^4}{R\zeta_{\min}\xi_{\min}^3}\sum_{i=1}^{m}(\xi_i - \widehat{\xi}_i)^2.
\end{aligned}
$$

Hence, proved. □

B.7 Proof of Lemma 5

Lemma 5. *For a given $\delta \in (0,1)$, after following the uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ for time τ, which is assumed to be a multiplicity of m/R, we can claim the following for the error in the estimates $\widehat{\boldsymbol{\xi}}$ produced using the estimator proposed in Lemma 3:*

$$
\mathbb{P}\left(\forall i \in [m] \colon |\widehat{\xi}_i - \xi_i| \le e^{\frac{\xi_{\max} m}{R}}\sqrt{\frac{R\log\frac{2m}{\delta}}{2\tau m}}\right) \ge 1 - \delta.
$$

Proof. Running the uniform-interval policy for time τ results in $N = \frac{\tau R}{m}$ observations collected for each webpage with time intervals $w_n = \frac{m}{R}$ for all $n = 1, \ldots, N$, including an observation made at $y_{i,0} := 0$, so that $\frac{1}{N}\sum_{n=1}^{N} w_n e^{-\xi_{\max} w_n} = \frac{m}{R}e^{-\xi_{\max} m/R}$. Substituting these in Lemma 3, we have that for the i-th webpage, with probability at most δ/m it holds

$$
|\widehat{\xi}_i - \xi_i| > \left(\frac{m}{R}e^{-\frac{\xi_{\max} m}{R}}\right)^{-1}\sqrt{\frac{\log\frac{2m}{\delta}}{2\frac{\tau R}{m}}} = e^{\frac{\xi_{\max} m}{R}}\sqrt{\frac{R\log\frac{2m}{\delta}}{2\tau m}}.
$$

By the union bound, the above event occur for any $i \in \{1, \dots, m\}$ with probability at most δ, which finishes the proof. $\square$

B.8 Utility of $\boldsymbol{\kappa}^{\text{ui}}$ and $\boldsymbol{\rho}^{\text{ur}}$

In this section, we show that the uniform-interval exploration policy $\boldsymbol{\kappa}^{\text{UI}}$ has lower regret than the uniform-rate $\boldsymbol{\rho}^{\text{UR}}$ exploration policy under the assumption that the exploration horizon τ is a multiple of $\frac{R}{m}$. The uniform-intervals policy $\boldsymbol{\kappa}^{\text{UI}}$ regularly refreshes each webpage at regular intervals of size $\frac{m}{R}$. Because τ is assumed to be a multiplicity of $\frac{m}{R}$, there will be exactly $\frac{\tau}{m/R}$ refreshes made of the webpages. By using the definition of utility given in (4.12), we can show that the expected utility of the uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ during the exploration phase is given by:

$$\begin{aligned}
&\mathbb{E}[U([0,\tau],\boldsymbol{\kappa}^{\text{UI}};\boldsymbol{\xi})] \\
&\quad = \frac{1}{m}\sum_{i=1}^{m}\zeta_i \int_0^{\tau} \mathbb{P}_{\boldsymbol{\kappa}^{\text{UI}}}(\textsc{Fresh}(i,t))\,dt \\
&\quad = \frac{1}{m}\sum_{i=1}^{m}\zeta_i \sum_{j=1}^{\frac{\tau}{m/R}} \int_{\frac{m(j-1)}{R}}^{\frac{mj}{R}} \mathbb{P}_{\boldsymbol{\kappa}^{\text{UI}}}\left(\textsc{Fresh}(i,t)\right)\,dt \\
&\quad = \frac{1}{m}\sum_{i=1}^{m}\zeta_i \sum_{j=1}^{\frac{\tau}{m/R}} \int_0^{\frac{m}{R}} \mathbb{P}_{\boldsymbol{\kappa}^{\text{UI}}}\left(\textsc{Fresh}(i,t)\right)\,dt \\
&\quad = \frac{1}{m}\frac{\tau}{\frac{m}{R}}\sum_{i=1}^{m}\zeta_i \int_0^{\frac{m}{R}} \mathbb{P}_{\boldsymbol{\kappa}^{\text{UI}}}\left(\textsc{Fresh}(i,t)\right)\,dt && \text{(B.20)} \\
&\quad = \frac{1}{m}\frac{\tau}{\frac{m}{R}}\sum_{i=1}^{m}\zeta_i \int_0^{\frac{m}{R}} e^{-\xi_i t}\,dt && \text{(B.21)} \\
&\quad = \frac{\tau}{m}\sum_{i=1}^{m}\zeta_i \frac{(1-e^{-\xi_i\frac{m}{R}})}{\xi_i\frac{m}{R}}
\end{aligned}$$

where the equality between (B.20) and (B.21) can be established by seeing that probability a page is fresh at time $t \in [0, \frac{m}{R}]$ is equal to the probability that no change event has occurred in $[0,t]$, *i.e.*, $E_i^{\emptyset}[0,t]$, which is equal to $\mathbb{P}(E_i^{\emptyset}[0,t]) = e^{-\xi_i t}$.

Next, the utility of the uniform-rates policy $\boldsymbol{\rho}^{\text{UR}}$ can be easily calculated under the conditions of the Lemma 1 by setting $\forall i.\, \rho_i = \frac{R}{m}$ in (4.14) as:

$$\mathbb{E}[U([0,\tau],\boldsymbol{\rho}^{\text{UR}};\boldsymbol{\xi})] = \frac{\tau}{m} \times F(\boldsymbol{\rho}^{\text{UR}};\boldsymbol{\xi}) = \frac{\tau}{m}\sum_{i=1}^{m}\frac{\zeta_i}{1+\xi_i\frac{m}{R}}$$

Hence, the regret suffered by the uniform-rate policy during exploration is greater than the regret suffered by the uniform-interval policy:

$$\begin{aligned}
&R(\tau,\boldsymbol{\rho}^{\text{UR}};\boldsymbol{\xi}) - R(\tau,\boldsymbol{\kappa}^{\text{UI}};\boldsymbol{\xi}) \\
&\quad = \frac{\tau}{m}\sum_{i=1}^{m}\zeta_i\left(\frac{1-e^{-\xi_i\frac{m}{R}}}{\xi_i\frac{m}{R}} - \frac{1}{1+\xi_i\frac{m}{R}}\right) \\
&\quad \geq 0
\end{aligned}$$

where the inequality follows from $e^x \geq 1 + x$, for any $x \in \mathbb{R}$.

B.9 Exploration using $\boldsymbol{\rho} \in \Delta_R$ and Committing to use $\boldsymbol{\kappa} \in \mathcal{K}_R$

To arrive at theoretical guarantees for the ETC algorithm, we perform exploration using uniform intervals policy $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$ while for the exploitation phase, we use a policy $\widehat{\boldsymbol{\rho}} \in \Delta_R$. In this section, we justify why we refrain from performing exploration using the policy $\boldsymbol{\rho}^{\text{UR}}$ or use a policy $\boldsymbol{\kappa} \in \mathcal{K}_R$ for commit phase.

B.9.1 Parameter Estimation using $\boldsymbol{\rho}^{\mathbf{ur}} \in \Delta_R$

The randomness added during the estimation phase makes it technically very difficult to bound the error in the estimates. Notice that Lemma 3 relies on knowing the distribution of the window lengths w_n and the total number of refreshes made N. If these quantities are random, *i.e.*, $N \sim \text{Poisson}(\frac{\tau R}{m})$ and w_n are inter-refresh times, then bounding the probability of arriving at accurate estimates $\widehat{\boldsymbol{\xi}}$, *i.e.*, $\mathbb{P}\left(|\widehat{\xi}_i - \xi_i| < \varepsilon\right)$, becomes intractable. On the other hand, we can circumvent these problems by using $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$ (see proof of Lemma 5).

B.9.2 Sensitivity of Utility to Parameter Estimation Accuracy using $\widehat{\boldsymbol{\kappa}} \in \mathcal{K}_R$

To allow for easy exposition, we will refer to policies in $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_m) \in \mathcal{K}_R$ via the corresponding policy in $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_m) \in \Delta_R$, such that $\rho_i = {}^{1}/_{\kappa_i}$. Then the differential utility function for policies in $\mathcal{K}_R$ can be written as (see Section B.8):

$$G(\boldsymbol{\rho}; \boldsymbol{\xi}) = \sum_{i=1}^{m} \zeta_i \frac{1 - e^{-\xi_i/\rho_i}}{\xi_i/\rho_i}$$

This function is also concave with respect to $\boldsymbol{\rho}$ and can be optimized using an algorithm similar to the one proposed by Azar et al. [2018] or Duchi et al. [2008], albeit with more involved calculations.

However, we run into a problem while trying to determine how sensitive the utility functions are to errors in estimation of the parameters $\widehat{\boldsymbol{\xi}}$. The second derivative of the utility function is given by

$$\frac{\partial^2 G(\boldsymbol{\rho}; \boldsymbol{\xi})}{\partial \rho_i^2} = -\frac{\zeta_i \xi_i e^{-\xi_i/\rho_i}}{\rho_i^3} \tag{B.22}$$

Contrasting it with the derivatives for the utility function for policies in Δ_R

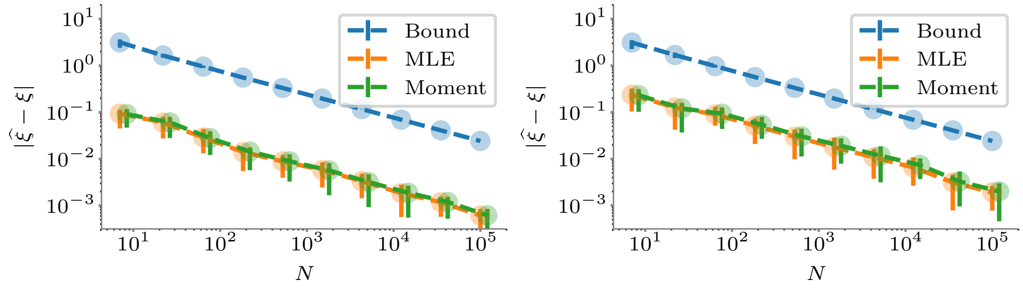
$$\frac{\partial^2 F(\boldsymbol{\rho}; \boldsymbol{\xi})}{\partial \rho_i^2} = -\frac{2\zeta_i \xi_i}{(\xi_i + \rho_i)^3}$$

reveals that while the second derivative of $F(\boldsymbol{\rho}, \boldsymbol{\xi})$ can be bounded if we have a bound $\xi_{\min}$ on the values of ξ_i, no such bound can be proposed for the second derivative of $G(\boldsymbol{\rho}; \boldsymbol{\xi})$ in (B.22), which can be arbitrarily close to zero. Hence, as the curvature for the objective function $G(\boldsymbol{\rho}; \boldsymbol{\xi})$ cannot be bounded, we will not be able to provide a quadratic bound akin to Lemma 4, in this setting.

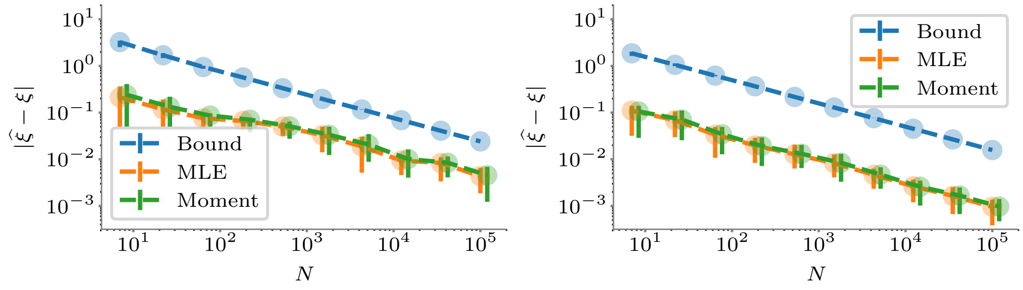
B.10 Performance of Moment Matching Estimator

In this section, we consider the performance of the moment matching estimator (4.17) and the bounds on its performance proposed in Lemma 3 on simulated and real data.

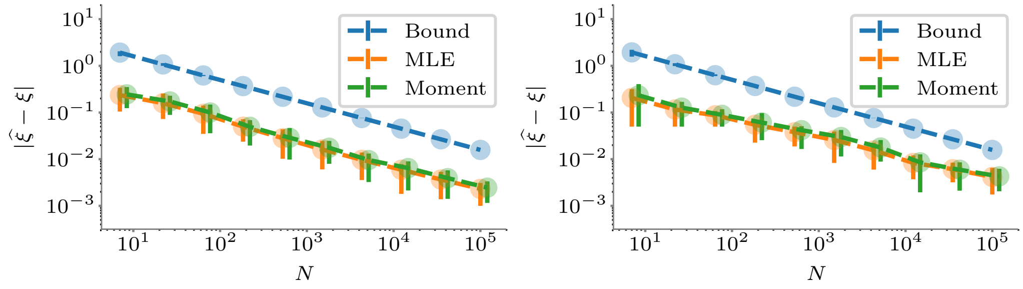
In the simulated experiments below, we have assumed that $\xi_{\min} = 0.1$ and $\xi_{\max} = 1.0$. For a fixed number of observations N, known $\boldsymbol{\xi}$ and a fixed random-seed, we simulate times to refresh a webpage $(y_0 := 0) \cup (y_n)_{n=1}^{N}$ as times drawn from a Poisson process with rate $\rho \in \{0.25, 0.75\}$. Then we draw $(o_n)_{n=1}^{N}$ by stochastically determining whether an event happened between $y_n - y_{n-1}$. Next, we calculate the estimates $\widehat{\xi}$ using the MLE estimator and the moments matching estimator. We also determine what is the bound on the error



(a) $\xi = 0.15$, $\rho = 0.25$ (b) $\xi = 0.50$, $\rho = 0.25$



(c) $\xi = 0.95$, $\rho = 0.25$ (d) $\xi = 0.15$, $\rho = 0.75$



(e) $\xi = 0.50$, $\rho = 0.75$ (f) $\xi = 0.95$, $\rho = 0.75$

Figure B.1: Error in the estimates produced by the FIG/visibility/moment matching estimator (in green) compared to the upper bound (in blue) and the MLE estimator (in orange) for three different values of ξ. To calculate the bound, $\xi_{\max}$ was assumed to be 1. The first row shows the estimation error when the refresh rate was given by $\rho = 0.25$ (refresh) events per unit time, while the second row shows the results for $\rho = 0.75$ events per unit time. The error bars show 25-75 percentiles of error across simulations.

proposed by Lemma 3 with $\delta = 0.1$. We record the error in the estimates and then re-run the simulation 50 times with different seeds. The results are shown in Figure B.1. Plots for other values of the parameters were similar qualitatively. It can be seen that the performance of the MLE estimator is not discernibly different from the performance of the moment matching estimator.

Similarly, we performed the same experiments with fixed-interval policies as well, *i.e.*, when we observed (refreshed) a page i at regular intervals of time $1/\rho$ instead of drawing the times from a Poisson process. Under this setting, both our estimator and the MLE estimator are identical and Figure B.2 shows the performance under the same setup as before. In both settings, we can see that the bound decreases with (i) increasing N within each plot, and with (ii) increasing ρ, which effects the size of interval $w_n = y_n - y_{n-1}$, across the rows. Also, the bound gets tighter as ξ gets closer to $\xi_{\max}$. Additionally, These figures also show that the bound is tight up-to constants irrespective of whether the observation (refresh) intervals are stochastic or deterministic, and it is tighter for stochastic intervals than for deterministic intervals.

For the real-data, we take a sample of 1000 webpages from the MSMACRO dataset [Kolobov et al., 2019b].

The dataset was crawled by Bing over a 14 week period with each page being visited roughly once every 2 days, *i.e.*, $\rho \approx 0.5$ updates/day. The best estimate we can make is given by the MLE estimate at the end of the 14 day period. We denote this as ξ_{MLE}. Figure B.3 shows that the error in estimates obtained after N observations. We see a similar trend as was seen in the synthetic experiments.

B.11 Parameter Estimation with Full Observations

In this section, we consider the full observability setting, *i.e.*, when instead of observing whether the page changed or not, one can observe the total number of changed which happened to the webpage in between two observations. This is often the case while retrieving, *e.g.*, ATOM/RSS feeds [Sia et al., 2007].

The total number of events which a Poisson process with rate ξ produces in time t is a Poisson random variable with mean $\xi \times t$. Additionally, we have:

Lemma 6 (Adapted from Pollard [2015])**.** *If X is a random variable with the distribution Poisson(λ), then:*

$$\mathbb{P}\left(X > \lambda + \varepsilon\right) \leq \exp\left(-\frac{\varepsilon^2}{2\lambda}\psi\left(\frac{\varepsilon}{\lambda}\right)\right)$$
$$\mathbb{P}\left(X < \lambda - \varepsilon\right) \leq \exp\left(-\frac{\varepsilon^2}{2\lambda}\psi\left(-\frac{\varepsilon}{\lambda}\right)\right)$$

where $\psi(t) = \frac{(1+t)\log(1+t)-t}{t^2/2}$ for $t \neq 0$ and $\psi(0) = 1$.

Assume that we are given a finite sequence of observation times $\{y_0 := 0\} \cup (y_n)_{n=1}^N$ in advance. Define x_i the number of events of the Poisson process which we observe taking place in the interval $[y_{i-1}, y_i)$. One can contrast it with the partial observability setting by comparing it with the definition of o_n given in (4.11). We will use the following estimator:

$$\widetilde{\xi} = \frac{1}{y_N}\sum_{i=1}^{N} x_i \tag{B.23}$$

and then clip it to obtain $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$.

Lemma 8. *Given observations times $(y_n)_{n=0}^N$ and observation of number of events $(x_n)_{n=1}^N$ for a Poisson process with rate ξ, for any $\delta \in (0,1)$, it holds that*

$$\mathbb{P}\left(|\widehat{\xi} - \xi| > \sqrt{\frac{2\xi_{\max}}{y_N \mathcal{M}}\log\frac{2}{\delta}}\right) < \delta$$

where $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$ and $\widetilde{\xi}$ is obtained by solving (B.23)*, and $\mathcal{M} = \psi\left(\frac{\xi_{\max}}{\xi_{\min}} - 1\right)$.*

Proof. We know that $\widetilde{\xi} = \sum_{i=1}^N x_i \sim \text{Poisson}(\xi \times y_N)$. Then using Lemma 6, substituting ε by $\varepsilon \times y_N$, we get

$$\mathbb{P}\left(\widetilde{\xi} < \xi + \varepsilon\right) \leq \exp\left(-\frac{\varepsilon^2 y_N}{2\xi}\psi\left(\frac{\varepsilon}{\xi}\right)\right) \tag{B.24}$$
$$\mathbb{P}\left(\widetilde{\xi} > \xi - \varepsilon\right) \leq \exp\left(-\frac{\varepsilon^2 y_N}{2\xi}\psi\left(-\frac{\varepsilon}{\xi}\right)\right) \tag{B.25}$$

Since $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$, we have $\xi, \widehat{\xi} \in [\xi_{\min}, \xi_{\max}]$ and, hence,

$$|\widetilde{\xi} - \xi| \geq |\widehat{\xi} - \xi| \tag{B.26}$$

$$\text{and} \quad |\widehat{\xi} - \xi| \leq \xi_{\max} - \xi_{min} \tag{B.27}$$

$$\implies \frac{|\widehat{\xi} - \xi|}{\xi} \leq \left(\frac{\xi_{\max}}{\xi_{\min}} - 1\right) \tag{B.28}$$

Define $\varepsilon' = \min\{\varepsilon, \xi_{\max} - \xi_{min}\}$.

It is easy to show that $\psi(t)$ is a decreasing function of t and, therefore, we have:

$$\psi\left(-\frac{\varepsilon'}{\xi}\right) \geq \psi\left(\frac{\varepsilon'}{\xi}\right) \geq \underbrace{\psi\left(\frac{\xi_{\max}}{\xi_{\min}} - 1\right)}_{\mathcal{M}} \tag{B.29}$$

where we have used inequality (B.28) in the last step.

The equations (B.24) and (B.25) can be combined and written as the following:

$$\begin{aligned}\mathbb{P}\left(|\widehat{\xi} - \xi| > \varepsilon\right) &\leq \mathbb{P}\left(|\widehat{\xi} - \xi| > \varepsilon'\right) \\ &\leq 2\exp\left(-\frac{\varepsilon'^2 y_N}{2\xi_{\max}}\psi\left(\frac{\varepsilon'}{\xi_{\min}}\right)\right) \\ &\leq 2\exp\left(-\frac{\varepsilon^2 y_N \mathcal{M}}{2\xi_{\max}}\right)\end{aligned}$$

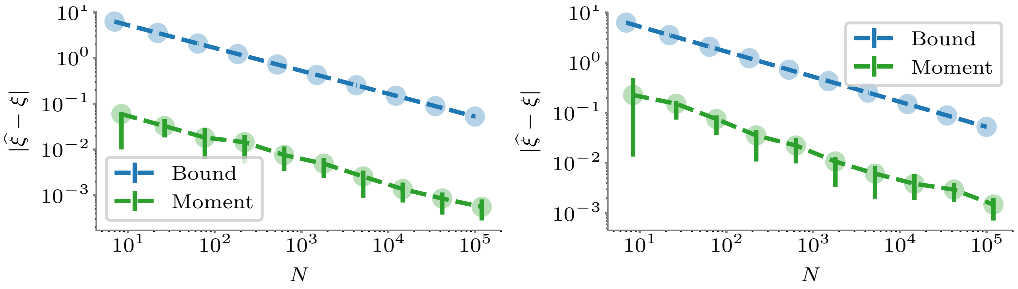
where the first and the last inequalities follow since $\varepsilon' \leq \varepsilon$.

Setting $\delta = 2\exp\left(-\frac{\varepsilon^2 y_N \mathcal{M}}{2\xi_{\max}}\right)$ and solving for ε, we get the desired result. ☐

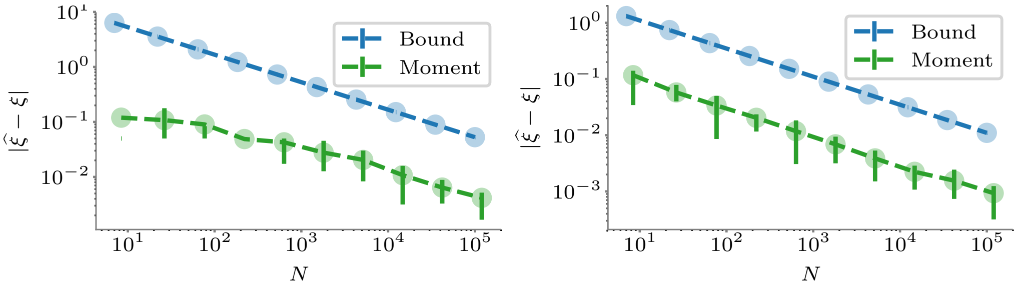
For the uniform-interval exploration till time τ, by using a union bound over all web-pages with Lemma 8, one can obtain:

$$\mathbb{P}\left(\forall i \in [m] : |\widehat{\xi}_i - \xi_i| \leq \sqrt{\frac{2\xi_{\max}}{\tau\mathcal{M}} \log\frac{2m}{\delta}}\right) \geq 1 - \delta \tag{B.30}$$

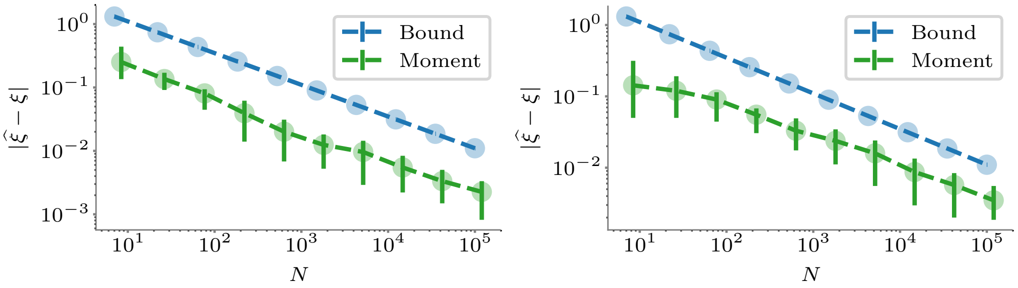
One can compare (B.30) with Lemma 5 to see the benefits of having full observability, *viz.*, (i) the bound does not depend on the bandwidth R as even a single observation at time τ provides the sufficient statistic for parameter estimation, and (ii) the dependence of the bound on $\xi_{\max}$ is approximately $\sim \mathcal{O}(\sqrt{\xi_{\max}})$ instead of $\sim \mathcal{O}(e^{c\xi_{\max}})$. However, since the dependence of the bound on the exploration time remains $\mathcal{O}(\sqrt{1/\tau})$, the regret suffered by the ETC algorithm is still $\mathcal{O}(\sqrt{T})$.



(a) $\xi = 0.15$, $\rho = 0.25$ (b) $\xi = 0.50$, $\rho = 0.25$



(c) $\xi = 0.95$, $\rho = 0.25$ (d) $\xi = 0.15$, $\rho = 0.75$



(e) $\xi = 0.50$, $\rho = 0.75$ (f) $\xi = 0.95$, $\rho = 0.75$

Figure B.2: Error in the estimates produced by the FIG/visibility/moment matching estimator for three different values of ξ under the uniform-interval setting. The MLE estimator is not plotted here because it coincides with the FIG/visibility/moments matching estimator. To calculate the bound, $\xi_{\max}$ was assumed to be 1 and that all intervals are equal with $w_n = {}^1/_\rho$. The first row shows the estimation error when the refresh rate was given by $\rho = 0.25$ (refresh) events per unit time, while the second row shows the results for $\rho = 0.75$ events per unit time. The error bars show 25-75 percentiles of error across simulations. Compared to Figure B.1, we can see that the bound is slightly looser in the deterministic setting, but the same trend is seen that the bound gets tighter the closer ξ gets to $\xi_{\max}$.

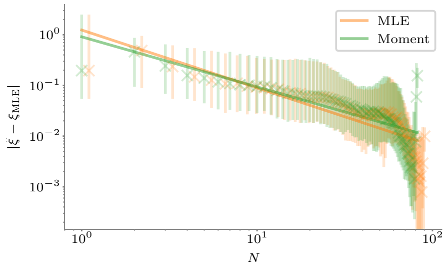


Figure B.3: Performance of the Moment Matching estimator and MLE estimator on real-data when compared against the best possible MLE estimate for each webpage (indicated by ξ_{MLE}. The sampling policy was decided by the Bing web-crawler and is close to $\rho \approx 0.5$.

Appendix C

Curriculum vitae

Current Position

- **Jun '15 – present** Ph.D. Candidate at MPI-SWS. Thesis Advisor: Dr. Manuel Gomez Rodriguez
- **Oct '19 – present** CTO/Co-founder of Reason.al

Education

- **Sept '09 – Feb '12** M.S. in Computer Science at EPFL, Switzerland
- **Aug '05 – May '09** B.Tech. in Electrical Engg. at Indian Institute of Technology, Kanpur

Academic Honors

- **2017-19** Yahoo! Academic Research Program (ARP) Collaborator
- **2016** Travel award for attending KDD '16
- **2012** Awarded *Prix ECLA Informatique* for highest GPA in Computer Science MS at EPFL
- **2009-10** Awarded the Academic Excellence scholarship for studying at EPFL
- **2009** Silver medal at the programming competition ACM-SWERC '09
- **2005-09** Recipient of the Aditya Birla Scholarship (Given to 10 students all over India)
- **2005-06** Recipient of the Academic Excellence Award at IITK
- **2005** Secured 108^{th} rank in Joint Entrance Examination for IITs (Percentile: 99.97%)

Publications

- **U. Upadhyay**, R. Busa-Fekete, W. Kotlowski, D. Pal, B. Szorenyi. *Learning to Crawl.* **Oral Presentation**, Association for the Advancement of Artificial Intelligence (AAAI) 2020.
- A. De, A. Singla, **U. Upadhyay**, M. Gomez-Rodriguez. *Can a User Guess What Her Followers Want?* ACM Int'l Conference on Web Search and Data Mining (WSDM) 2020.
- B. Tabibian, **U. Upadhyay**, A. De, A. Zarezade, B. Schölkopf, M. Gomez-Rodriguez. *Enhancing human learning via spaced repetition optimization.* Proceedings of the National Academy of Sciences (PNAS) 2019.

- **U. Upadhyay**, A. De, A. Pappu, M. Gomez-Rodriguez. *On the Complexity of Opinions and Online Discussions.* ACM Int'l Conference on Web Search and Data Mining (WSDM) 2019.
- **U. Upadhyay**, A. De, M. Gomez-Rodriguez. *Deep Reinforcement Learning of Marked Temporal Point Processes.* Conference on Neural Information Processing Systems (NeurIPS) 2018.
- A. Zarezade, A. De, **U. Upadhyay**, H. R. Rabiee, M. Gomez-Rodriguez. *Steering Social Activity: A Stochastic Optimal Control Point of View.* Journal of Machine Learning Research (JMLR) 2018.
- **U. Upadhyay**, I. Valera, M. Gomez-Rodriguez. *Uncovering the Dynamics of Crowdlearning and the Value of Knowledge.* ACM Int'l Conference on Web Search and Data Mining (WSDM) 2017.
- A. Zarezade, **U. Upadhyay**, H. Rabiee, M. Gomez-Rodriguez. *RedQueen: An Online Algorithm for Smart Broadcasting in Social Networks.* ACM Int'l Conference on Web Search and Data Mining (WSDM) 2017.
- S. L. Blond, C. Gilbert, **U. Upadhyay**, M. Gomez-Rodriguez, D. Choffnes. *A Broad View of the Ecosystem of Socially Engineered Exploit Documents.* Network and Distributed System Security symp. (NDSS) 2017.
- N. Du, H. Dai, R. Trivedi, **U. Upadhyay**, M. Gomez-Rodriguez, L. Song. *Recurrent Temporal Point Processes: Embedding Event History to Vector.* ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) 2016.
- R. Khalili, N. Gast, M. Popovic, **U. Upadhyay**, J.-Y. Le Boudec. *MPTCP is not Pareto-optimal: Performance issues and a possible solution.* **Best Paper Award**, ACM SIGCOMM Int'l Conference on emerging Networking Experiments and Technologies (CoNEXT) 2012.
- **U. Upadhyay**. *Reliable communication and control for the Smart Grid*, MS Thesis, EPFL 2012.
- G. Mermoud, M. Mastrangeli, **U. Upadhyay**, A. Martinoli. *Real-Time Automated Modeling and Control of Self-Assembling Systems.* IEEE Int'l Conference on Robotics and Automation (ICRA) 2012.
- **U. Upadhyay**, N. Yoshinaga, S. Itaya, R. Tanaka, T. Konishi, S. Doi, K. Yamada, P. Davis. *Evaluation of Participants in Group-discussions on Twitter.* 8^{th} Applications of Social Network Analysis (ASNA) 2011.
- G. Mermoud, **U. Upadhyay**, W. C. Evans, A. Martinoli. *Top-Down vs Bottom-Up Model-Based Methodologies for Distributed Control: A Comparative Experimental Study.* Int'l Symp. on Experimental Robotics (ISER) 2010.
- R. Arora, **U. Upadhyay**, R. Tulshyan, K. Deb, J. Dutta. *A Parallel Randomized Algorithm for Solving Large Convex Minimax Problem.* Conference on Simulated Evolution and Learning (SEAL) 2010. Appears in LNCS, Springer, Vol. 6475.
- R. Arora, **U. Upadhyay**. *Can ETF Arbitrage be Extended to Sector Trading?* 1^{st} IIMA Conference on Advanced Data Analysis, Business Analytics & Intelligence, Indian Institute of Management, Ahmedabad, June - 2009

Work and Services

- **Apr '12 — Nov '15** Software developer and data scientist at Better AG, Zurich.
- Reviewer for: ICLR ('20), ICML ('19,'18), NeurIPS ('19,'18,'17,'16,'15), ICDM ('19), WSDM ('16), KDD ('19,'16), IJCAI ('16), SDM ('16), WWW ('16), AISTATS ('16), AAAI ('16), CIKM ('15)
- Teaching assistant for Human Centered Machine Learning 2018-19 at Saarland University
- Taught a tutorial on Network Analysis in Machine Learning Summer School (MLSS) 2017 as MPI-IS.
- Volunteer for NeurIPS 2017
- Invited talk at Yahoo! Research NYC on *RedQueen* in Dec. '17.
- Co-organized and taught a tutorial on Point Processes in MLSS 2016 in Càdiz, Spain
- Teaching assistant for the seminar course on Social and Information Networks at TU-KL, 2016