Methods for Improving Usability of Online User Generated Content

Lahari Poddar

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Department of Computer Science
School of Computing
National University of Singapore

2019

Supervisor
Professor Wynne Hsu

Thesis Examiners
Associate Professor Roger Zimmermann
Professor Leong Tze Yun
Professor Liu Huan, Arizona State University

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Lahari Poddar

November 2019

Abstract

User Generated Content (UGC) in forms of reviews, numeric ratings, blogs, posts in forum and social media are present in an overwhelming amount to help users make informed decisions about various products or services. Even though helpful, unfortunately many of these posts are not accurate, and might be biased by an individual's opinion or idiosyncratic experiences. This limits their usability as general reliable information sources. As opposed to prior work on binary truth discovery, we argue that UGC can not be judged by such a harsh universal lens of credibility; due to the fine grained subjectivity present in them owing to individual preferences. We also believe that the nature of UGC is strongly domain-dependent and it is crucial to capture the domain-specific nuances for modelling user feedbacks properly.

In this thesis, we focus on a few widely popular domains where people increasingly rely on UGC, namely, e-commerce/services and e-health. In these domains people can share their feedback on an entity (e.g. products in e-commerce, hotels or restaurants in services, drugs and treatments in health forums) in various forms (such as ratings, reviews, posts, lists of observed side effects). We hypothesize that such user feedbacks might be influenced by some underlying confounding factors, that make one user's experience different from another, even for the same entity. Faced with such varying opinions about the same entity, it becomes difficult for a person to make a decision about its quality. For instance, in the context of products, when one looks at conflicting ratings given by users on different aspects of an item, he/she needs to be aware of the biases which influenced their ratings, to estimate the true quality of the item. While going through diverse reviews written by strangers, it is important to know whether a particular opinion expressed in a review is prevalent or rare, before relying on it completely. For health-related information, having a long list of side effects associated with a particular drug, reported by various people with diverse backgrounds, is confusing and intimidating for a person to whom they might not even apply. In social media where people freely argue and voice their opinions on recent events, it is essential to know the veracity of their claims to prevent being mislead by unreliable information.

We propose a range of data driven methods to automatically handle such inherent subjectivity in user opinions, and identify the roles of the confounding factors behind the observed UGC footprint. We devise new frameworks based on probabilistic graphical models as well as neural networks accordingly. We have validated our models by using them for practical applications such as, (1) quantifying the aspect biases of users to better interpret their observed ratings, (2) retrieving supporting reviews for an individual's opinion to facilitate consensus modeling, (3) predicting user specific drug side effects, and (4) detecting veracity of rumors on social media. Experimental evaluation on a number of real world datasets show the effectiveness of our models for handling user generated content and sets new benchmarks across domains.

Publications

The components of the thesis have been published as separate independent top-tier conference papers:

1. **Lahari Poddar**, Wynne Hsu, and Mong Li Lee. *Quantifying aspect bias in ordinal ratings using a bayesian approach.* In Proceedings of the 26th International Joint Conference on Artificial Intelligence (**IJCAI**). AAAI Press, 2017. (Section 3) [90]

2. **Lahari Poddar**, Wynne Hsu, and Mong Li Lee. *Author-aware Aspect Topic Sentiment Model to Retrieve Supporting Opinions from Reviews.* In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (**EMNLP**). 2017. (Section 4) [89]

3. **Lahari Poddar**, Wynne Hsu, Mong Li Lee, Shruti Subramaniyam *Predicting Stances in Twitter Conversations for Detecting Veracity of Rumors: a Neural Approach*, In Proceedings of IEEE 30th International Conference on Tools with Artificial Intelligence (**ICTAI**). 2018 , **Best Student Paper Award** (Section 6) [91]

4. **Lahari Poddar**, Wynne Hsu, and Mong Li Lee. *Predicting User Reported Symptoms Using a Gated Neural Network.* Accepted in IEEE 31st International Conference on Tools with Artificial Intelligence (**ICTAI**). 2019 . (Section 5) [92]

Contents

List of Figures

List of Tables

Chapter 1

Introduction

User Generated Content (UGC) has now become prevalent due to the rise of social media (e.g. Facebook, Twitter), online review portals (e.g. Amazon, Yelp, TripAdvisor), and forums (StackOverflow, HealthBoard), which facilitate sharing of information on a wide range of topics like health, politics, movies, products, travel, and more. Before making a decision, it is natural for a person to seek opinions from others who have done it before. Previously, we asked people around us, now we rely on online UGC from strangers. A recent survey [6] found that 86% of the respondents feel UGC is generally a good indicator of the quality of a brand, service, or products. A high percentage of people do not complete various purchases without consulting UGC, including major electronics (44%) and cars (40%), as well as hotel stays (39%), and travel to specific destinations (32%). Even for medical information, online health communities constitute an important source, with 59% of the adult US population seeking health-related information from online resources [25], and nearly half of US physicians relying on them for professional use [24].

1.1 Motivation

The open nature of UGC platforms attracts a lot of users to express their opinions freely and share with others. These platforms have facilitated democratization of content production; earlier traditional "gatekeepers" such as newspaper editors and publishers had to approve all content and information before it could be published. Now with the advent of

web technology, large numbers of individuals are able to freely post their opinions and experiences online, with little or no filters. Therefore, the generated content could be highly noisy, unreliable [76], subjective, spams [74, 80] or even rumors [65]. This brings us to an important question and the central theme of this thesis.

How can this vast amount of information be reliably used?

Answering this question requires a systematic study of information reliability across many domains that allow user generated content. There exist different lines of research work that try to tackle this problem in part. Existing work regarding credibility analysis tries to assess the binary truth regarding an entity. These methods try to automatically classify claims in true vs. false [94], reviews in spam vs. genuine [74], posts in deceptive vs. real [51], reported side effects in substantiated vs. unsubstantiated [76]. We acknowledge that such binary credibility problems exist in these platforms and it is necessary to first filter out such intentionally misleading or deceptive information, but it is also important to to go one step further.

"Two people can look at the exact same thing and see something totally different."

Different people come from different backgrounds, have different preferences and can have completely different experiences with same entity. We argue that even genuine opinions expressed in a 'real' user's post may still not generalize and be applicable for everyone. Issues of information reliability in online UGC have not been studied much in literature in light of this varying user experiences. In this thesis we primarily wish to focus on the inherent biases and individual experiences present in a user's feedback that makes it difficult to accept that as a universal fact and therefore limits its usability as a general information source. We want to devise methods and mechanisms that can help information seekers use the information present in UGC reliably in light of the individual subjectivity.

We hypothesize that there are some **confounding factors** that determine a user's experience with an entity and thus influence her feedback. We aim to learn and incorporate

these factors into our Machine Learning based models, to help an information seeker gain better insights from tons of UGC available on the web. In light of the confounding factors, the diverse feedback from users would be easier to interpret for the information seekers and they would be able to decide which feedback to rely on.

Depending on the domain, these confounding factors could be quite distinct from each other. Let us provide more context around this by a few illustrations.

(1) Consider the user feedbacks left in forms of reviews and ratings by various users on e-commerce websites like `Amazon, Ebay` or websites for hotels or restaurants like `TripAdvisor, Yelp`. The confounding factor here could be a user's preferences or biases for different aspects of an item, that determine a user's expectations from the item and in turn affect their experience. For example, for a user if *cleanliness* of a hotel is most important, her rating or comments on that aspect is likely to be more critical compared to her opinion on other aspects. For different users these biases might be different, hence their feedback on the same hotel would look very different from one another. This makes it very difficult for an information seeker to interpret these conflicting ratings without knowing the underlying user biases. Learning about these biases would help interpret a user's review or rating better and enable a person to make an informed decision regarding which opinions to rely on.

In addition to the overall experience captured in explicit numeric ratings, the textual reviews express user opinions at a finer granularity. People often leave specific feedback regarding the things they did or did not like about an entity. For example, for a hotel these can range from a leaky tap in the bathroom, or spotting a bed bug or annoying noises from a construction nearby, to the wide variety of tropical fruits available in the breakfast menu and so on. If a person is considering booking a hotel and comes across a review mentioning for e.g. a poor experience with housekeeping service, then it is important for her to know if that was an occasional problem or happens frequently with other guests or the particular user generally has higher expectations from service of a hotel. Given the large volume of reviews, it is impossible for an individual to go through all the posts in order to determine whether the opinion or complaint is frequent or biased. A framework

that can assist a person to look for consensus around a particular opinion expressed in a review will enable her to verify whether it is prevalent.

(2) In online health forums, users describe their experiences with an entity such as drug or a treatment in terms of their effectiveness and observed symptoms or side effects. However, the experienced symptoms could be widely different for different patients depending on some confounding factors, like, their demographic profile, existing medical conditions or other concurrent drug usage. This will make it very difficult for a user with no medical training, to accurately claim a symptom as a side effect of a drug. Such reports could also cause anxiety among people trying to learn about drug side effects through web search before deciding to consume a medicine. Knowing about the likely side effects that can occur for a patient, given her medical conditions, would help people make informed decisions when choosing between alternate treatments.

1.2 Contributions

These real-world challenges motivate us to develop solutions for addressing the reliability concerns inherent to using information from User Generated Content. However, we realize that it is infeasible to build a universal solution that can handle all sorts of UGC, due to the aforementioned domain-specific nuances. Guided by our key hypothesis around user subjectivity, we develop data-driven solutions using probabilistic graphical models and neural network architectures for modeling UGC, tailored to the specific domain at hand.

This thesis is a step towards alleviating the above-mentioned issues in this complex and pressing task of reliable use of information in the following major UGC domains.

e-Commerce: We aim to uncover the effect of users' latent aspect preferences or biases that affect their ratings. We hypothesize a user's rating for an aspect (e.g. service) of an item (e.g. hotel), depends on both the user's bias of the aspect and the quality of the item for that aspect. We develop a probabilistic graphical model (AspectBias) for the observed aspect ratings that jointly infers each user's aspect bias and the latent intrinsic quality of an item. We introduce latent user groups in our model that leverages similarity

of preferences among different sets of users, to deal with the data sparsity issue commonly observed in e-commerce platforms. Our model also overcomes two key limitations of prior work on modeling aspect ratings by (i) encoding the correlation among aspects through multivariate Gaussian distributions, and (ii) modeling the proper ordinal nature of user ratings through proper statistical formulation. We describe our proposed approach and its evaluations in detail in Chapter 3.

In our next chapter (Chapter 4), we aim to capture the fine grained opinion expressed in reviews in order to build a framework that can help a user verify whether an opinion is prevalent. Unlike ratings, people express specific feedback on many aspects of an item in their reviews through unstructured and noisy text with varying vocabulary. This makes it a challenging problem to capture opinions and determine their 'equivalence' to one another for consensus modeling. We first develop an Author-aware Aspect Topic Sentiment model (Author-ATS) for capturing opinions expressed in a review. In contrast to existing document topic models, our approach is suitably designed for capturing opinions by using hierarchical probabilistic modeling of text that assumes each word in a review expresses a sentiment towards some aspect of an entity. This model also encodes the characteristics of the author by considering (i) their aspect bias and (ii) the natural coherent writing style of reviews. By using the opinion model learned by Author-ATS, we further develop a framework (SURF) that helps in finding supporting reviews for a target opinion in order to facilitate consensus modeling.

e-Health: In this chapter of the thesis we focus on uncovering the confounding factors behind people's varying experiences with the same medical treatment. From a real world dataset collected from a health tracking app, we observe that different people experience different symptoms with varying degree of severity while undergoing the same treatment. Apart from the drug, they could potentially stem from multiple confounding factors such as the characteristics of the patients, her existing medical conditions, concurrent drug usage etc. We conduct an initial study and find that among the symptoms reported by users, there exists a significant percentage of *unsubstantiated* (not associated with the drug as per expert medical knowledge base side effects). We further find that

many of these side effects are, in fact, more correlated to the underlying medical condition(s) of the user than the drug for which they are reported. This motivates us to model this complex relationship between a user, her pre-existing medical condition(s) and the treatment(s) to better understand the symptoms she might expect. We develop a neural architecture, namely Multi Objective Mixture of Experts (MoMEx) for personalized prediction of the side effects and their severity score based on the interaction between user, drug, and conditions. Our architecture considers the role of these three confounding factors and probabilistically combines their predictions using a gating network. This allows us further insights into the decision making of the model to provide explanation for why the model predicts a certain symptom for a user. We describe this in detail in Chapter 5.

Social Media: Apart from the subjectivity issues present in the domains discussed above, we realize that the reliability problem also exists for factoid UGC platforms. Lately, due to the popularity of social media and its huge network effects, arresting the spread of misinformation in open platforms like Twitter and Facebook has demanded urgent research attention [127, 29, 2]. We dedicate the last chapter of this thesis (Chapter 6) towards tackling the information reliability problem in an open domain platform like social media. We propose to use the wisdom of crowds in order to build a framework for early detection of rumors. We notice that when an unverified news start spreading while a lot of people merely comment or blindly support the story, some people point out discrepancies or raise questions or doubts on the authenticity of its source or point out discrepancies in the story. Mining these discussions around the story and leveraging the wisdom of people can help detect a false rumor early. Thus we build a two-stage framework that can determine the veracity of a claim floating around in social media while using the clues in people's discussions around it. We first develop a novel neural network architecture to determine the stances of people engaging in a conversation on Twitter regarding a rumor. We then aggregate the stances to predict the veracity of the overall rumor circulating on social media. This framework should help in flagging false rumor stories early, before they get widely circulated and become disruptive.

We carry out extensive evaluation of all our models using real world datasets from

multiple domains such as `TripAdvisor, OpenTable, Yelp` for e-commerce and services; a health tracking app named `FlareDown` for e-Health, and one of most popular social media `Twitter`. Experimental results demonstrate the efficacy of our models with respect to competitive state-of-the-art methods and establish new benchmarks across domains. We also analyze the outputs of our model with various case studies to illustrate confounding factors and our ability to model them properly. We further design practical web applications that can provide insights to users for helping them make better sense of the conflicting and overwhelming amount of UGC data.

1.3 Organization

The thesis is organized as follows. Chapter 3 describes our approach for modeling the latent users biases in observed aspect ratings. In Chapter 4. we present our SUpporting Reviews Framework for consensus modeling. Chapter 5 demonstrates our analysis of reported side effects in a popular health forum and details of our neural network approach for predicting the user reported drug side effects by considering confounding factors. In Chapter 6, our work on developing a neural network architecture for rumor detection is presented. Finally, we conclude the report in Chapter 7 with summarizing our findings and outlining few future directions for research.

Chapter 2

Related Work

This chapter provides an overview of related work in relevant domains, the state-of-the-art approaches, their background and limitations. We first present some background on the technical components that we would be using in our models across different chapters. This is followed by brief descriptions of the other related modeling techniques that we compare our approaches with.

2.1 Probabilistic Graphical Models

In the following subsections we provide a brief overview of Probabilistic Graphical Models (PGM), its inference mechanisms and its application to text for topic models. For readers interested to learn about PGM in-depth, we highly recommend referring to these comprehensive materials [41, 103].

Probabilistic Graphical Models (PGM) provide a unified framework to capture dependencies between random variables using joint probability distributions. PGM bridges the concepts of probability and graph theory. It represents the relationship between a set of variables - where the variables are represented as nodes and their interactions as edges in a graph. PGMs leverage the *local relationships* in a graph to factor the complete joint distribution to more economic conditional distributions. For example, consider there are n discreet random variables in a graph $\{x_1, x_2, \cdots, x_n\}$, each of which can take up r distinct values. Then the complete joint distribution would require r^n number of values to

store and learn, which will be intractable. In a real-world scenario all variables might not be dependent on every variable in the graph and some variables might be *conditionally independent* given a subset of variables. The presence of such dependencies between subsets of variables enables us to factor the graph into smaller sets of variables. The overall joint distribution can then be computed as the product of these conditional distributions among subsets of variables.

Graphical Models primarily are of two types: directed graphical models (known as Bayesian Networks or **D**irected **A**cyclic **G**raph) and undirected graphical models (known as **M**arkov **R**andom **F**ields). In this thesis, we focus on Bayesian Networks.

Bayesian Inference

PGMs model the variables and their relationships and can therefore be used to learn the values of unobserved variables from the rest of the graph. In most real applications of PGM, an exact inference becomes computationally intractable due to the large number of connected components in the graph and we need to resort to *approximate posterior inference*. There are different algorithms of approximate inference for learning of PGMs. The two most popular families of inference algorithms are Markov Chain Monte Carlo (MCMC) Sampling and Variational Inference. MCMC inference approximates a posterior distribution by drawing a large number of samples from the distribution. Under the family of MCMC sampling based algorithms, Gibbs Sampling is one of the most widely used ones.

Gibbs Sampling is an iterative algorithm. In each iteration, it samples a value for each latent variable, conditioned on other variables. If we continue to do this many times(iterations), the resulting sample will be a sample from the exact posterior. The reason is that we have defined a Markov chain whose state space are the latent variables and whose *stationary distribution* is the posterior. Therefore, after a sufficiently large number of iterations the marginal distributions of the latent variables will be the exact posterior. Collapsed Gibbs Sampling is a variant of Gibbs Sampling, where we marginalize some variables while sampling a variable. This generally leads to a faster convergence. We use

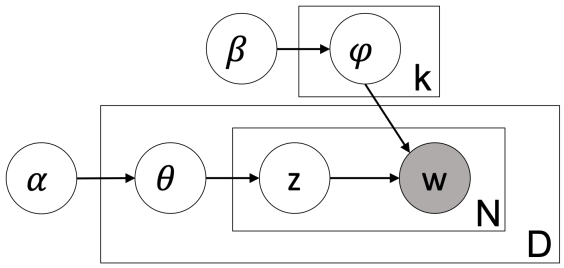


Figure 2.1: Graphical Model for LDA shown using Plate notation. The outer plate represents documents(M), while the inner plate represents the N words(w) within a document generated from topics(z). The greyed variable w is the only observed variable whereas the others are latent.

Collapsed Gibbs Sampling for inference of our PGMs.

Topic Models

Topic Models apply the principles of PGMs for textual documents to learn about latent semantic clusters (topics) from sets of documents. Latent Dirichlet Allocation (LDA)[7] is the pioneering work in topic modeling from text using PGM. LDA assumes that each document has a distribution of semantic topics and each topic is defined as a distribution over words.

Let's assume a collection of documents (D), where each document(d) consists of N_d words $\{w+1, w_2, \cdots, w_{N_d}\}$. Figure 2.1 shows the graphical representation of the LDA model.

LDA assumes the following generative process

- For each topic j, $1 \leq j \leq k$, draw word-topic distribution ϕ_j from Dirichlet(β)
- for each document d in D,
  - Draw a Multinomial topic mixture θ_d from Dirichlet(α)
  - For each word position i, where $1 \leq i \leq N_d$
    - Draw a topic z_i from Mulitinomial(θ)
    - Draw a word token w_i from Multinomial(β), conditioned on z_i

where θ represents document-topic distribution, ϕ represents word-topic distribution and k denotes the number of topics.

LDA has spurred interest in topic models and has inspired a large body of research works in the following decade on developing advanced topic models for various applications of text understanding. For modeling different facets of opinionated texts there has been some advanced topic models proposed in the literature, which we contrast and compare with our proposed model in Chapter 4. Apart from its inability to handle opinions in text, LDA also makes a number of simplifying assumptions regarding the structure of a document and the distribution of topics inside a document. Some of these assumptions we try to relax in our proposed model Author-ATS as described in Section 4.3.

2.2 User Modeling Frameworks

Since we wish to understand the latent user biases and preferences behind the observed feedback, out work often overlaps with Recommendation Systems (RS). Recommendation Systems are usually used for commercial purposes for recommending a product (e.g. Amazon) or movie (e.g. Netflix) or hotel (e.g. Expedia) to a user based on his/her past preferences expressed through numeric ratings.

Collaborative Filtering

Collaborative Filtering is one of the most popular approaches for building a recommendation system based only on the ratings given by users for different items. It uses a rating matrix (e.g. shown in Figure 2.2). The basic assumption of collaborative filtering is that users with similar preferences would like similar items. Therefore, given the past choices made by the users, the system tries to predict the values of the missing entries in the matrix.

Non-negative Matrix Factorization (NMF) [54] has been a widely adopted technique for building collaborative filtering systems. For a rating matrix R with m rows (users) and n columns (items), NMF factors the matrix in two matrices $U \in m \times k$ and $v \in k \times n$. This maps the users and items to a shared latent space of dimension k, where the i^{th} user is represent as the i^{th} row in U matrix, denoted by $\mathbf{u_i} \in \mathbb{R}^k$. Similarly the j^{th} item is

Items

Users	i1	i2	i3	i4
u1	4	2	?	1
u2	4	?	?	?
u3	2	3	1	?
u4	3	?	5	2
u5	?	1	?	2

Figure 2.2: A sample Rating Matrix consisting of 4 items and 5 users. The numbers in a matrix cell (i, j) denotes the rating given by the i^{th} user for the j^{th} item. ? denotes missing entries.

represented by $\mathbf{v_j} \in \mathbb{R}^k$. The predicted rating r_{ij} of user i for the j^{th} item is computed as a product of their latent representations,

$$r_{ij} = \mathbf{u_i}^{\mathrm{T}} \mathbf{v_j} \tag{2.1}$$

One pertaining challenge for user feedback datasets is the data sparsity problem - not always are a lot of feedback is available from the same user to reliable model their preferences. CF based methods often suffer due to this challenge. The above described matrix factorization method have been generalized to probabilistic models, Probabilistic Matrix Factorization (PMF) [72] and its Bayesian extension BPMF [106]. These probabilistic approaches can handle data sparsity better and scale linearly with the number of observations. Another competitive generative model for rating prediction is URP [3]. URP is a generative collaborative filtering model for rating prediction and learns a rating profile for modeling a user's ratings. It has been shown to outperform other mixture based generative models for the task of rating prediction. SVD++ [49] is one of the most competitive rating prediction algorithms that merges two powerful concepts in collaborating filtering, namely, latent factorization and neighborhood approach. Factorization Machines (FM) [101] combine the advantages of Support Vector Machines with that of latent factorization models for general prediction tasks, including rating prediction. These methods are some of the state-of-the-art rating prediction models across datasets from different domains and we evaluate our proposed models against them. Experimental results show that our models can comfortably outperform them by modeling the confounding factors

properly in order to predict the observed ratings.

Neural Collaborative Filtering

One inherent limitation of the traditional CF approaches is its inability to model non-linear relations between the user and latent factors. Neural Collaborative Filtering (NCF) [33, 128] builds a collaborative filtering system based on neural networks that can encode non-linear complex relationship between user and item latent factors. The input layer of the network consists of user vector and item vector, which are sparse one-hot encoding of the user and item's identities. The latent feature vectors of the user ($\mathbf{u_i}$) and item ($\mathbf{v_j}$) are then learned using an embedding layer to transform the one-hot vectors to lower-dimensional representations. These latent vectors are then fed to Multi-Layer Perceptrons with added non-linearity to make the final prediction. However these models focus on implicit user feedback for item recommendation instead of rating prediction.

User Modeling in Text

Apart from the Recommendation Systems that work on numeric ratings, there have been some research work to model user preferences as expressed in their review texts. For incorporating the author information in a topic model, the User-Sentiment topic model [137] considers the topic-sentiment distribution of a review from the author perspective. However, it ignores the characteristics of the entity being reviewed. PDA-LDA [135] associates its Dirichlet prior distribution with user and item topic factors. The work in [130] models aspects and sentiments based on the demography of authors. However, such demographic information are not always available and it cannot model the bias or preference of an individual. In our proposed models we do not rely on the availability of further demographic information for a user and rather rely on their past interactions with other items to model their biases.

2.3 Information Mining from Online UGC

Credibility Analysis

Online UGC come with a pertinent credibility concern. There has been extensive research work in the domain of binary truth discovery for resolving conflicting data from multiple sources [57]. They have evolved from determining truth values of structured factual claims [132, 82, 58] to identifying veracity of unstructured textual claims over the web [94, 139, 78] in recent years. These approaches consider the linguistic cues in the claim statement and credibility of the sources to make an assessment. However, the focus of these works is determining a binary truth (true vs. false) about an entity or story and they require expert knowledge bases to resolve the same.

Opinion Mining

The study of subjectivity of user feedback is related to the research on opinion mining from reviews and ratings. There has been substantial research to mine online reviews using topic models [83, 120, 59, 39, 75, 12]. The Topic Aspect Model (TAM) [83] jointly discovers aspects and topics from documents. The aspect and topic are independent and each aspect affects all topics in similar manner. However, in reviews, the topics discussed are often closely related to an aspect. JTV [120] encodes topic-viewpoint dependency, but assumes that a document contains only one aspect. MC-LDA [12] employs must-link and cannot-link constraints to extract coherent aspects but does not consider the sentiment polarity of words. JST [59] assumes that there is a single sentiment polarity for a review and the topics are chosen conditioned on that, while ASUM in [39] assumes that all words in a sentence are associated with the same topic and sentiment. However, in a realistic scenario sentiments may vary depending on the topics discussed in a review, e.g., an author might like the *location* of a hotel but not the *service* of the staff. We try to incorporate such dependencies when encoding opinions in our model. There has also been some work that uses supervision from aspect ratings to model review text like [119, 63, 125]. In contrast our model for opinion mining from text can learn the latent aspects and their sentiments

in an unsupervised manner.

Online Health Information

With people turning to online communities to share their health related information, many research works have focused on detecting Adverse Drug Events or side effects mentions from post text [136, 79, 40, 22, 88, 55, 31, 114] to augment existing medical knowledge base. However, there has been increasing concerns regarding the credibility of such online medical claims regarding whether mentioned side effects are truly due to the drug or not [14, 86]. Recently a few machine learning based solutions have been developed to identify trustworthy textual claims made in online health communities. A few frameworks [81, 107] have been developed that automatically assess new health information with the help of reliable knowledge in external health websites. In [56] the authors enhance the framework of truth discovery from multiple sources to incorporate the semantic meaning of a post text and aim to identify credible claims. In [76] the authors develop a probabilistic graphical model to infer the credibility of a user statement regarding side effects of a drug by jointly inferring user trustworthiness and language objectivity of the textual post. These works also fall into the line of finding a single truth value for a claim. In contrast, we also appreciate the fact that different people may genuinely experience different outcomes from the same treatment due to confounding factors like other concurrent treatments or their medical conditions and so on. Therefore, it is necessary to model the patient experiences as a personalized prediction problem in order to understand the treatment outcomes for different patients.

Social Media

Detecting misinformation on social media has received attention from the research community recently. Research on determining rumor veracity on social media have utilized hand-crafted features such as posting and re-tweeting behavior, textual content and links to external sources [9, 129, 115, 62]. In the recent SemEval 2017 Challenge [18], many have used hand crafted feature-based approaches to tackle the task of rumor detection in

conjunction with stance prediction [112, 109, 122, 23]. Several works have examined using propagation patterns to detect rumors [53, 52, 66]. The cascading spread of misinformation in Facebook through photos and their captions, have been studied by analyzing comments linking to rumor debunking websites [26]. In [53, 52], a time-series model captures the periodic bursts in volume particular to false rumors whereas [66] use tree kernels to capture the propagation pattern. The work in [138] considers the enquiring reactions of people to detect rumours. However, they use a handful of cue terms such as 'not true', 'unconfirmed' or 'debunk' to find questioning and denying tweets. [64] employ Hawkes process to use both stance and temporal information of tweets but disregard their conversation structure.

Advances in deep learning have motivated researchers to explore solutions for the rumor debunking problem using recurrent neural networks [65] and convolutional neural networks [134]. [65] use the temporal sequence of tweets as a variable length time series and represent them using stacked Gated Recurrent Units (GRU) [13]. [134] use CNN instead of GRU for the task. These deep learning based methods outperform hand-crafted feature based methods due to their ability to model higher dimensional complex interactions between the underlying features. However, none of them utilizes the conversational context of tweets to analyze the stances of people towards a rumor and determine its veracity.

Chapter 3

Improving Usability of Ratings: Quantifying Aspect Bias

3.1 Introduction

In this chapter, we focus on user feedback in forms of ratings for different aspects of items. With explosive growth and easy availability of information on the web, we base our purchasing decisions increasingly on the opinion of others while knowing next to nothing about them. Before visiting a restaurant, if we ask multiple friends of ours for their opinion, we might end up with mixed reactions. Since we know them individually, we know their tastes and expectations, so we know whose raves or rants are to be taken with a pinch of salt. However, presented with an overwhelming number of opinions from strangers for an item, it is difficult to know whom to trust and how much. Each individual user is different and rate the same item differently depending on his/her expectations from the item. While we try to judge the quality of an item based on its ratings given by different people, we are unaware of these underlying biases of people that influence those ratings.

An item typically has many aspects and not all aspects are equally important to every user. To some user, the *cleanliness* of a hotel is most important and he/she tends to rate this aspect stringently, but is lenient when rating *food* or *amenities*. Other users may have a different set of preferences and their aspect ratings for the same item could be vastly

different. Hence individual biases determine the ratings we observe, making it difficult to interpret conflicting ratings without knowing the underlying user biases. Therefore, the confounding factor that we wish to uncover in this chapter is the aspect biases of users, that influence their ratings and obfuscate the true quality of an item.

An item's quality is often estimated by considering a simple average of all user ratings. However, such an average is an inadequate estimation of the true quality of an item, given the varied biases of users. For an item with only a few ratings this is aggravated, since even its average ratings are highly susceptible to those few users' biases. In order to reliably estimate the quality of an item and to make an informed purchase decision, it is important for a person to be aware of the latent aspect biases of users that affect their ratings. To properly interpret an individual's rating and the quality of an item, one needs to look at (i) how the individual rated other items on the same aspect to gauge his/her aspect bias, as well as, (2) how other users rated that particular item's aspect, to estimate its true quality. This requires modeling of both the user and the item simultaneously in a unified framework.

There have been many works on rating prediction in the past decade [54, 49, 35, 68, 87, 106, 105, 85]. Most of them are based on collaborative filtering approaches where deviation terms are used to account for rating bias of a user and item. For the task of aspect rating prediction even though one can train multiple models for each aspect, they are unable to capture the correlation between aspects. For example, a person who is fussy about *cleanliness* of a hotel, is more likely to be choosy about *room* than the *location* of a hotel. Recently, researchers have studied the prediction of latent aspect ratings using review texts associated with the ratings [125, 123]. However, such descriptive texts are not always available, and even if they are, they do not comment on each aspect individually making the aspect rating prediction task hard.

We propose a unified probabilistic model to quantify the underlying user biases for different aspects that lead to the observed aspect ratings for different items. We directly model the correlation between aspects by allowing a covariance structure among them. This models a realistic scenario where a user's bias, and in turn her rating of one aspect,

Figure 3.1: Sample restaurant's ratings with color coded user aspect bias shown beside the ratings.

may be correlated with another aspect. We show that it is possible to detect the underlying aspect biases of individual users that are consistent across their ratings on different items. Typically such a model will require a lot of ratings from an individual user to learn his/her bias properly. This requirement can be limiting in an e-commerce set-up where the sparsity of datasets are generally very high. In our model, we mitigate this by exploiting the similarity between rating characteristics of different users. We can learn the aspect bias of users even with few ratings, by introducing latent user groups, based on the rating behavior of users on various aspects. For example, in the domain of hotels, one of the groups might generally give low ratings for *cleanliness* while another user group gives higher ratings for *food*.

Figure 3.1 shows an example application of the model where the learned user aspect bias is displayed beside the ratings. People with a negative bias tend to be more critical about the aspect and generally underrate the aspect than other users, whereas people with a positive bias for an aspect tend to overrate it. Knowing the aspect biases of individuals, other users can better interpret their ratings. For a person looking at the ratings of a hotel in order to decide whether to make a reservation, it might be helpful to know the

aspect preferences of users and she can decide to rely more on the ratings of users whose preferences resonate with her own. Furthermore, this is beneficial for service providers to focus on improving the aspects of an item that consumers *truly* care about.

Another technical limitation of previous works on modeling user ratings is a fundamentally wrong assumption that observed ratings are continuous [54, 49, 35, 68, 87, 106, 105, 85]. Whereas, in reality most observed ratings in e-commerce websites are ordinal in nature. There have been very few attempts to address the ordinal nature of ratings. A model using regression is developed in [113] to handle ordinal ratings as a special case. The work in [50] proposes a *wrapper* around a CF method for ordinal data. Both of these works use a logit model for ordinal regression. In contrast, most statistical approaches handle ordinals using ordinal probit model [1, 104, 77]. Although these approaches allow Bayesian inference, it necessitates using truncated Gaussian distributions and forced ordering of cut-off points. This leads to complicated and even sub-optimal inference.

Our model incorporates the ordinal nature of observed ratings through proper statistical formulation. This provides a better fit with the real world ratings data. However, modeling the ordinal nature of observed ratings as well the correlation between aspects introduce non-conjugacy into our model, making Bayesian inference very challenging.

To eliminate the non-conjugacy of Gaussian prior-Categorical likelihood, we utilize stick-breaking formulation with Pólya-Gamma auxiliary variable augmentation. The construction proposed in the paper is efficient and generic. It will help developing inference mechanisms for various applications that need to model ordinal data in terms of continuous latent variables with a correlation structure.

Experiments on two real world datasets from TripAdvisor [1] and OpenTable [2] demonstrate that the proposed model provides new insights in users' rating patterns, and outperforms state-of-the-art methods for aspect rating prediction.

To the best of our knowledge, this is the first work to model ordinal aspect ratings parameterized by latent multivariate continuous responses, with a simple, scalable and fully Bayesian inference.

[1] `https://www.tripadvisor.com`
[2] `https://www.opentable.com`

3.2 Ordinal Aspect Bias Model

In this section, we describe the design of our Ordinal Aspect Bias model and present a Bayesian approach for inference.

Suppose we have J users and I items. A user can give ratings on A aspects of the item. Let R be the set of observed ratings where $\mathbf{r_{ij}}$ is an A dimensional vector denoting the rating of user j for item i on each of its aspects. Each $\mathbf{r_{ij}}$ is a discrete value between 1 and K corresponding to a K-level scale (*poor* to *excellent*). We assume that $\mathbf{r_{ij}}$ arises from a latent multivariate continuous response $\mathbf{v_{ij}}$ which is dependent on two factors : (i) the intrinsic quality of the item on the aspect and (ii) the bias of the user for the aspect.

The intrinsic quality of an item $\mathbf{z_i}$ is an A dimensional vector, drawn from a multivariate normal distribution, with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. We use multivariate normal distribution to account for the correlation among the subsets of aspects of an item. For example, it is highly unlikely for a hotel to have excellent *room* quality but very poor *cleanliness*, but it is possible to have a good *location* and average *food* choices. Such correlations among subset of aspects are captured by the covariance matrix. The parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ are given a conjugate normal-inverse Wishart (NIW) prior.

The preference of a user for an aspect is captured by a bias vector $\mathbf{m_g}$ of dimension A. If a user places great importance on a particular aspect (e.g. *cleanliness*), this will be reflected in his ratings across all hotels. In other words, her rating on the *cleanliness* aspect will tend to be lower than the majority's rating for *cleanliness* on the same hotel. We cluster users with similar preferences into different user groups and associate a bias vector $\mathbf{m_g}$ with each group. The membership of a user j in a user group is denoted as s_j where s_j is drawn from a categorical distribution θ with a Dirichlet prior parameter α.

Given the intrinsic quality $\mathbf{z_i}$ and bias $\mathbf{m_g}$, the latent response $\mathbf{v_{ij}}$ is drawn from a multivariate Gaussian distribution with $\mathbf{z_i} + \mathbf{m_{s_j}}$ as mean and a hyper-parameter $\mathbf{B}$ as covariance. The formulation of the mean is intuitive - a user's response depends on the item's intrinsic quality for an aspect offset by his/her own bias. Here we have used an additive bias, experimenting with other forms would be a possible direction for future

exploration.

Given the latent response $\mathbf{v_{ij}}$, we need to sample the observed rating vector $\mathbf{r_{ij}}$. Note that since the observed ratings are ordered and discrete, they should be drawn from a categorical distribution. However, the latent response $\mathbf{v_{ij}}$ is given a multivariate Gaussian prior. In order to have a fully Bayesian inference, we need to *transform* this categorical distribution to a Gaussian form to exploit conjugacy. This is the central technical challenge for our proposed model.

The two most popular choices for such parameterizations are - ordered logit and ordered probit models. While both have received wide coverage, inferences for them remain inefficient [43]. We propose an alternative here. We develop a stick-breaking mechanism with logit function to map the categorical likelihood to a binomial form. Thereafter, leveraging the recently developed Pólya-Gamma auxiliary variable augmentation scheme [93], the binomial likelihood is transformed to Gaussian, thus establishing conjugacy and enabling an effective posterior inference.

These two concepts have been sparsely explored in the literature before, but, independently. The authors of [121] used stick-breaking formulation to parameterize the underlying continuous rating. However, since the non-conjugacy challenge remained, it made an MCMC sampling non-trivial and they performed an approximate variational Bayesian inference. For correlated topic models [10], Pólya-Gamma auxiliary variable augmentation is used with logistic-normal transformation. None of these works use stick-breaking likelihood with Pólya-Gamma variable augmentation to exploit conjugacy to facilitate Gibbs sampling.

The technical aspects of of our model construction are described in detail in the following sections. The generative process of the model is as follows:

1. Draw a multinomial group distribution θ from Dirichlet (α).
2. For each group $g \in 1, \cdots, G$ draw a bias offset $\mathbf{m_g}$ from $N_A(0, \Lambda)$
3. For each user $j \in 1, \cdots, J$, sample a group s_j from Cat (θ)
4. For each item $i \in 1, \cdots, I$, sample an intrinsic rating $\mathbf{z_i}$ from $N_A(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
5. For each rating $\mathbf{r_{ij}} \in R$

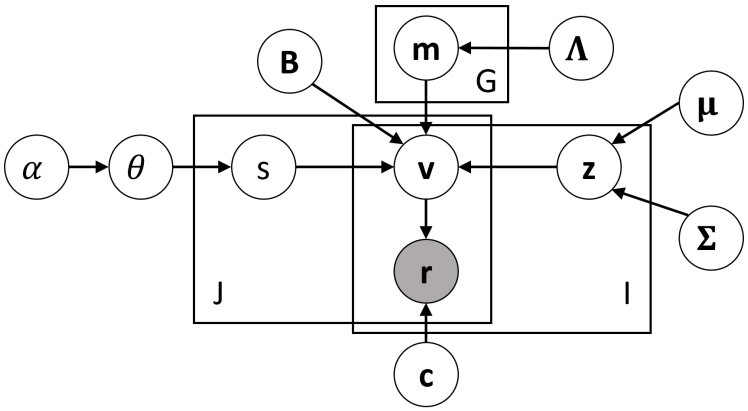


Figure 3.2: Ordinal Aspect Bias Model

(a) draw latent continuous rating $\mathbf{v_{ij}}$ from $N_A(\mathbf{z_i} + \mathbf{m_{s_j}}, \mathbf{B})$

(b) draw observed ordinal rating $\mathbf{r_{ij}}$ from Cat $(SB(\mathbf{v_{ij}}, \mathbf{c}))$

where $SB(\mathbf{v_{ij}}, \mathbf{c})$ refers to the stick-breaking parametrization of the continuous response $\mathbf{v_{ij}}$ using cut-points $\mathbf{c}$. Figure 3.2 shows the proposed graphical model using plate notation.

3.2.1 Stick-Breaking Likelihood

We first discuss how to map the categorical likelihood of $\mathbf{v_{ij}}$, denoted as $Lik(\mathbf{v_{ij}})$, to a binomial form. Let r_{ija} denote the observed ordinal rating of item i, by user j on aspect a, and is drawn from a categorical distribution over K categories. Since the categories are ordered, we utilize a stick-breaking parameterization for the probabilities $P(r_{ija} = k)$ where $k \in \{1, \cdots, K\}$.

Suppose we have a unit length stick where the continuum of points on this stick represents the probability of an event occurring. If we break this stick at some random point p, then we have a probability mass function over two outcomes (with probabilities p and $1 - p$). By breaking the stick multiple times, we obtain a probability mass function over multiple categories. Figure 3.3 shows an illustration of this. If we consider four ordered categories, the Stick-Breaking Likelihood can be written as:

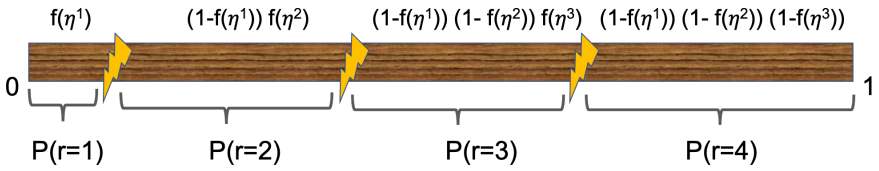


Figure 3.3: Illustration of Stick Breaking Process with a unit length stick.

$$P(r = 1|\boldsymbol{\eta}) = f(\eta^1) \tag{3.1}$$

$$P(r = 2|\boldsymbol{\eta}) = (1 - f(\eta^1))f(\eta^2) \tag{3.2}$$

$$P(r = 3|\boldsymbol{\eta}) = (1 - f(\eta^1))(1 - f(\eta^2))f(\eta^3) \tag{3.3}$$

$$P(r = 4|\boldsymbol{\eta}) = (1 - f(\eta^1))(1 - f(\eta^2))(1 - f(\eta^3)) \tag{3.4}$$

where $P(r = k)$ represents the probability of an observed rating being k and $\boldsymbol{\eta}$ is a function of the data, and $f()$ is a function to map η^k's value between $[0, 1]$. Following the stick breaking principle, we parameterize the probability of each ordinal rating r_{ija} being assigned the categorical value k i.e. $P(r_{ija} = k)$, using a function of the covariate η^k_{ija}. We define η^k_{ija} as:

$$\eta^k_{ija} = c_k - v_{ija} \tag{3.5}$$

where $\mathbf{c} = \{c_1, \cdots, c_{K-1}\}$ is a cut-point vector with $c_1 < c_2 < \cdots < c_{K-1}$. These represent the boundaries between the ordered categories. Thus η^k_{ija} represents a discriminative mapping of the underlying continuous rating v_{ija} onto the k dimensional ordinal space, using cut-point vector c.

Next, the probability of observing the vector of ratings $\mathbf{r_{ij}}$ is defined as a product of probabilities of observing each of the aspect ratings r_{ija} given the values of $\boldsymbol{\eta_{ija}}$, i.e.

$$P(\mathbf{r_{ij}}|\boldsymbol{\eta_{ija}}) = \prod_{a=1}^{A} P(r_{ija}|\boldsymbol{\eta_{ija}}) \tag{3.6}$$

Hence the likelihood of $\mathbf{v_{ij}}$ is:

$$Lik(\mathbf{v_{ij}}) = P(\mathbf{r_{ij}}|\mathbf{v_{ij}}, \mathbf{c}) = P(\mathbf{r_{ij}}|\boldsymbol{\eta_{ij}}) = \prod_{a=1}^{A} P(r_{ija}|\boldsymbol{\eta_{ija}}) \tag{3.7}$$

To squash $\boldsymbol{\eta_{ija}}$ within [0,1] we use a sigmoid function on it, denoted by $f(x) = \frac{e^x}{1+e^x}$. Sigmoid function enables us to use Pólya-Gamma augmentation scheme to handle the non-conjugacy subsequently.

For identifiability, we set $f(\eta_{ija}^K) = 1$. Generalizing from equation 3.1 to 3.4, the stick-breaking likelihood can be written as:

$$P(r_{ija} = k) = \prod_{k'<k} (1 - f(\eta_{ija}^{k'}))f(\eta_{ija}^{k}) \tag{3.8}$$

By encoding the discrete rating r_{ija}, with a 1-of-K vector $\mathbf{x_{ija}}$ where

$$x_{ija}^k = \begin{cases} 1 & \text{if } r_{ija} = k \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

we now rewrite the likelihood of $\mathbf{v_{ij}}$ (equation 3.7) in binomial terms:

$$Lik(\mathbf{v_{ij}}) = \prod_{a=1}^{A} P(r_{ija}|\boldsymbol{\eta_{ija}}) = \prod_{a=1}^{A} P(\mathbf{x_{ija}}|\boldsymbol{\eta_{ija}}) = \prod_{k=1}^{K-1} Binom(x_{ija}^k|N_{ija}^k, f(\eta_{ija}^k)) \tag{3.10}$$

where

$$N_{ija}^k = 1 - \sum_{k'<k} x_{ija}^{k'}$$

3.2.2 Pólya-Gamma Variable Augmentation

Next, we explain how to transform the above binomial likelihood to a Gaussian form via Pólya-Gamma (PG) auxiliary variable augmentation scheme. The integral identity at the heart of the PG augmentation is:

$$\frac{(e^\psi)^a}{(1+e^\psi)^b} = 2^{-b} e^{\kappa\psi} \int_0^\infty e^{-\omega\psi^2/2} p(\omega) d\omega \tag{3.11}$$

where $\kappa = a - b/2$, $b > 0$ and $\omega \sim PG(b, 0)$.

By expanding the binomial likelihood in Eqn. 3.10, we get

$$P(\mathbf{x_{ija}}|\boldsymbol{\eta_{ija}}) = \prod_{k=1}^{K-1} \binom{N_{ija}^k}{x_{ija}^k} (f(\eta_{ija}^k))^{x_{ija}^k} (1 - f(\eta_{ija}^k))^{N_{ija}^k - x_{ija}^k} \tag{3.12}$$

By replacing $f(\eta_{ija}^k)$ with the sigmoid function, we get

$$P(\mathbf{x_{ija}}|\boldsymbol{\eta_{ija}}) = \prod_{k=1}^{K-1} \binom{N_{ija}^k}{x_{ija}^k} \frac{(e^{\eta_{ija}^k})^{x_{ija}^k}}{(1 + e^{\eta_{ija}^k})^{N_{ija}^k}} \tag{3.13}$$

The above equation is in similar form as the left hand side of PG augmentation scheme (equation 3.11). Hence using the integral identity of PG augmentation, we can now rewrite the categorical likelihood of $\mathbf{v_{ij}}$ as:

$$Lik(\mathbf{v_{ij}}) = \prod_{a=1}^{A} P(\mathbf{x_{ija}}|\boldsymbol{\eta_{ija}}) \tag{3.14}$$

$$\propto \prod_{a=1}^{A} \prod_{k=1}^{K-1} e^{\kappa_{ija}^k \eta_{ija}^k} \int_0^\infty e^{-\omega_{ija}^k (\eta_{ija}^k)^2/2} p(\omega_{ija}^k) d\omega_{ija}^k \tag{3.15}$$

where $\kappa_{ija}^k = x_{ija}^k - N_{ija}^k/2$, $\psi_{ija}^k = \eta_{ija}^k$ and $p(\omega_{ija}^k)$ is $PG(N_{ija}^k/2, 0)$ independent of ψ_{ija}^k.

By the exponential tilting property of PG distribution, we can draw the auxiliary variable as

$$\omega_{ija}^k \sim PG(N_{ija}^k, \eta_{ija}^k) \tag{3.16}$$

Conditioning on $\boldsymbol{\omega_{ij}}$, $Lik(\mathbf{v_{ij}})$ can be transformed to a Gaussian form:

$$
\begin{aligned}
Lik(\mathbf{v_{ij}}) &\propto \prod_{k=1}^{K-1}\prod_{a=1}^{A} e^{\kappa_{ija}^k \eta_{ija}^k} e^{-\omega_{ija}^k (\eta_{ija}^k)^2/2} \\
&\propto \prod_{k=1}^{K-1}\prod_{a=1}^{A} exp\{\kappa_{ija}^k (c_k - v_{ija}) - \omega_{ija}^k (c_k - v_{ija})^2/2\} \\
&\propto \prod_{k=1}^{K-1}\prod_{a=1}^{A} exp\{-\omega_{ija}^k ((c_k - v_{ija}) - \frac{\kappa_{ija}^k}{\omega_{ija}^k})^2\} \\
&\propto \prod_{k=1}^{K-1} exp\{-\frac{1}{2}(\frac{\boldsymbol{\kappa_{ij}^k}}{\boldsymbol{\omega_{ij}^k}} - (\mathbf{c_k} - \mathbf{v_{ij}}))^T \boldsymbol{\Omega_{ij}^k} (\frac{\boldsymbol{\kappa_{ij}^k}}{\boldsymbol{\omega_{ij}^k}} - (\mathbf{c_k} - \mathbf{v_{ij}}))
\end{aligned}
\tag{3.17}
$$

where $\boldsymbol{\kappa_{ij}^k}, \boldsymbol{\omega_{ij}^k}$ are vectors of dimension A, $\boldsymbol{\Omega_{ij}^k}$ is a diagonal matrix of $(\omega_{ij1}^k, \omega_{ij2}^k, \cdots, \omega_{ijA}^k)$.

Here, we assume the values in the A-dimensional cut-point vector $\mathbf{c_k}$ are all equal to c_k. In practice, if we need different cut-points for different aspects, $\mathbf{c_k}$ can be set accordingly.

3.2.3 Bayesian Inference

As we have transformed the likelihood to a Gaussian form, we now proceed to present a Gibbs sampler for a fully Bayesian MCMC inference with exact sampling. We describe the sampling of all our latent variables, user groups $\mathbf{s}$, bias offset of user groups $\mathbf{m}$, intrinsic ratings $\mathbf{z}$, cut-points $\mathbf{c}$ and latent continuous ratings $\mathbf{v}$. For faster mixing rates, we first integrate out the group distribution by exploiting the Dirichlet-Multinomial conjugacy. We factor the joint probability of these variables as:

$$
P(\mathbf{r}, \mathbf{v}, \mathbf{m}, \mathbf{z}, \mathbf{s}, \mathbf{c}) = P(\mathbf{r}|\mathbf{v}, \mathbf{c}) P(\mathbf{v}|\mathbf{m}, \mathbf{z}, \mathbf{s}) P(\mathbf{c}) P(\mathbf{z}) P(\mathbf{s}) P(\mathbf{m})
$$

Sampling Bias Offset of User Groups. For each user group g, we sample its bias offset $\mathbf{m_g}$ from the Gaussian posterior:

$$
P(\mathbf{m_g}|\boldsymbol{\Lambda}, \mathbf{v}, \mathbf{z}) \propto P(\mathbf{m_g}|\boldsymbol{\Lambda}) \prod_{j \in J[g]} \prod_{i \in I[j]} P(\mathbf{v_{ij}}|\mathbf{m_g}, \mathbf{z_i}, \mathbf{B})
$$

where $J[g]$ is the set of users belonging to group g and $I[j]$ is the subset of items rated

by user j.

Since the prior is a multivariate Gaussian $N_A(0, \Lambda)$ and the observations $\mathbf{v_{ij}}$ are also drawn from a multivariate Gaussian $N_A(\mathbf{z_i} + \mathbf{m_g}, \mathbf{B})$, the posterior of $\mathbf{m_g}$ is given by a Gaussian $N_A(\hat{\mathbf{m}}_\mathbf{g}, \hat{\boldsymbol{\Lambda}}_{\boldsymbol{g}})$ with

$$\begin{aligned} \hat{\mathbf{m}}_\mathbf{g} &= \hat{\boldsymbol{\Lambda}}_{\boldsymbol{g}}(\mathbf{B}^{-1} \sum_{j \in J[g]} \sum_{i \in I[j]} (\mathbf{v_{ij}} - \mathbf{z_i})) \\ \hat{\boldsymbol{\Lambda}}_{\boldsymbol{g}} &= (n_g \mathbf{B}^{-1} + \boldsymbol{\Lambda})^{-1} \end{aligned}$$

where n_g is the total number of ratings observed for users belonging to group g.

Sampling User Groups. We integrate out the group distribution θ by exploiting Dirichlet-Multinomial conjugacy, and sample the group of each user j as:

$$P(s_j|\alpha, \mathbf{m}, \mathbf{v}) \propto P(s_j|\alpha) \prod_{i \in I[j]} P(\mathbf{v_{ij}}|\mathbf{m_{s_j}}, \mathbf{z_i}, \mathbf{B})$$

where $I[j]$ are the subset of items rated by user j, the prior $P(s_j|\alpha)$ is given by the Dirichlet distribution. The likelihood is the multinomial distribution given by the probability of observing all the ratings of the user j given bias m_{s_j}.

Sampling Intrinsic Ratings. Similar to the bias offsets of user groups, we sample intrinsic rating $\mathbf{z_i}$ of each item i from a Gaussian distribution $N_A(\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{i}})$ where

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{\boldsymbol{i}} &= \hat{\boldsymbol{\Sigma}}_{\boldsymbol{i}}(\mathbf{B}^{-1} \sum_{j \in J[i]} (\mathbf{v_{ij}} - \mathbf{m_{s_j}}) + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}) \\ \hat{\boldsymbol{\Sigma}}_{\boldsymbol{i}} &= (n_i \mathbf{B}^{-1} + \boldsymbol{\Sigma})^{-1} \end{aligned}$$

where n_i is the total number of ratings observed for item i and $J[i]$ is the subset of users who have rated item i. The prior parameters $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ of the intrinsic ratings are given

a conjugate Normal-Inverse Wishart (NIW) prior and sampled.

Sampling Latent Continuous Ratings. The latent continuous ratings, $\mathbf{v_{ij}}$ have a Gaussian prior $N_A((\mathbf{z_i} + \mathbf{m_{s_j}}), \mathbf{B})$ and a categorical likelihood $P(r_{ija}|\ \mathbf{v_{ij}}, \mathbf{c})$. We have transformed the categorical likelihood to the conditional Gaussian form (recall Eqn. 3.17). The posterior can be formulated as:

$$
\begin{aligned}
P(\mathbf{v_{ij}}) &\propto P(\mathbf{v_{ij}}|\mathbf{m_{s_j}}, \mathbf{z_i}, \mathbf{B}) * Lik(\mathbf{v_{ij}}|\boldsymbol{\omega}, \mathbf{r_{ij}}, \mathbf{c}) \\
&\propto exp\{-\frac{1}{2}(\mathbf{v_{ij}} - (\mathbf{z_i} + \mathbf{m_{s_j}}))^T \mathbf{B}^{-1}(\mathbf{v_{ij}} - (\mathbf{z_i} + \mathbf{m_{s_j}}))\} \\
&* \prod_{k=1}^{K-1} exp\{-\frac{1}{2}(\frac{\boldsymbol{\kappa}_{ij}^{\boldsymbol{k}}}{\boldsymbol{\omega}_{ij}^{\boldsymbol{k}}} - (\mathbf{c_k} - \mathbf{v_{ij}}))^T \boldsymbol{\Omega}_{ij}^{\boldsymbol{k}}(\frac{\boldsymbol{\kappa}_{ij}^{\boldsymbol{k}}}{\boldsymbol{\omega}_{ij}^{\boldsymbol{k}}} - (\mathbf{c_k} - \mathbf{v_{ij}}))\}
\end{aligned}
$$

Since both the prior and likelihood are now Gaussian, we have the following Gibbs sampler:

$$
\begin{aligned}
\mathbf{v_{ij}} &\sim N_A(\boldsymbol{\mu}_{ij\omega}, \boldsymbol{\Sigma}_{ij\omega}) \\
\omega_{\mathbf{ija}} &\sim PG(\mathbf{N_{ija}}, \mathbf{v_{ija}} - \mathbf{c})
\end{aligned}
$$

where

$$
\begin{aligned}
\boldsymbol{\mu}_{ij\omega} &= \mathbf{B^{-1}}(\mathbf{z_i} + \mathbf{m_{s_j}}) + \sum_{\mathbf{k=1}}^{\mathbf{K-1}} \boldsymbol{\Omega}_{ij}^{\boldsymbol{k}}(\mathbf{c_k} - \frac{\boldsymbol{\kappa}_{ij}^{\boldsymbol{k}}}{\boldsymbol{\omega}_{ij}^{\boldsymbol{k}}}) \\
\boldsymbol{\Sigma}_{ij\omega} &= \mathbf{B^{-1}} + \sum_{\mathbf{k=1}}^{\mathbf{K-1}} \boldsymbol{\Omega}_{ij}^{\boldsymbol{k}}
\end{aligned}
$$

Sampling Cut-Points. Sigmoid function in the stick-breaking formulation allows us to sample cut-points while ensuring their relative order without additional constraints. Figure 3.4 shows probability distributions for simulated cut-points. For each category, the $xaxis$ denotes the latent continuous ratings and the $yaxis$ denotes probability of belonging to that category.

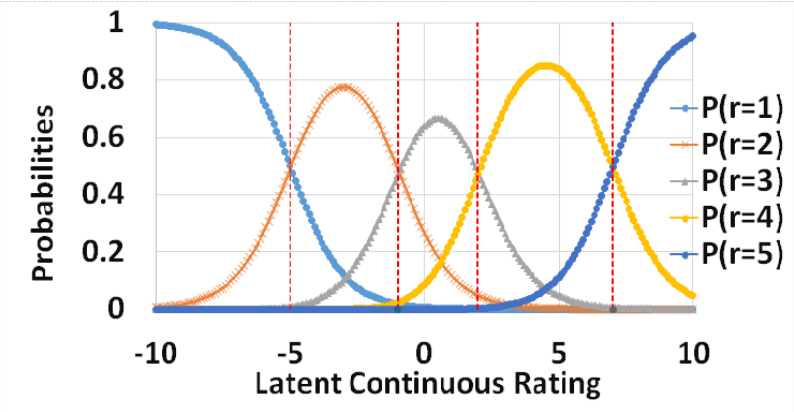


Figure 3.4: Category probabilities for cut-points (-5,-1,2,7)

The following lemma gives the relationship between cut-points, latent continuous ratings, and the observed ratings.

Lemma 3.2.1. *If* $v_{ija} > c_k - ln\ (1 - e^{-(c_{k+1}-c_k)})$, *then* $P(r_{ija} = k+1) > P(r_{ija} = k)$.

Proof. Let $\delta_k \geq -\ ln\ (1 - e^{-(c_{k+1}-c_k)})$. By replacing v_{ija} with $(c_k + \delta_k)$ in Eqn. 2, we have

$$
\begin{aligned}
P(r_{ija} = k) &= \prod_{q<k}(1 - f(c_q - c_k - \delta_k))(f(c_k - c_k - \delta_k)) \\
&= \prod_{q<k}(1 - f(c_q - c_k - \delta_k))(f(-\delta_k)) \\
P(r_{ija} = k+1) &= \prod_{q<k}(1 - f(c_q - c_k - \delta_k))(1 - f(-\delta_k))(f(c_{k+1} - c_k - \delta_k))
\end{aligned}
$$

Taking the ratio, we have

$$
\begin{aligned}
\frac{P(r_{ija} = k+1)}{P(r_{ija} = k)} &= \frac{(1 - f(-\delta_k))(f(c_{k+1} - c_k - \delta_k))}{f(-\delta_k)} \\
&= (\frac{e^{\delta_k}}{1 + e^{\delta_k}} * \frac{1}{1 + e^{c_k + \delta_k - c_{k+1}}})/(\frac{1}{1 + e^{\delta_k}}) \\
&= \frac{e^{\delta_k}}{1 + e^{c_k - c_{k+1} + \delta_k}}
\end{aligned}
$$

Since $\delta_k \geq -ln(1 - e^{-(c_{k+1}-c_k)})$, we see that $\frac{e^{\delta_k}}{1+e^{c_k - c_{k+1} + \delta_k}} > 1$. Hence, $P(r_{ija} =$

$k+1) > P(r_{ija} = k)$. □

We have shown that

$$P(r_{ija} = k+1) > P(r_{ija} = k) \text{ when } v_{ija} \geq (c_k - ln(1 - e^{-(c_{k+1} - c_k)})$$

Similarly,

$$P(r_{ija} = k) > P(r_{ija} = k-1) \text{ when } v_{ija} \geq (c_{k-1} - ln(1 - e^{-(c_k - c_{k-1})}))$$

This implies, when v_{ija} is within the range $(c_{k-1} - ln(1 - e^{-(c_k - c_{k-1})}), c_k - ln(1 - e^{-(c_{k+1} - c_k)})]$, then $P(r_{ija} = k)$ has the maximum probability over all other categories. In other words, for v_{ija} in the stated range, we have $\text{argmax}_{k'} P(r_{ija}|v_{ija}, k') = k$.

Hence, given the sampled values of v_{ija} we can constrain the possible set of values for the cut-points. We sample cut-point c_k from a uniform distribution within the range:

$$c_k \sim U[max\{v_{ija}| \operatorname*{argmax}_{k'} P(r_{ija}|v_{ija}, k') = k\} - ln(1 - e^{-(c_k - c_{k-1})}),$$
$$min\{v_{ija}| \operatorname*{argmax}_{k'} P(r_{ija}|v_{ija}, k') = k+1\} - ln(1 - e^{-(c_k - c_{k-1})})]$$

3.3 Experiments

For evaluation we use hotel ratings from TripAdvisor [125] and restaurant ratings from Opentable.com [3]. TripAdvisor lets its users rate a hotel in multiple aspects, namely, Service, Value, Room, Location. In OpenTable a user can rate each restaurant on Ambience, Food, Service and Value. From OpenTable.com we gathered all the restaurant ratings in New York Tri-State area. Table 3.1 shows the details of the datasets. The "#Items" , "#Users" and "#Ratings" columns are the number of users, items and ratings respectively, in each of the datasets.

[3] www.opentable.com

Dataset	# Items	# Users	# Ratings	Aspects
TripAdvisor	12,773	781,403	1,621,956	Service, Value, Room, Location
OpenTable	2805	1997	73,469	Ambience, Food, Service, Value

Table 3.1: Statistics of experimental datasets.

3.3.1 Rating Prediction

One application of Ordinal Aspect Bias model is predicting observed aspect ratings. We perform five-fold cross validation on user-item pairs, and take expected value of an aspect rating as the predicted rating. Note that all the aspect ratings for the same user-item pair will be in the same training or test set. By default, the number of user groups are set to 10.

For comparison, we first implemented the following baselines:

- **Continuous Aspect Bias model** is the continuous variant of our model where observed ratings are assumed to be continuous. Observed ratings are drawn from a (conjugate) multivariate Gaussian distribution, with mean as the true rating of the item offset with the bias of the user's group.
- **Ordinal** and **Continuous No Bias model** assume users are not biased. The observed ratings for an item are drawn from only the true rating of the item.
- **Ordinal** and **Continuous Global Bias model** assume all users have the same bias. All ratings for an item are drawn from the true rating of the item offset with a global bias.

For all the models, we infer the latent continuous rating in the test phase with Gibbs sampling using the other parameters learned during training phase. Since all these methods are generative models we use test set log likelihood as a measure of generalization power of the model. The higher the likelihood, the better is a model's generalization power on unseen data. We also use one of the most popular metric for rating prediction evaluation, namely, Root Mean Square Error (RMSE). RMSE measures the standard deviation of the prediction errors and is very commonly used for evaluation of regression analysis.

Model	TripAdvisor Data		OpenTable Data	
	log LL	RMSE	log LL	RMSE
Ordinal Aspect Bias	**-557.08**	**1.00**	**-493.79**	**1.03**
Continuous Aspect Bias	-1050.32	3.13	-560.14	2.21
Ordinal No Bias	-689.76	1.47	-546.25	1.95
Continuous No Bias	-1904.64	3.52	-651.16	2.39
Ordinal Global Bias	-2438.52	2.85	-570.28	2.37
Continuous Global Bias	-2632.95	3.91	-595.62	2.41

Table 3.2: Log likelihood and RMSE results on the test set. Log LL is higher the better, RMSE is lower the better. All comparisons are statistically significant (paired *t-test* with $p < 0.0001$).

Model	TripAdvisor Data							
	Service		Value		Room		Location	
	RMSE	FCP	RMSE	FCP	RMSE	FCP	RMSE	FCP
PMF	2.006	0.501	1.933	0.526	1.836	0.592	2.127	0.603
BPMF	1.414	0.586	1.373	0.571	1.314	0.614	1.209	0.651
URP	1.179	0.489	1.156	0.515	1.194	0.513	1.001	0.492
SVD++	**1.064**	0.578	1.079	0.562	1.093	0.639	0.894	0.665
BHFree	1.143	0.553	1.199	0.582	1.124	0.624	1.007	0.671
LARA	1.193	0.576	1.221	0.531	1.087	0.558	1.170	0.672
OrdRec + SVD++	1.348	0.619	1.344	0.613	1.359	0.654	1.173	0.702
AspectBias	1.067	**0.646***	**1.063***	**0.645***	**1.045**	**0.678***	**0.854***	**0.717**

Table 3.3: RMSE and FCP results for rating prediction on TripAdvisor dataset. RMSE values are the lower the better, FCP is higher the better. "*" denotes statistical significance with the runner up for $p < 0.005$

Table 3.2 shows mean log LL and RMSE of the competitive methods on test data. For both datasets Ordinal Aspect Bias model performs the best, demonstrating the need to consider both user's aspect bias and the proper ordinal nature of ratings. We note that as expected, the Global Bias and No Bias models perform the worst. This clearly demonstrates that users have distinct aspect preferences, rendering such a Global model insufficient. We also note that the ordinal models always outperform their continuous counterpart, thus proving the efficacy of modeling the proper nature of user ratings.

Next, we compare our proposed model with a number of state-of-the-art rating prediction models, namely, PMF [105], BPMF [106], URP [68, 3], SVD++ [49] and BHFree [87] and introduced in our background chapter (refer to Section 2.2). These are some of the most popular and competitive Collaborative Filtering (CF) based rating prediction algorithms. We used the implementation and best parameter settings published on Li-

Model	OpenTable Data							
	Ambience		Food		Service		Value	
	RMSE	FCP	RMSE	FCP	RMSE	FCP	RMSE	FCP
PMF	2.584	0.524	2.232	0.530	2.388	0.511	2.151	0.521
BPMF	1.154	0.490	0.992	0.532	1.426	0.498	1.302	0.519
URP	0.952	0.557	0.818	0.551	1.144	0.522	1.120	0.514
SVD++	**0.944***	0.525	0.831	0.544	**1.088**	0.544	1.131	0.517
BHFree	0.956	0.483	0.812	0.499	1.151	0.512	1.096	0.495
LARA	1.150	0.538	2.242	0.514	2.444	0.549	1.089	0.526
OrdRec + SVD++	1.337	0.672	1.121	0.613	1.533	0.618	1.521	0.623
AspectBias	0.953	**0.854***	**0.787***	**0.850***	1.134	**0.842***	**1.043***	**0.864***

Table 3.4: RMSE and FCP results for rating prediction on OpenTable dataset. RMSE values are the lower the better, FCP is higher the better. "*" denotes statistical significance with the runner up for $p < 0.005$

bRec.net website for these models. All these methods treat ratings as continuous values. We also compare with OrdRec [50] which can wrap collaborating filtering methods to tackle ordinal rating. These models do not consider the dependency between aspects and cannot predict multiple aspect ratings for a user-item pair. We therefore train them separately for each aspect. We further compare with LARA [125], which models latent aspect ratings using review texts. LARA considers the correlation between aspects.

For evaluation we use Root Mean Squared Error (RMSE) which is a popular metric of evaluation for rating prediction. Since RMSE cannot capture personalization or ordinal rating values, we also use FCP to measure the fraction of correctly ranked pair of items for each user. Table 3.3 and Table 3.4 shows the results for TripAdvisor and OpenTable dataset respectively.

We see that the proposed model outperforms state-of-the art methods in most cases. This is due to its ability of modeling aspect correlations and the ordinal nature of ratings which the other CF based methods are unable to leverage. Compared to the other method that models aspect correlations, LARA, we also note that AspectBias has significantly smaller predictive errors for all aspects. LARA relies on review texts to predict aspect ratings. Therefore, it requires a review text to mention all the aspects and authors sentiment for them to predict aspect ratings accurately, which is not always the case. For example, for hotel reviews people often describe their in-room experiences in detail, unlike the *service* or *location*, which explains comparatively better performance of LARA for the

Method	TripAdvisor Data	OpenTable Data
PMF	0.016	0.142
BPMF	0.219	0.133
URP	0.238	0.177
SVD++	0.364	0.201
BHFree	0.359	0.205
LARA	0.289	0.152
OrdRec + SVD++	0.148	0.262
OrdinalAspectBias	**0.404**	**0.298**

Table 3.5: Pearsons Correlation of aspect ranking

aspect *room*.

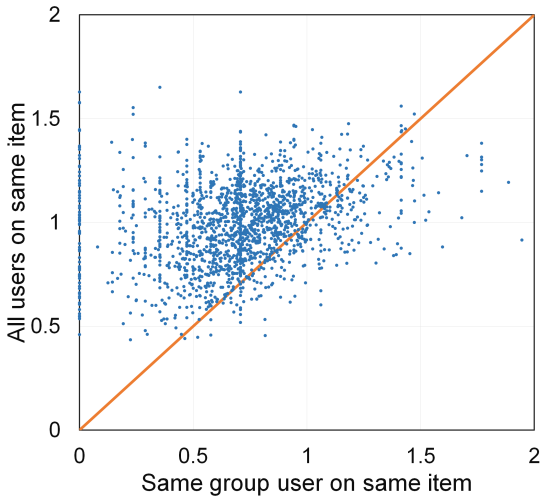
The relative ranking of aspects for a user-item pair is also important to understand which aspects of an item the user liked better. Table 3.5 shows the Pearson correlation coefficient of aspect ranking for a user-item pair, compared to its ground truth ranking. Clearly, Ordinal Aspect Bias model outperforms all other methods for the task of relative ranking of aspects. This validates that our model is able to learn aspect rating behavior of users accurately in order to achieve this prediction accuracy.

3.3.2 Evaluation of User Groups

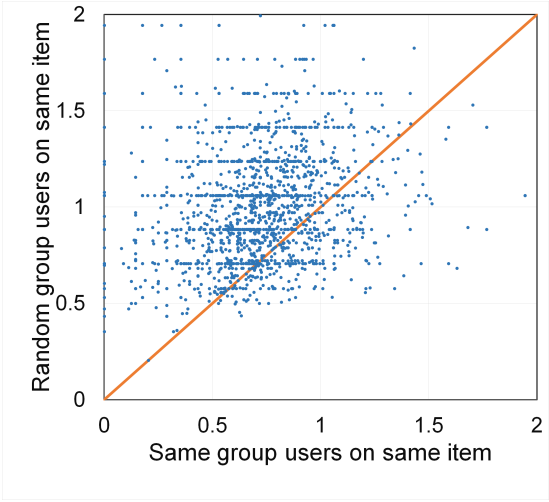
A significant advantage of our model is that it can infer latent user groups depending on their rating behaviors across multiple items. In this set of experiments, we evaluate whether the inferred latent user groups are meaningful. We show that if two users are assigned to the same group, then their ratings on the same items for the same aspects are indeed similar.

We perform a test similar to the work in [125]. We look at the standard deviation of the set of users belonging to a group who have rated the same item [125]. For each aspect of each item, we compare the standard deviation of ratings of each user group with that of two control groups : (i) of all users who have rated the item, and (ii)a random set of users who have rated the item. The size of the random set is kept the same as the size of the user group.

Figure 3.5 shows the scatter plots of the standard deviations for both datasets. In all figures we observe that most of the points lie above the line $y = x$, indicating that

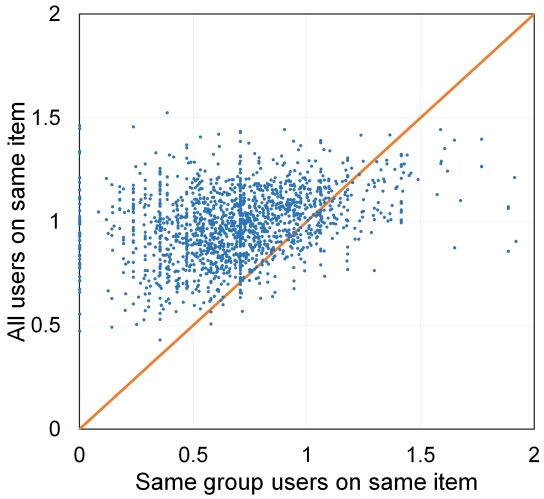


(a) User group vs. All users

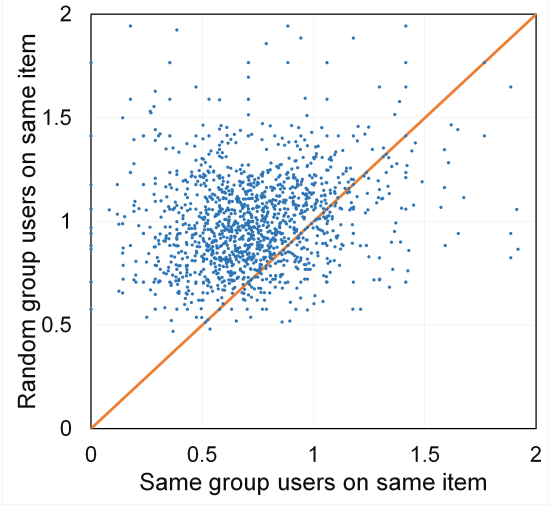


(b) User group vs. Random users

(i) TripAdvisor



(c) User group vs. All users

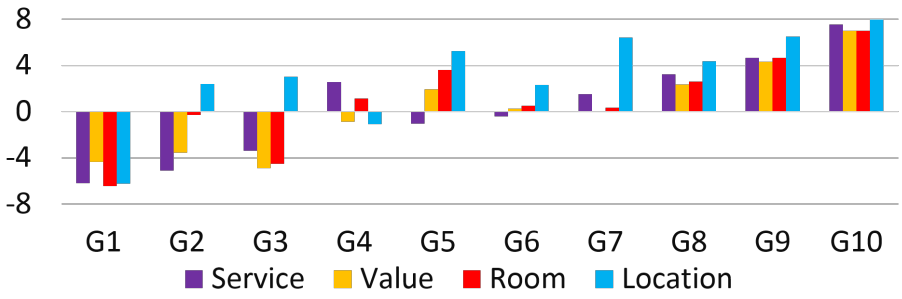


(d) User group vs. Random users

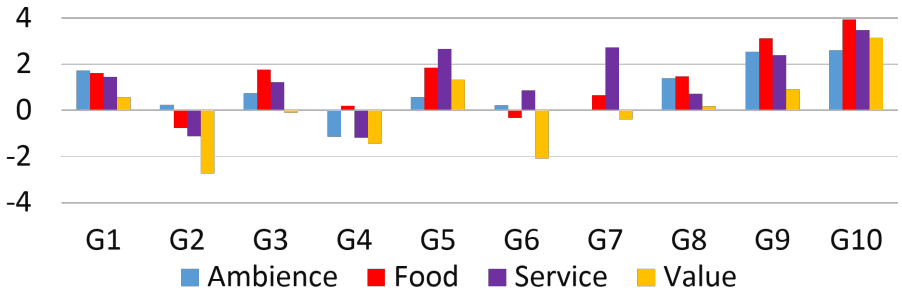
(ii) OpenTable

Figure 3.5: Scatter plot of standard deviations of aspect ratings.

users who belong to the same group have smaller standard deviation compared to the control group. This implies that the latent user groups obtained by the proposed model can effectively cluster users who give similar aspect ratings to the same item.



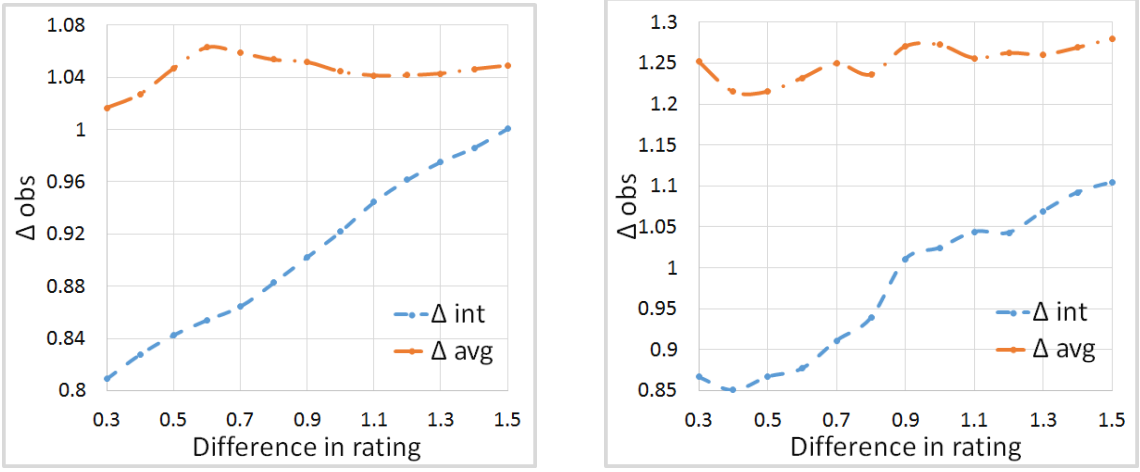
(a) TripAdvisor



(b) OpenTable

Figure 3.6: Mean bias value of user groups.

Figure 3.6 shows the mean ratings of 10 user groups after scaling the ratings to the range [-10, 10]. The TripAdvisor dataset captures the ratings of users for hotels worldwide. For the TripAdvisor dataset, we observe that the first user group seems to be quite critical whereas the last three user groups are positive. We also observe the correlation of aspect biases for different groups. For example, group 5 and 7 seem to have similar biases for *Room* and *Value* whereas group 4 is demanding about *Value* and *Location*. Considering all the ratings of the users belonging to group 5 and 7, we see that their ratings for *Value* are indeed most correlated with their ratings for *Room* than other aspects. On the other hand, for group 4 their ratings for *Value* are highly correlated with their ratings for *Location*. This suggests that for good *Value* for money, some users prefer good *Location* while some prioritize better *Room* quality and by modeling the covariance structure among aspects we are able to uncover such dependencies. Most of the user groups give positive ratings for the aspect *Location*. This can probably be attributed to the fact that



(a) OpenTable (b) TripAdvisor

Figure 3.7: Correlation with Δobs

they have done their research on the location of a hotel and have a fairly good idea about it before booking online. Thus, they are more likely to be satisfied with the location.

OpenTable contains ratings of restaurants in the New York TriState area. We can see from Figure 3.6b that users in group 4 who are particular about *Ambience* are also demanding about *Service* and *Value*.

3.3.3 Intrinsic Quality of Items

Often one forms a judgment about the quality of an item by the average rating it has received. However, if an item has received only a few ratings or is still in its early stages, it is difficult to form an accurate opinion concerning its quality. In this set of experiments, we show that the intrinsic quality, learned by the proposed model, is correlated with users' perception of the item's true quality, even for items with few ratings.

We focus on items with less than 30 ratings and whose intrinsic quality and average rating for an aspect differ by at least 0.5. Since an item's true quality is unknown, we estimate it by the relative difference in the observed ratings of the same user on a pair of items. This is because if the qualities of two items are similar, a user will rate them similarly. In other words, the difference in the observed ratings by the *same user* on two *similar quality items* should be small.

For each pair of items rated by the same user on the same aspect, let their difference

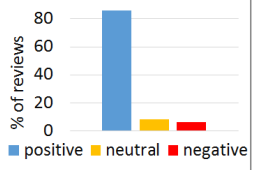
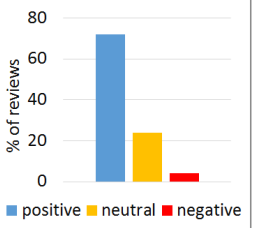
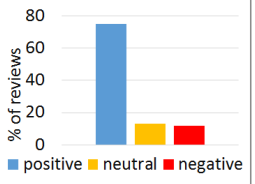
in observed ratings be Δobs, difference between their average ratings be Δavg and difference between the learned intrinsic ratings be Δint. Figure 3.7 shows the correlation between Δobs and Δint, as well as the correlation between Δobs and Δavg aggregated over all aspects. We observe that for both datasets, as Δint increases, Δobs also increases. However, Δavg remains almost constant. This indicates that Δobs is closely correlated with Δint, whereas Δavg appears to be independent of Δobs. This confirms that the learned intrinsic rating is better able to reflect users' perception of the true quality of an item compared to using average ratings of the items.

With the ability of uncovering intrinsic ratings Aspect Bias model can help a user compare true quality of two items without being misguided by their average ratings. We believe this will be immensely beneficial for users as well as service providers to estimate the true quality of an item even before it has seen a lot of ratings.

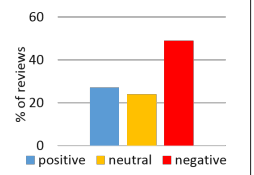
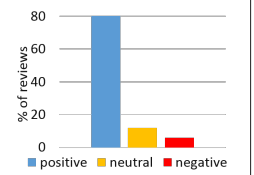
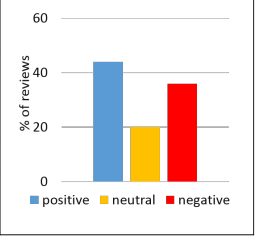
3.3.4 Case Study

Finally, we present the reviews of two randomly sampled users from OpenTable to demonstrate that the aspect bias learned by our model correlates with their review texts (see Figure 3.8). The first user is from group G2 in Figure 3.6 that is particularly critical about *Value*. From the reviews of this user (Figure 3.8a), as well as the reviews of randomly selected users from other groups for the same item, we see that the user from group 2 is indeed critical. We further confirm this observation by manually going through 100 randomly sampled reviews and tabulate the sentiment distribution of each item. We observe that this user is consistently critical even though the majority opinion is positive. The second user belongs to group G6 in Figure 3.6, whose group bias is more or less neutral on the aspect *Ambience*. Figure 3.8b shows that her opinion closely mirrors that of the sentiment distribution of the majority of opinions. She is positive on item 2 and mentions issues related to noise on items 1 and 3, which is reflective of the consensus of opinions of those items.

These two cases further strengthen the fact that the group bias captured by our model is accurate and can help us better interpret a users' rating.

Item	Reviews	Sentiment Distribution of Reviews
Item 1	**... this place is a trap, very expensive for the poor food quality and lack of service...**	
	... very tasty Greek food. Service was very friendly...	
	...prices were pretty reasonable for the portion and quality...	
	...very good lunch spot.. prices were very reasonable....	
Item 2	**...it is a pity-current status deserves a total reconsideration of how to keep the good name it had... waiters are tired, food is far from acceptable...**	
	...very pleasant ambience, excellent food and service.. very friendly home like atmosphere...	
	...while restaurants come and go, with 50 years in business they are definitely doing it right.. never disappoints..	
	...superb menu with generous portions.. highly recommended..	
Item 3	**...It's a shame that such famous owner offers such inferior food. Amounts served are ridiculous, quality barely acceptable, exorbitant price...**	
	...what appeared to be a small portion was exactly well served and balanced	
	The portions were generous, food was tasty and splendidly presented..	
	Exquisite food, expert service and excellent advice on wine..	

(a) Critical user on *Value*

Item	Reviews	Sentiment Distribution of Reviews
Item 1	...my complaints are the wait and the fact that it is very noisy...	
	...the noise level here! impossible to hear your companions. It is tiresome to keep asking people to repeat themselves...	
	... very very loud...	
	... so incredibly noisy. We had to speak so loudly to be heard that our voices were hoarse by the end of the meal ...	
Item 2	..we went for Restaurant week, Waitress was helpful, engaging and hilarious...	
	..wonderful, friendly and efficient service..	
	..delicious restaurant week menu at a reasonable price with great service. We will be back...	
	... great food, with great service...	
Item 3	...still I would not recommend this place because the noise level was so high.. The have NO cushioning of sound!!...	
	... however, the noise level is just not acceptable if you want a conversation that does not require shouting or leaning across table to hear ...	
	... we had a great meal but it was crowded and very loud	
	...it is a little loud partly because the ceilings are so high and because the kitchen area is open	

(b) Neutral user on *Ambience*

Figure 3.8: Reviews of user belonging to "critical" and "neutral" group contrasted with other reviews on the same items from OpenTable dataset

3.4 Summary

We have presented a novel approach to understand users' aspect bias, while capturing aspect dependencies as well as the proper ordinal nature of user responses. Our construction of the stick-breaking likelihood coupled with Pólya-Gamma auxiliary variable augmentation has resulted in an elegant Bayesian inference of the model. Empirical evaluation on two real world datasets demonstrates that through proper statistical modeling of data we are able to capture users' rating behavior and outperform state-of-the-art approaches. Furthermore, our model is effective in user modeling, analyzing users' aspect preferences and provides a better product quality estimation even when the product has received few ratings.

Chapter 4

Improving Usability of Reviews: Finding Supporting Opinions

4.1 Introduction

We now focus on improving the usability of individual opinions expressed in user reviews as general information sources. In the previous chapter (Section 3), we studied the effect of varying aspect preferences of users on their observed ratings. Such aspect biases determine the way a person perceives an item and also influence the sentiments expressed in her reviews. Therefore, we wish to capture the aspect biases of authors in order to interpret their reviews better. In addition to the overall sentiment expressed in reviews, in order to make an informed decision a user often reads through many reviews looking for some specific feedback on the item. For example, if a person needs to book a hotel and plans to do an early check-in and comes across a review that mentions a hassle-free early check-in (as shown in Figure 4.1), it will be helpful to know whether other guests also had similar experiences. If a review complains about bed bugs or noise from construction nearby, then it is important to know if that was an occasional problem based on a single users experience or happens frequently. Opinions expressed in a review could be colored by a person's bias or be a stand-alone or rare experience. One tends to look for a consensus around an opinion to verify if it is fairly common, before trusting a stranger

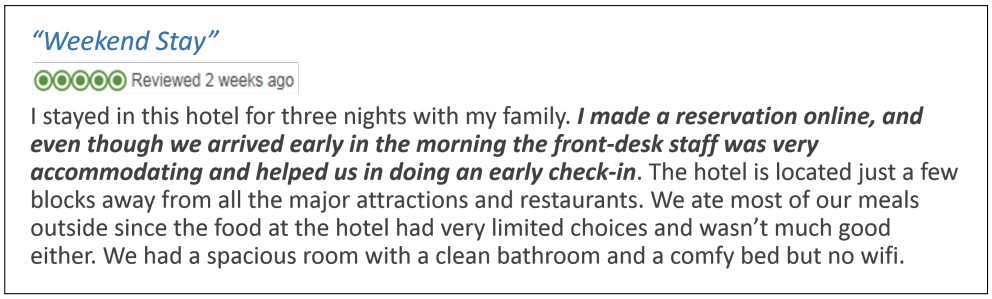


Figure 4.1: A sample hotel review

completely. According to a recent survey published by TripAdvisor [1], 80% of users have to read through at least 6-12 reviews before making a decision and over 64% people would tend to ignore extreme and rare comments in reviews. However, given the large volume of reviews generally present for such items, manually searching for consensus around an opinion can be an extremely time-consuming and daunting task. Current service providers such as TripAdvisor or Hotels.com offer little or no support for finding reviews that express similar opinions.

In this work, we study the problem of finding supporting sentences from reviews that corroborate the opinions expressed in a target review sentence. This can be of great practical importance to the users by enabling them to easily look for appropriate comments on the specific issues they are interested in. This will also be beneficial for service providers to address and improve their service on the issues commonly mentioned by the users. To this end, we propose a framework called SUpporting Reviews Framework (SURF), that first identifies opinions expressed in a review, and then finds similar opinions for it from other reviews.

A review is a collection of sentences where each sentence may have multiple semantic segments, separated by punctuation and/or conjunctions. Each segment expresses an opinion that can be represented as a combination of aspect, topic and sentiment. An aspect refers to the overall theme of a segment, a topic is the specific subject or issue discussed in the segment and the sentiment for each topic can be neutral, positive or negative. Table 4.1 shows the segments and the possible latent aspect, topics and sentiment for a sentence of the hotel review in Figure 4.1. In contrast to our work in previous chapter (Chapter

[1] https://www.tripadvisor.com/TripAdvisorInsights/n2665/5-tips-inspired-our-new-traveler-survey

3), we do not have explicit labels for aspects in the reviews and now wish to find these aspects automatically through our model.

Review Sentence	**Segments**	**Aspect**	**Topic**	**Sentiment**
We had a big room with clean bathroom and a comfy bed, but no wifi	We had a big room with clean bathroom	room	room	positive
			bathroom	positive
	a comfy bed	room	bed	positive
	no wifi	amenities	wifi	negative

Table 4.1: Opinion structure for a review sentence

Given an opinion (in a target segment), we say that a review supports the opinion, if it contains some segment whose aspect, topic and sentiment are similar to those in the target segment. Finding such supporting reviews is a challenge since reviews are typically short unstructured text and discuss a wide range of topics on various aspects with differing sentiments and vocabulary used. Furthermore, sentiments may not be expressed explicitly using common words such as 'good' or 'bad', but can often be subtle and contextual.

Topic modeling approaches (as discussed in Section 2.1) have been widely used to reduce the effect of huge vocabulary by grouping words in topics. However, one fundamental assumption of topic models is the *independence* of topics even in the same document i.e. the topics of all words in the same document are assumed to be independent. This fails to capture the natural coherence present in user generated text such as reviews, which rarely consist of isolated, unrelated sentences, but are composed of collocated, structured and coherent groups of sentences [36]. We observe that an author's train of thoughts when writing a review is often linear, i.e., they will finish discussing one aspect before moving on to the next. In Figure 1, we see that the user first commented on the *Service* aspect (*"front-desk staff was very accommodating"*), then the *Location* aspect, followed by the comment on *Food*, and finally moved on to *Room*. This shows that the aspects discussed in a review are not chosen from a simple *independent mixture*, but rather, words in close proximity tend to discuss the same aspect. Furthermore, within a review the aspects discussed in the current segment will affect the possible aspects for the successive segments.

We explicitly model such review thought patterns by constraining aspect transition between segments. Previously, HTMM [30] and HTSM [98] have modeled topic coher-

ence and semantic shifts by considering topic transition between sentences. However they do not capture the non-repetitive discourse observed in reviews. Another line of work in the literature captures the sequential nature of ideas among segments, especially seen in movies or books using a progressive topical dependency model [20, 21]. However, unlike books, the sequence of topics in reviews is not significant, but, once a topic has been discussed in a review, it is unlikely to be mentioned again in a later segment. From this perspective, our modeling objective is similar to labeled LDA [99], where topic distribution of a document is constrained. However, unlike labeled LDA, the possible aspects of a segment are need to be dynamically constrained depending on aspects discussed in previous segments.

We dynamically constrain aspect transition between segments using a review specific Markov chain. Each segment in a review is assumed to discuss a single aspect. The possible aspects for a segment are limited and made dependent on the aspects already discussed in the previous segments of the review. By tracking aspects of previous segments we are able to ensure constrained aspect sampling for accurate modeling of a review structure. This non-iterative nature of discourse has not been considered by any existing work.

For modeling an opinion properly, capturing the sentiment expressed for an aspect is also of utmost important. The standard topic model LDA [7] assumes words come from only the topic dimension and do not capture sentiment. There has been multiple extensions of LDA to model sentiment. JST [59] introduces a latent sentiment varibale in the model but assumes documents to have only a single sentiment. In ASUM [39] the authors assume that all words in a sentence are associated with the same topic and sentiment, which is often not true in the case of reviews. In our model, we wish to handle the more realistic scenario where sentiments may vary depending on the topics discussed in a review, e.g., an author might like the *location* of a hotel but not the *service* of the staff. Some recent works [44, 39, 73, 124, 118, 119] have developed models to capture both aspect and sentiment. However, they do not consider the preferences of authors, or the inherent quality of the entity for the aspect. In a hotel review, the sentiment expressed for *service* depends on both the service standard of the hotel (evident from the sentiment

distribution of *service* of all reviews for the hotel) and the expectation of the author for *service* (evident from the sentiment distribution of the author on *service* across all hotels). A hotel's location can be "*just a few minutes walk from major attractions*" for someone, but if a user places high importance on the aspect *distance*, the same hotel could be "*too far away*" for her, and her negative sentiment will be reflected in her reviews across hotels. We take this into account by making the sentiment distribution of a review dependent on both the entity and the author.

We propose an Author-aware Aspect Topic Sentiment model (Author-ATS) to capture the diverse opinions expressed in reviews, taking into account user preferences and thought patterns. The model considers a word to be generated from a hierarchy of aspect, topic and sentiment and encodes the coherent structural property of a review by dynamically constraining aspect distributions. We also develop a non-parametric version of Author-ATS based on Dirichlet Process called Author-ATS (DP). In Author-ATS (DP) we do not have to pre-specify the number of topics and the model can figure that out on its own..

We develop a framework called SUpporting Review Framework (SURF) that utilizes the Author-ATS model to compute the similarity of an opinion in a target segment to those in the review corpus, and returns the top-k supporting reviews. Our similarity measure takes into account both the lexical and semantic meaning of a segment in evaluating its support to the opinion in a target segment. Since a target sentence may contain opinions on multiple aspects, SURF will return a diverse set of answers with supporting sentences for each aspect in the target sentence.

Extensive experiments on real world review datasets from TripAdvisor and Yelp show the effectiveness of Author-ATS in modeling opinions compared to existing topic models. Furthermore, SURF outperforms keyword-based approaches and word embedding based similarity measures in finding supporting opinions. To the best of our knowledge, this is the first work to find supporting reviews for an opinion expressed in user generated contents.

Overall, the key contributions of this work are as follows:

- To the best of our knowledge, this is the first work to introduce the problem of finding supporting sentences for a given opinion expressed in user generated contents to facilitate the validation of an opinion through consensus among authors.

- We propose novel parametric and non-parametric versions of Author aware Aspect Topic Sentiment model (Author-ATS)that considers a word to be generated from a hierarchy of aspect, topic and sentiment. The models encode the coherent structural property of a review by dynamically constraining aspect distributions. They also account for the role of both author and entity in a review, by deriving the sentiment from an author-entity specific joint distribution.

- We present a similarity measure that considers both lexical and semantic similarity, using the aspect-topic-sentiment inferred by Author-ATS for ranking opinion sentences according to their support to a given sentence.

- Extensive evaluations on real world review datasets (TripAdvisor and Yelp) show the effectiveness of Author-ATS in modeling opinions compared to existing topic models. Furthermore, for the task of finding supporting opinions SURF outperforms state-of-the-art keyword-based and word embedding approaches.

4.2 Overview of SURF

We start with an overview of our proposed framework for retrieving reviews containing supporting opinions as depicted in Figure 4.2. It consists of two main components. The first component takes as input a set of online reviews and parses the reviews into segments. These segments are then used to train an Author-ATS model. With the trained model, we represent each sentence in the review corpus as a mixture of aspects, topics and sentiments and store them in a database.

The second component takes as input a target sentence in a review and computes the similarity between the target sentence and sentences from other reviews. We define a new similarity measure, Lexical Semantic Similarity (LSS). It considers lexical similarity of

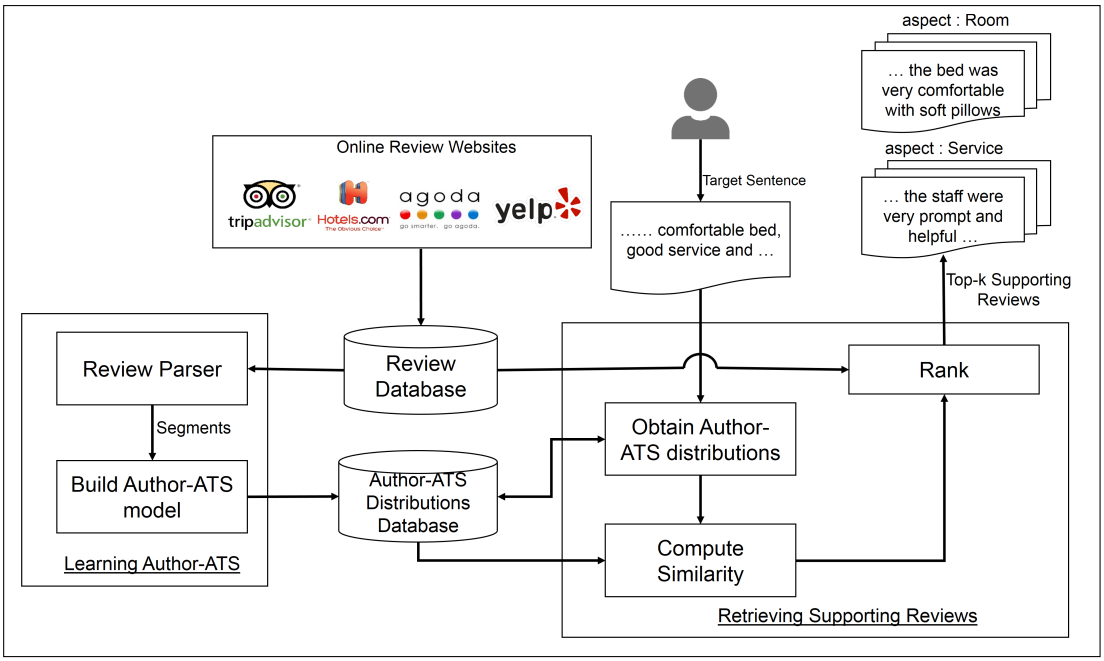


Figure 4.2: Overview of SURF

the two sentences, as well as their semantic similarity computed using the distributions learned by Author-ATS model. For sentences with multiple aspects, we proportion the top k returned supporting reviews according to the importance of each aspect in the target sentence to ensure diversity of the results.

We describe these two components in detail in the following two sections.

4.3 Author-ATS Model

In this section we present the proposed Author-ATS model to learn the aspect, topic and sentiment distributions of a review, taking into account (1) the natural writing style of a person and (2) aspect biases of the author. The standard LDA [7] models a document as a mixture of topics and a topic as a distribution over words. But this fails to represent a more complex structure of opinions, where topics are often implicit and sentiments are subtle. Author-ATS models an opinion as hierarchical dependent mixtures, where words are generated from a three-level hierarchical structure of aspects, topics and sentiments. We assume there are A distinct aspects for a domain, for each aspect there are Z topics and for each aspect-topic pair S possible sentiments. We treat a segment as the basic semantic

unit, discussing a particular aspect. A review sentence has one or more segments, and each segment is a collection of words. Each word is associated with an aspect, topic and sentiment. In other words, a review r is a collection of D_r segments where each segment is a document d, consisting of N_d words.

In the following subsections we describe the assumptions and detailed construction of the proposed model.

4.3.1 Constrained Aspect Generation

We explicitly model the behavior that after an author has finished discussing an aspect and has moved on to the next, he or she is unlikely to return to it again. We assume that each document d discusses a single aspect a_d. The aspect distribution σ_r is drawn from a Dirichlet with parameter α. In order to model the linear writing style of authors, we constrain the possible aspects that can be sampled from σ_r. Whenever an author starts writing a segment, he or she can choose to either (a) talk about an aspect not yet discussed, or (b) continue with the aspect of the previous segment. This is captured by imposing the constraint that the aspect of the j^{th} document is dependent on the aspects of the $(j-1)^{th}, (j-2)^{th}, \cdots, 1^{st}$ documents of the same review.

With this we relax the *independent mixture* assumption of the standard LDA model for aspects and form a review-specific Markov chain (see Figure 4.3). Such a higher order Markov chain would normally incur intractable computational complexity due to the exponential size of transition probability matrix. However, in our case, the transition probability can be determined by overall aspect distribution of the review, σ_r and a list of possible aspects for the segment. Since we assume a non-repetitive nature of discourse, the number of possible aspects for a segment is monotonically decreasing for successive segments. This special property enables us to devise a dynamic programming strategy to solve the problem with linear complexity.

Each document is associated with a binary aspect vector Λ. We restrict the sampled aspect of a document to be drawn from only the aspects that are turned on, in Λ of that document. For a document d, $\Lambda_d =< l_1, \cdots, l_A >$ where each $l_a \in \{0, 1\}$ and A is the

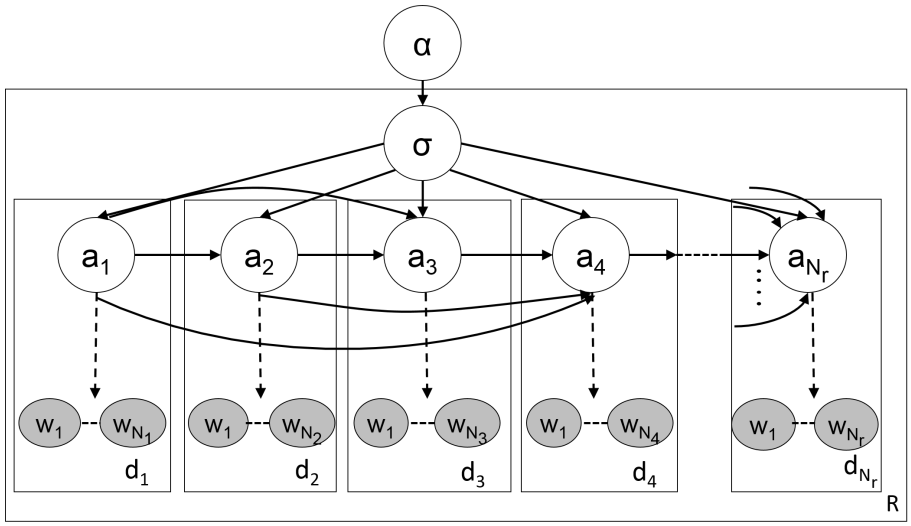


Figure 4.3: Constrained aspect generation in Author-ATS. Aspects in review form a Markov chain.

total number of aspects. Traditionally, for a document d, an aspect a_d is sampled from a multinomial distribution σ_r. Here, we restrict the possible sampled aspects to the list Λ_d. A value of 1 for the entry l_a indicates that the aspect a can be sampled, while 0 indicates that the aspect should not be sampled.

We generate Λ_d by tossing a Bernoulli coin for each aspect a with prior probability Φ_a for value 0. We set Φ_a as the sampling probability for aspects which have been sampled for a previous document. This ensures that an aspect which has been discussed before has lesser probability of coming up again. We set $\Phi_a = 0$ for aspects not sampled in the past, and for the aspect of (immediately) preceding segment. This models aspect coherency in a review document where an author either chooses to discuss a new aspect or continues to talk about the current one.

We define the list of possible aspects for the document d to be $\lambda_d = \{a \mid \Lambda_d[a] = 1\}$. We sample an aspect a_d from σ_r with the constraint that $a_d \in \lambda_d$ i.e. an aspect can be sampled for a document only if it is turned on in the binary aspect vector for the document and thereby exists in the list of possible aspects for the document. Thus, the aspect transition probability among documents becomes dependent on σ_r and the vector λ_d. Unlike regular topic models, Author-ATS is no longer invariant to reshuffling of words and is able to model linear aspect coherency in a review.

4.3.2 Author-Entity dependent Sentiment Distribution

We account for the dual role of entity and author in a review, by observing that the sentiments expressed are influenced by both the quality of the entity being reviewed and the preferences or biases of the author. We use two Dirichlet distributions to derive sentiment, namely, entity-dependent distribution (ξ) and author-dependent distribution (χ). For each aspect-topic combination, ξ is drawn from a Dirichlet distribution with prior γ^1 and χ is drawn from a Dirichlet distribution with prior γ^0.

Since online reviews describe experiences of people, some words tend to appear frequently regardless of the aspect being talked about (e.g.: 'hotel','trip' or 'mobile', 'phone' for hotel and mobile reviews respectively). We call them *domain stopwords* as they are not specific to any aspect. It is not possible to collect these words with off-the-shelf stopword dictionary since they are domain dependent. We use a binary switching variable y_i to determine the type for the i^{th} word. If $y_i = 0$, then the word is aspect neutral (domain stopword); and if $y_i = 1$, it is aspect dependent.

The generative process of the model is as follows:

- Draw a multinomial word distribution ϕ_0 for domain stopwords and ϕ_1 for each aspect, topic and sentiment words from Dir (ω).
- For each author u, draw a multinomial sentiment mixture χ for each aspect and topic from Dir(γ^0)
- For each entity e, draw a multinomial sentiment mixture ξ for each aspect and topic from Dir (γ^1)
- For each review r:
    1. Draw multinomial aspect mixture σ from Dir(α)
    2. For each document $d \in r$:
        (a) Draw Λ_d from Bernoulli (Φ)
        (b) Draw a type mixture ψ from Beta (δ_0, δ_1)

(c) Sample an aspect a_d from σ s.t. $a_d \in \lambda_d$

(d) For sampled aspect a_d, draw a topic mixture θ from Dir (β)

(e) For each word position i where $0 \leq i \leq N_d$

i. Sample a type y_i from ψ

ii. Sample a topic z_i from θ

iii. Sample a sentiment s_i from χ and ξ

iv. Sample a word w_i from $\begin{cases} \phi_0 & \text{if } y_i = 0, \\ \phi_1 & \text{if } y_i = 1 \end{cases}$

Note that for the first document of a review, we set λ_0 to the set of all possible aspects, such that there is no constraint when sampling for the first segment of a review. Figure 4.4 shows the plate notation for Author-ATS model.

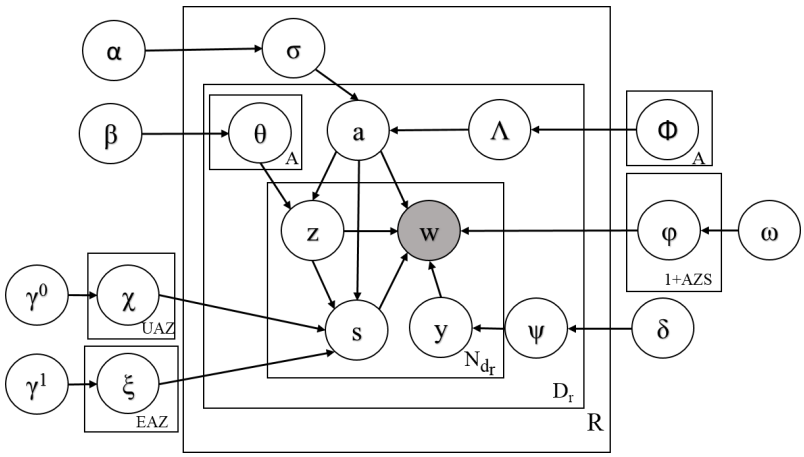


Figure 4.4: Graphical representation of Author-ATS

4.3.3 Bayesian Inference

The exact inference for the posterior distribution is intractable. We employ collapsed Gibbs sampling for inference. Markov chain introduced for aspect coherency makes the aspects non-exchangeable, hence sampling an aspect for a segment will also affect all subsequent segments. Since the exact sampling for this would be computationally expensive, we propose the following approximate posterior considering only the previous segments, which has been shown to work well in similar cases previously [71].

We sample an aspect (a_d) for each document based on the posterior probability of the type, topic and sentiment assignment of each word in the document and the aspects sampled for preceding documents in the review.

$$P(a_d|\vec{a_{-d}}, \vec{y}_{-d}, \vec{z}_{-d}, \vec{s}_{-d}, \vec{w}) \propto P(a_d|\vec{a_{1:d-1}}) \prod_{z=1}^{Z} \prod_{s=1}^{S} \frac{\sum_{w=1}^{W} B(n_w^{a_d,z,s} + \omega)}{\sum_{w=1}^{W} B(n_w^{a_d,z,s,-d} + \omega)} \tag{4.1}$$

$$P(a_d|\vec{a_{1:d-1}}) \propto \begin{cases} \frac{n_{a_d}^{r,-d} + \alpha}{\sum_{a \in \lambda_d} n_a^{r,-d} + |\lambda_d| * \alpha} & \text{if } a_d \in \lambda_d \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

where $B(\vec{x})$ is the multidimensional extension of the Beta function. The notation $n_a^{b,-c}$ refers to the number of times a has been assigned to b excluding current occurrence c, e.g. $n_{a_d}^{r,-d}$ denotes the number of documents in review r that has been assigned aspect a_d excluding current document d.

The target aspect a_d is dependent on the aspects sampled for the 1^{st} to $(d-1)^{th}$ documents of the review, denoted by $\vec{a_{1:d-1}}$. We restrict the target aspect a_d to belong to the set defined by Λ_d of the document d to achieve coherence among aspects respecting the nature of discourse observed in review writing styles. This constrained aspect sampling differentiates Author-ATS from existing topic modeling works on review text by explicitly modeling the topic coherence of opinionated text.

After sampling the aspect for the document, we jointly sample the latent type, topic and sentiment for each word within the document. The posterior for the i^{th} word of document d (written by author u for entity e) is given as:

$$P(y_i, z_i, s_i|a_d, \vec{w}, \vec{y}_{-i}, \vec{z}_{-i}, \vec{s}_{-i}) \propto P(y_i|d) * P(z_i|a_d, d) * P(s_i|a_d, z_i, u, e, d) * P(w_i|y_i, a_d, z_i, s_i,)$$
$$\propto \frac{n_{y_i}^{d,-i} + \delta_{y_i}}{\sum_{y=0}^{1}(n_y^{d,-i} + \delta_y)} * \frac{n_{z_i}^{d,a_d,-i} + \beta}{\sum_{z=1}^{Z} n_z^{d,a_d,-i} + Z\beta} * \Big(q_1 \frac{n_{s_i}^{u,a_d,z_i,-i} + \gamma^0}{\sum_{s=1}^{S} n_s^{u,a_d,z_i,-i} + S\gamma^0}$$
$$+ q_2 \frac{n_{s_i}^{e,a_d,z_i,-i} + \gamma^1}{\sum_{s=1}^{S} n_s^{e,a_d,z_i,-i} + S\gamma^1}\Big) * \frac{n_{w_i}^{\zeta,-i} + \omega}{\sum_{w=1}^{W} n_w^{\zeta,-i} + W\omega}$$

$$\begin{aligned} y_i = 0 &\Rightarrow \zeta = y_i \\ y_i = 1 &\Rightarrow \zeta = a_d, z_i, s_i \end{aligned} \tag{4.3}$$

For sampling sentiment, instead of using a single Dirichlet density we use a Dirichlet mixture as the prior [110, 111]. It is a weighted combination of two individual Dirichlet densities χ and ξ. Mixture coefficients q_1, q_2 are set to 0.5, giving equal weights to both author and entity. The probability for choosing a sentiment for a word depends on how many times the sentiment was chosen by the author of the document for that aspect-topic combination and how many times it was chosen for that particular entity. This ensures that the chosen sentiment reflects both the entity's quality for that topic as well as the author's preferences.

4.3.4 Non-parametric Author-ATS (DP) Model

While the number of aspects for a domain are limited, the number of topics for each aspect may vary significantly and can be difficult to estimate. For restaurants, the topics for *ambiance* are fewer (e.g. music, crowd etc.) compared to *food*. This motivates us to propose a non-parametric version of the Author-ATS model where the number of topics can be automatically discovered.

In this non-parametric version, topic inference is done through Chinese Restaurant Process (CRP), a popular variant of Dirichlet Process (DP). In a Chinese restaurant with infinite number of tables, each with infinite capacity, CRP determines if a customer

chooses to sit at an occupied table (with a probability proportional to the number of customers already sitting at the table), or an unoccupied one.

Following the idea of CRP, each observed aspect dependent word can either be assigned to an existing topic or to a new topic. The conditional distributions for the Gibbs sampler are given by:

$$P(y_i, z_i, s_i | a_d, \vec{w}, \vec{y}_{-i}, \vec{z}_{-i}, \vec{s}_{-i}, \beta, \gamma^0, \gamma^1, \delta_0, \delta_1, \omega) \propto$$

$$\begin{cases} \frac{n_{y_i}^{d,-i} + \delta_{y_i}}{\sum_{y=0}^{1}(n_y^{d,-i} + \delta_y)} * \frac{n_{z_i}^{d,a_d,-i}}{\sum_{z=1}^{Z} n_z^{d,a_d,-i} + \beta} * \Big(q_1 \frac{n_{s_i}^{u,a_d,z_i,-i} + \gamma^0}{\sum_{s=1}^{S} n_s^{u,a_d,z_i,-i} + S\gamma^0} + \\ q_2 \frac{n_{s_i}^{e,a_d,z_i,-i} + \gamma^1}{\sum_{s=1}^{S} n_s^{e,a_d,z_i,-i} + S\gamma^1}\Big) * \frac{n_{w_i}^{\zeta,-i} + \omega}{\sum_{w=1}^{W} n_w^{\zeta,-i} + W\omega}; \text{ for an existing topic} \\ \\ \frac{n_{y_i}^{d,-i} + \delta_{y_i}}{\sum_{y=0}^{1}(n_y^{d,-i} + \delta_y)} * \frac{\beta}{\sum_{z=1}^{Z} n_z^{d,a_d,-i} + \beta} * \Big(q_1 \frac{\gamma^0}{S\gamma^0} + q_2 \frac{\gamma^1}{S\gamma^1}\Big) * \frac{\omega}{W\omega}; \text{ for a new topic} \end{cases}$$

$$\begin{aligned} y_i = 0 &\Rightarrow \zeta = y_i \\ y_i = 1 &\Rightarrow \zeta = a_d, z_i, s_i \end{aligned} \tag{4.4}$$

4.4 Retrieving Supporting Reviews

Given a target sentence in a review SURF computes its similarity with other review sentences using the distributions learned by Author-ATS and returns a list of supporting reviews. A sentence supports another sentence if they are either **lexically similar** (have similar words) or **semantically similar** (implicitly expressing a similar viewpoint).

4.4.1 Lexical Similarity

Two sentences are lexically similar if they share keywords that are important for an aspect. While describing *Service* of a hotel, if two review sentences both use the same word like 'helpful', they should have high lexical similarity. A popular method for computing lexical similarity is the vector-space model. If we treat each review sentence as a vector then lexical similarity between them ($lexical\_sim$) is computed as the cosine-

similarity between these two vectors. The i^{th} entry of a vector signifies the importance of the corresponding word to its assigned aspect computed using the standard *tf-idf* weighting scheme. In tf-idf, *tf* stands for term frequency and *idf* for inverse document frequency. For our purpose, we define tf-idf of a word(w) with respect to an aspect(a) as:

$$tf(w,a) = \sum_{d=1}^{D} P(w|d,a) \tag{4.5}$$

$$P(w|d,a) = \begin{cases} P(w) & \text{if } w \text{ assigned to } a \text{ in } d \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

$$idf(w,\mathbf{A}) = log\frac{A}{1 + |a \in \mathbf{A} : \exists d \in \mathbf{D}, P(w|d,a) > 0|} \tag{4.7}$$

$P(w)$ is the generation probability obtained from Author-ATS model. As shown in equation 4.6, if a word w in document d is assigned to aspect a by the Author-ATS model, then $P(w|d,a)$ is its generation probability, or considered as 0 if it is assigned to some other aspect. For *tf* we consider the summation of generation probability of a word from the given aspect across all its occurrences and for *idf* we consider the number of different aspects (in aspect set $\mathbf{A}$) the word has been generated from.

Since words are important with respect to an aspect, unlike traditional tf-idf, these values are computed across reviews on the whole corpus. Words frequently used for describing an aspect often tend to converge across reviews, even though written by different users.

4.4.2 Semantic Similarity

Two sentences can be semantically similar if they share the same sentiment for an aspect and topic even though they use different words. For example consider the two sentences, *"The hotel was quite close to space needle"* and *"Major attractions are just walking distance from the hotel"*. Even though they are lexically dissimilar, they have high semantic similarity as they both talk about the same aspect *Location* on the topic 'attractions' with

a positive sentiment.

Let C be the set of words in a sentence. The sentiment for an aspect in a sentence is taken to be the sentiment, majority of the corresponding aspect's words in the sentence belong to. Two sentences are considered to be similar under an aspect only if they share the same sentiment. Aspect-topic probability of a sentence is defined as the ratio of generation probability of words generated from the aspect-topic pair a, z to the summation of generation probabilities of all the words in the sentence.

$$P(C|a,z) = \frac{\sum_{w \in C} P(w|w \text{ has aspect } a \text{ and topic } z)}{\sum_{w \in C} P(w)} \tag{4.8}$$

We define sim_0 to measure the similarities between two sentences (C_1 and C_2) having the same aspect, topic and sentiment, and sim_1 to measure the similarities of two sentences with the same aspect and sentiment but discussing different topics.

$$sim_0(C_1, C_2, a) = \sum_{z=1}^{Z} P(C_1|a,z)P(C_2|a,z) \tag{4.9}$$

$$sim_1(C_1, C_2, a) = \sum_{z_1, z_2 \in [1 \cdots Z] z_1 \neq z_2} P(C_1|a,z_1)P(C_2|a,z_2) \tag{4.10}$$

Intuitively, two sentences about the same aspect on the same topic should have high semantic similarity; whereas two sentences that talk about two different topics under the same aspect should have a relatively lower similarity. The semantic similarity between two sentences is:

$$semantic\_sim(C_1, C_2, a) = sim_0(C_1, C_2, a) + \delta sim_1(C_1, C_2, a) \tag{4.11}$$

where δ is a damping factor with value less than 1.

Lexical-semantic similarity (LSS) of two sentences with same sentiment for an aspect is measured as a weighted combination of their $lexical\_sim$ and $semantic\_sim$ as,

$$LSS(C_1, C_2, a) = \lambda lexical\_sim(C_1, C_2, a) + (1 - \lambda) semantic\_sim(C_1, C_2, a) \quad (4.12)$$

where λ is an empirically chosen parameter to trade-off between the weights for semantic and lexical similarity.

4.4.3 Ranking of Reviews

Given a review sentence, we employ kNN search to find the k most similar sentences for each of its aspects according to LSS measure. Since a target sentence C may contain multiple aspects, we determine the importance of an aspect a to C as follows:

$$Imp(C, a) = \frac{\sum_{w \in C} P(w|w \text{ has aspect } a)}{\sum_{w \in C} P(w)} \quad (4.13)$$

For each aspect a with $Imp(C, a) > 0$, we return the top $k * Imp(C, a)$ sentences from the review corpus. For example, consider the following sentence

- *very friendly staff, free wifi and a wondeful choice of free breakfast*

It is found to contain aspects 'Service', 'Amenities', 'Food' with importances 0.25, 0.15 and 0.6 respectively. In Top 5 supporting sentences, we present a mix of supporting sentences for each aspect in proportion to their importance in the target sentence (i.e. 1, 1, 3 respectively for our example). The Top 5 supporting sentences retrieved for the above example are:

1. *hotel is very nice and staff is friendly*
2. *free parking and wifi*
3. *free breakfast buffet was plentiful*
4. *rooms were clean, breakfast was very good*

5. *rooms are fresh, staff is friendly and most importantly for me the breakfast was amazing*

Proportionately allocating supporting sentences from each aspect in the top-k results diversifies the result set and ensures that a user is able to find information about whichever aspect of the target sentence she wished to verify.

4.5 Experiments

We perform two sets of experiments to evaluate our proposed framework. We first compare Author-ATS with state-of-the-art topic models using perplexity on test data and also show a qualitative analysis. Then we evaluate the performance of SURF, for the task of retrieving supporting opinions using human annotation, against keyword based search engine Lucene and a competent word embedding model Word2Vec. We use two large real world datasets: (a) hotel reviews from TripAdvisor [124], and (b) restaurant reviews from Yelp.com. Table 4.2 shows the statistics of the two datasets.

Dataset	# entity	# author	# review	# sentence	# vocab
TripAdvisor	12,773	781,403	1,621,956	20,244,293	980,323
Yelp	578	16,981	25,459	232,107	56,200

Table 4.2: Statistics of datasets used

4.5.1 Preprocessing

We pre-process both datasets by first converting all words to lower-case forms and removing domain independent stopwords[2]. We retain some negation stopwords (e.g.: *not, can't, didn't*) and join them with the next word (so that 'not good' is treated as a single unit) to help discover sentiment properly. We use common punctuations used for marking end of sentences like '.', '?', '!' to split a review into sentences. To further split a sentence into segments we use punctuations used to separate clauses like ',', ';' and conjunctions like 'and', 'however', 'but' as separators. Apart from the punctuations used for splitting

[2]http://www.ranks.nl/stopwords

sentences we use punctuations used to separate clauses within a sentence like ‘,’, ‘;’ as well as conjunctions like ‘and’, ‘however’, ‘but’ as separators.

Aspect words need not necessarily be single words but may consist of highly co-occurring words (e.g. ‘front-desk’, ‘walking distance’). In computational linguistics Pointwise Mutual Information (PMI) [67] has been widely used for studying such associations between words and finding collocations. We compute PMI score for each bigram in the corpus and if the score is found to be considerably high (0.05 in our experiments) they are treated as a single word.

To make the discovered aspects understandable and intuitive to humans, we provide a few domain dependent seed words to the models. The seeds are only used during initialization and subsequent iterations of Gibbs sampling are not dependent on them. Table 4.3 lists the aspect seed words used in our experiments for both domains. We also use a domain independent subjectivity lexicon[3] to initialize sentiment distributions.

Aspects	Seed Words
Value for Money	value, rate, price
Room	room, bed, bathroom, clean
Location	location, walk, minute
Service	staff, reservation, front-desk
Food	restaurant, breakfast, buffet
Amenities	pool, parking, internet, wifi

(a) TripAdvisor Dataset

Aspects	Seed Words
Value for Money	value, rate, portions, price
Service	ambience, wifi, music, service
Food	steak, rice, burger, cocktail

(b) Yelp Dataset

Table 4.3: Sample Aspect Seed Words

4.5.2 Parameter Settings

We empirically set the parameter values for ATS based models as: $\delta_0 = 3.0, \delta_1 = 2.0, \alpha = 0.1, \beta = 0.1, \gamma_0 = 1.0, \gamma_1 = 1.0, \gamma_2 = 3.0$. These parameter values can be interpreted

[3]http://mpqa.cs.pitt.edu/lexicons/subj_lexicon

as our prior beliefs for the variable counts. For the non-parametric model we set the concentration parameter β to be 10^{-8}.

We use symmetric priors for all parameters except for δ and γ. The verbosity of reviews is captured by a slightly higher prior frequency for *domain stopwords* (δ_0) than *aspect* words (δ_1). Similarly, for sentiment priors, often the probability of neutral words is more compared to sentiment words (e.g. *people at the reception desk were very friendly*). Thus, it is more natural to choose an asymmetrical prior for sentiments for both entity specific as well as author specific distributions. The iteration number for Gibbs sampler was set to 1000. The value for damping factor δ is set to 0.4 and the value for the trade-off parameter λ in equation 4.12 is set to 0.6 empirically.

4.5.3 Evaluation of Author-ATS Model

In this set of experiments, we examine the ability of Author-ATS to capture the opinions in reviews.

We use perplexity as a measure of convergence of topics to indicate the generative power of the models. Perplexity is derived from the likelihood of unseen test data and is a standard measure for evaluating topic models.

$$Perplexity(D_{test}) = exp\left(-\frac{\sum_{d \in D_{test}} logP(d|D_{train})}{\sum_{d \in D_{test}} N_d}\right) \tag{4.14}$$

The lower the perplexity, the less confused the model is on seeing new data, implying a better generalization power.

We compare with the following state-of-the-art opinion models:

- **LDA [7]** : The basic topic model where words are generated from latent topic dimensions. This does not consider the sentiment of words.

- **TAM [83]**: A topic model for opinion mining where words are generated from a two-level hierarchy of aspect and topic. The aspect and topic are independent and each aspect affects all topics in similar manner.

- **JTV [120]**: A topic model especially for contentious documents where each word has a topic and a viewpoint (sentiment).

We also implement a baseline model **ATS** based on three-level Aspect-Topic-Sentiment hierarchy. We use this model to show the performance gain by just considering a hierarchical dependency between these dimensions while capturing an opinion.

For fair comparison, we try to keep the total number of dimensions as close as possible across models. In TripAdvisor dataset, we set the number of topics in the LDA model to 100. For AuthorATS and ATS, we use 6 aspects, 5 topics for each aspect and 3 sentiments (positive, negative and neutral). The number of aspects and sentiments are the same for Author-ATS (DP) but the number of topics was discovered by Dirichlet Process automatically. TAM does not model sentiment, hence we use 6 aspects and 16 topics. For JTV, we use 33 topics and 3 sentiments. In Yelp dataset, we use 3 aspects, but use asymmetric number of topics for each of them. We observe that in restaurant reviews people talk mostly about the food or drinks and discuss very few topics under price or service. We use 1, 1, 100 as the number of topics for aspects *Value for Money*, *Service* and *Food* respectively in our parametric models. Hence we use 103 topics for LDA, 3 aspects-33 topics for TAM (it does not allow asymmetric number of topics for different aspects), 33 topics and 3 viewpoints for JTV. We partition our dataset into train (80%) and test (20%) sets and report five fold cross validation results.

Table 4.4 shows the results. We first note that all hierarchical models outperform the basic topic model LDA. JTV outperforms the simpler LDA model by assuming that each word in a review is chosen not only for its topic but also for the sentiment it conveys. However, JTV assumes only a single aspect in the document, which does not hold true for reviews, as they discuss multiple aspects of an item. TAM assumes a two level hierarchy of aspect and topic, but in their modeling aspect and topics are independent. However, in reviews, the topics discussed are often closely related to an aspect. Our ATS model considers the appropriate relation between aspect-topic-sentiment in modeling of words and can outperform other models. Author-ATS further improves the performance by considering author and entity characteristics as well as the thought patterns of the authors.

We note that the performance of the non-parametric model is comparable with Author-ATS, making it easier to use the model for any new domain without having much prior knowledge.

Model	**TripAdvisor**	**Yelp**
LDA	5070	5737
TAM	2980	3468
JTV	3430	4370
ATS	2385	3337
Author-ATS	**2212**	**2784**
Author-ATS(DP)	2300	2829

Table 4.4: Perplexity values for different models.

Table 4.5 shows the top words extracted by Author-ATS as domain stopwords. Words like 'hotel', 'stay', 'trip' etc. are extracted as stopwords for hotel domain since they occur very frequently irrespective of the aspect being discussed. Although these words do not convey any aspect information, they are domain dependent and are not found in a general stopword dictionary.

Dataset	Domain Stopwords
TripAdvisor	hotel, nice, stay, trip, times, day, place, back
Yelp	good, place, food, time, order, bit, make

Table 4.5: Domain stopwords from Author-ATS.

Table 4.6 shows top words extracted for a few aspects, categorized into topic-sentiment groups. As we can observe that the majority of the words are correctly clustered in aspects, and further into specific topics. For example, the first topic for aspect *Room* is about in-room experience ('bed','king-size','view'), whereas the second topic seems to be about bathroom ('shower', 'towels', 'tub'). We also observe that the model is able to obtain contextual sentiment terms which are aspect-topic coherent. For example, words such as 'noise', 'night', 'hear' could be assigned negative sentiment labels for topic 0 of *Room* due to the context in which they are used, e.g., when describing a room, these words probably indicate a noisy room bothering their sleep at night.

Impact of Seed Words We vary the number of seed words for an aspect and examine its

Aspect: **Room**		
Topic 0		
Positive	Negative	Neutral
bed	noise	room
comfortable	night	floor
spacious	sleep	view
king-size	window	size
clean	hear	modern
Topic 1		
Positive	Negative	Neutral
bathroom	small	room
large	door	bathroom
tub	barely	shower
shower	tiny	water
shampoo	kitchen	towels

Aspect: **Service**		
Topic 0		
Positive	Negative	Neutral
staff	night	staff
extremely	greet	call
welcoming	problem	front-desk
care	asked	service
friendly	manager	shuttle
Topic 1		
Positive	Negative	Neutral
card	called	check-in
reservation	upgrade	day
airport	manager	arrived
polite	rude	directions
excellent	questions	time

Table 4.6: Top words for aspect-topic-sentiments found by Author-ATS for TripAdvisor dataset.

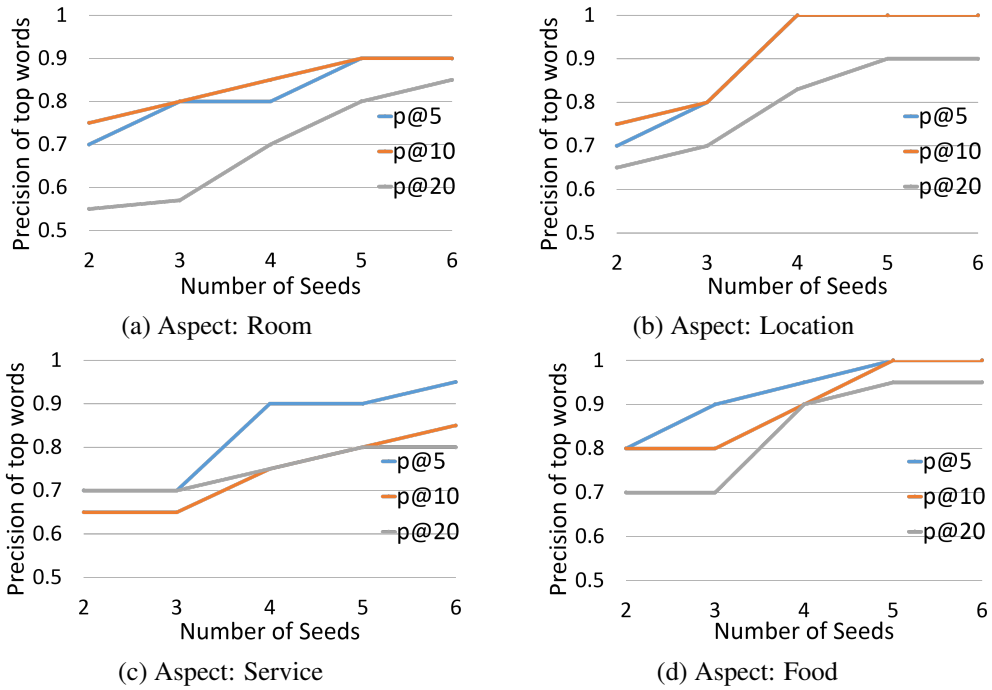
effect on the aspect discovery. We use $p@n$, the fraction of correctly discovered aspect words among the top n words, to evaluate the quality of the results.

The average precision of top-n words for different aspects is obtained by taking the average over all combinations $\binom{6}{m}$ of seed words where m is the number of selected seed words, $2 \leq m \leq 6$. Figure 4.5 shows the results. We observe that the average precision increases with the number of seeds, and stabilizes when $m \geq 4$. This demonstrates that providing a handful of seed words can go a long way for discovering intended, explainable domain specific aspects.

4.5.4 Evaluation of SURF

We now evaluate Author-ATS model and LSS measure on retrieving sentences that are *relevant* to a target sentence. A sentence is considered relevant if it expresses similar opinions as the target sentence. A sentence with multiple aspects is relevant if it expresses at least one of the opinions in the target sentence. Precision of the top-k answers are manually determined by three annotators and conflicts are resolved by majority voting.

Recall that LSS considers both lexical and semantic similarity. The computation of semantic similarity requires the aspect-topic-sentiment distribution which is only available



(a) Aspect: Room

(b) Aspect: Location

(c) Aspect: Service

(d) Aspect: Food

Figure 4.5: Impact of varying number of seeds.

in the baseline ATS and Author-ATS models. We define a similarity measure called CJSD that can be used by the various topic models to facilitate comparison. CJSD measures the lexical similarity of two sentences as the cosine similarity of their tf-idf vectors, while the semantic similarity is measured by the similarity of their topic distributions using Jensen-Shannon Divergence(JSD) as follows:

$$CJSD(s_1, s_2) = \lambda \, cosine\_sim(s_1, s_2) + (1 - \lambda) \, JSD(s_1, s_2) \tag{4.15}$$

We randomly select 5 hotels from TripAdvisor and 5 restaurants from Yelp datasets. For each hotel/restaurant, we randomly pick 10 target sentences and retrieve their supporting sentences. The topic distributions of these sentences are obtained using LDA, TAM, JTV, and the proposed models ATS and Author-ATS.

Table 4.7 shows the average precision for top 5, 10 and 20 results retrieved using various topic models with similarity measure CJSD. We see that Author-ATS model always outperforms other topic models for the task of retrieving supporting sentences. This is consistent with the perplexity results of the models obtained previously.

Table 4.8 shows the average precision using variants of the proposed model with LSS.

	TripAdvisor			Yelp		
	p@5	p@10	p@20	p@5	p@10	p@20
LDA	0.56	0.48	0.45	0.43	0.42	0.42
TAM	0.58	0.53	0.52	0.49	0.47	0.47
JTV	0.51	0.47	0.53	0.41	0.41	0.43
ATS	0.62	0.60	0.55	0.60	0.57	0.44
Author-ATS	**0.68**	**0.62**	**0.61**	**0.60**	**0.58**	**0.56**

Table 4.7: Average precision using CJSD

	TripAdvisor			Yelp		
	p@5	p@10	p@20	p@5	p@10	p@20
ATS	0.69	0.62	0.58	0.62	0.59	0.58
Author-ATS	**0.74**	**0.66**	**0.60**	**0.68**	**0.64**	**0.62**
Author-ATS (DP)	0.64	0.63	0.57	0.62	0.56	0.54

Table 4.8: Average precision using LSS

Clearly, using LSS always yields a better precision compared to using CJSD, with the best performer being the Author-ATS with LSS combination. SURF framework utilizes this combination for retrieving top-k supporting reviews.

Next, we compare SURF with the following methods:

- **Lucene:** A popular keyword based ranking method. It is implemented using Apache Lucene[4]. We used its default combination of vector space model and boolean model for retrieval.

- **Word2Vec:** [70] A state-of-the-art algorithm for word embeddings using neural network. It uses a shallow, two layer neural network to map words to a vector space. Supporting sentences are ranked with Word Mover's distance using the word embeddings. We use the Word2Vec implementation of gensim[5] and train on TripAdvisor dataset using CBOW algorithm with context window set to 5 as recommended by the authors. We do not train Word2Vec on the Yelp dataset as it is too small. We set the vector dimension to 500 based on grid search. We also compare with Word2Vec model pre-trained on the large GoogleNews dataset[6].

Table 4.9 shows the average precision for the top 5, 10 and 20 results retrieved using

[4]https://lucene.apache.org/core/
[5]https://pypi.python.org/pypi/gensim
[6]https://code.google.com/archive/p/word2vec/

Lucene, Word2Vec and SURF. We observe that Word2Vec performs better when trained on review data, compared to the model trained on general news data. This confirms that domain knowledge is important. Word2Vec trained on domain data can outperform the keyword based similarity measure employed by Lucene due to considering the semantic similarity of words. It is evident from the results that SURF significantly outperforms existing approaches for opinion search by considering both the lexical and semantic similarities.

	p@5	p@10	p@20
Lucene	0.67	0.58	0.52
Word2Vec (GoggleNews)	0.62	0.48	0.39
Word2Vec (TripAdvisor)	0.70	0.61	0.51
SURF	**0.74**	**0.66**	**0.60**

(a) TripAdvisor

	p@5	p@10	p@20
Lucene	0.61	0.54	0.49
Word2Vec (GoogleNews)	0.52	0.47	0.37
SURF	**0.68**	**0.64**	**0.62**

(b) Yelp

Table 4.9: Comparison with Lucene and Word2Vec

For evaluating the coherence of retrieved set of supporting reviews for an aspect, we look at their corresponding user given aspect ratings. For each aspect of each review sentence, we retrieve its top-k supporting sentences. Then we compute the standard deviation of the ratings for that aspect in the retrieved supporting reviews. We aggregate the standard deviation values for each aspect over all the reviews and look at the average value. Figure 4.6 shows results for two aspects from the TripAdvisor dataset. Other aspects also had similar trends.

We rank the retrieved results based on their similarity to the target sentence. We observe that as expected, the average standard deviation increases as we retrieve more results, implying similarity among the opinions reduces if we keep increasing the size of the retrieved set. We observe that SURF has a smaller average standard deviation compared to Word2Vec and Lucene. The gap between the performance of SURF and the other methods also widens as the size of the retrieved results increases. This demonstrates SURF's

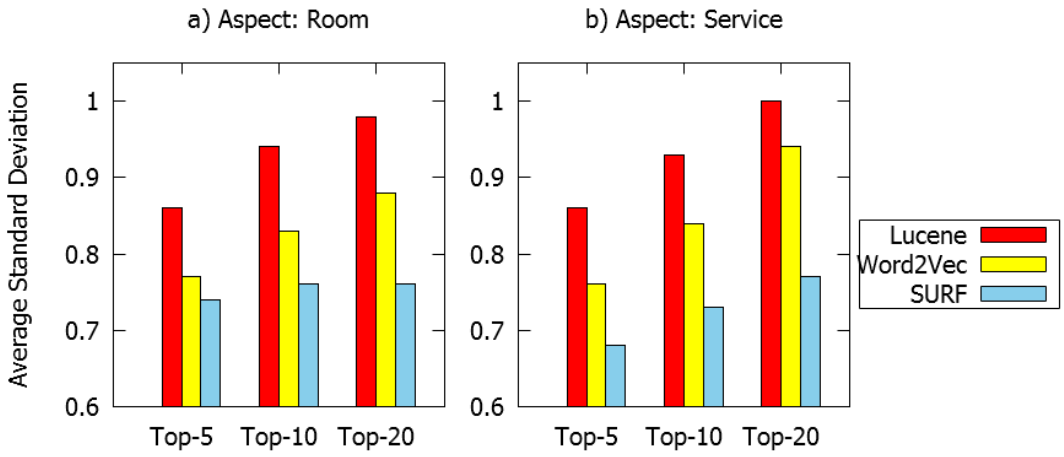


Figure 4.6: Average standard deviations of aspect ratings for supporting reviews. Smaller deviation implies greater coherence.

superiority in retrieving reviews with similar opinions. While Word2Vec can capture semantic similarity and sentiments, the results retrieved are rather noisy, especially as k increases. On the other hand, Lucene relies solely on word similarity ignoring the context or underlying semantics, resulting in a large standard deviation.

4.5.5 Case Study

Table 4.10 shows samples of supporting sentences extracted by the different methods. We observe that the sentences retrieved by SURF are semantically similar although the words may be quite different from the target sentence. In contrast, Lucene may retrieve irrelevant sentences matching a keyword used in a totally different context. Word2Vec considers words used in proximity of one another (e.g. *bed, pillow* with *microwave, coffemaker* etc.) to be similar which clearly does not always imply conformity of opinions.

Furthermore, the retrieved results of SURF are categorized according to their aspects. This makes it easy for users to interpret the results. Particularly if a target sentence has multiple aspects, then SURF will retrieve results for each aspect. For example, for the second target sentence shown in Table 4.10, the results contain supporting statements for both *room* and *service*. If a user then wishes to view more results for one of those aspects it will be possible for SURF to fetch more results only for that aspect.

Target Sentence: bedroom had the most comfortable mattress, feather soft pillows as well as firmer ones, they thought of keeping every guest comfortable		
Supporting Sentences by **SURF**	Supporting Sentences by **Lucene**	Supporting Sentences by **Word2Vec**
Aspect : **Room** **Statement**: bill clinton suite was huge with two baths, a wonderful jacuzzi and a comfortable bed	bed was very comfortable, as were the large pillows	The room had a microwave, coffemaker, hairdryer, bottled water replenished each day **(x)**
Aspect : **Room** **Statement**: the beds are the most comfortable of any hotel I have stayed in	we were recommending it for our out of town wedding guests, and wanted to make sure they were comfortable **(x)**	It really is a shame because the bed and pillows were super comfortable and we could have had a great night sleep on both nights
Aspect : **Room** **Statement**: the beds were comfortable and they had a good selection of towels	who would have imagined that somebody actually thought about where a guest would watch tv **(x)**	They took regular sized hotel rooms and divided them into a sitting room with a bedroom with a door, keeping the bathroom to divide the two areas **(x)**

(a) Target Sentence with Single Aspect

Target Sentence: the check in was quick, with friendly polite service, and the room was very big with a very comfortable king size bed		
Supporting Sentences by **SURF**	Supporting Sentences by **Lucene**	Supporting Sentences by **Word2Vec**
Aspect : **Room** **Statement**: bed was extremely comfortable, I'm hard to please in the department because I sleep on a sleep number bed at home	the room was a great size; bed was very comfortable	The first room assigned was very small and dingy with one king sized bed that just fit **(x)**
Aspect : **Room** **Statement**: room size was large and bed was comfortable	king size bed was comfy	bathroom was well furnished with soap, shampoo/ conditioner, very large, soft towels - perfect **(x)**
Aspect : **Service** **Statement**: service is very friendly	our room faced denny park **(x)**	the room was large and the bed very comfortable and our room faced the street and it was very quiet

(b) Target Sentence with Multiple Aspects

Table 4.10: Sample Supporting Sentences Retrieved by SURF, Lucene and Word2Vec. Aspects shown for SURF are discovered by Author-ATS model.

4.6 Summary

In this chapter we have proposed a framework for finding supporting sentences to help a user get an idea of consensus when researching about an entity. To this end, we have developed a hierarchical topic model to jointly infer aspect-topic-sentiment for capturing an opinion expressed in a review sentence. We have also defined a fine-grained similarity measure considering both lexical and semantic similarity to retrieve similar sentences. Author-ATS model encodes the coherent writing style of a review by constraining the aspect distributions dynamically. It considers the sentiment distribution of a review to have influence of both the author and the entity. Experimental results on two real world review dataset indicate that the proposed approach outperforms existing techniques for opinion modeling as well for retrieving supporting opinions.

Chapter 5

Improving Usability of Health Information Online: Modeling User-Drug Interactions

5.1 Introduction

In the last decade, reporting health information online has become widespread via social networking sites (e.g., Twitter), health forums (e.g. WebMD, HealthBoards), health monitoring apps (e.g. Flaredown, Symple) and so on. People not only rely on online opinions for product purchases but also for information on different diseases and treatments. People increasingly search for health information, with 59% of the adult US population seeking health-related information online [25], and nearly half of US physicians relying on them for professional use [24]. Lately with the advent of health 2.0, users across the globe take a part actively by not only looking up health information online but also by self-reporting clinical experiences with treatments.

Traditionally, pharmaceutical companies carry out laboratory clinical trials and post-market surveillance, to discover side effects of drugs. However they are either limited in number or incur significant time delays to gather enough information [16, 100]. With the abundant amount of self-reported medical information available online, recently there has been to a surge of research interest to discover medical insights such as identifying potential side effects of drugs [88, 136, 55] from these resources. While these large amount of user-reported health information can help complement existing medical knowledge and

speed up discoveries of potential drug reactions [116, 126], there remains a widespread concern of whether the reported side effects are truly due to the drugs [14, 86].

Similar to the confounding factors explored in the domain of e-commerce in previous chapter (Chapter 3, Chapter 4), there could be some underlying factors that affect a patient's experience with a particular treatment and thus make the experienced side effects different for different people. Table 5.1 shows experiences with two drugs by different patients. From the sample reports it can be seen that while using the same drug, different people experience different symptoms with varying severity.

Treatments	**User**	**Severity Rating**	**Reported Symptoms**
Clonazepam	u1	3	decreased appetite, paralyzing anxiety
	u2	1	diarrhea
	u3	4	dizziness, nausea, vomiting, fatigue, tiredness
Levothyroxine	u4	4	nausea, dizziness, dissociation
	u5	3	weight gain
	u6	3	weight gain, hair loss, quivering, insomnia

Table 5.1: Sample symptom reporting by different patients for two treatments in Flaredown app.

In a realistic scenario, patients experience a set of clinical symptoms which could potentially stem from multiple **confounding factors**. This makes it difficult to claim if the symptoms are side effects of a drug, by a patient with little medical training. Furthermore, a patient is often under the influence of multiple drugs, and the experienced symptom could be a synergistic effect caused by a combination of the concurrent drug use, instead of being side effects of only one of them.

Our preliminary investigation on a real-world dataset shows that among the reported symptoms, there exists a significant percentage of *unsubstantiated*[1] side effects. Many of these symptoms are, in fact, more correlated to the underlying medical condition(s) of the user than the treatments. With more and more people seeking health-related information online [25], it is important that these sources provide accurate information tailored according to individual user's condition, to prevent unnecessary anxiety [5, 108]. This will help in reducing the number of users who might be reluctant to take a drug due to the long

[1] not associated with the drug as per expert medical knowledge

list of reported side effects associated with it, even though many of those are not applicable to her. This motivates us to develop a framework to better characterize the complex relationship between user–condition–treatment and personalize the prediction of possible symptoms and their severity for a specific user. This motivates us to develop a framework to better characterize the complex relationship between user–condition–treatment and personalize the prediction of possible symptoms and their severity for a specific user. Such a system would allow the patient to make an informed decision when choosing between alternate treatments, by weighing in the impact of potential side effects on the expected quality of life.

We formulate a multi-objective learner to predict both the set of symptoms and the severity rating that a user reports while being administered with a set of treatments. We design a novel deep neural network architecture called **M**ulti **o**bjective **M**ixture of **Ex**perts (MoMEx) to encode the complex relationship between user–condition–treatment combination and the target variable of symptoms. MoMEx uses a gating network inspired from the mixture of experts model [38, 42]. It probabilistically combines the predictions from three *local expert* networks that are built to predict symptoms based on user, her set of medical conditions, and a combination of treatments. The gating network has an added advantage in that we are able to use the probabilities assigned to each of the local experts to explain why the model predicts a certain symptom. This transparency of the predictive framework is crucial for a user to make a better health choice decision with confidence.

The key contributions of this work are as follows:

- Systematically investigating the nature of self reported symptoms in an online health tracking app and their correlation with the user and her pre-existing medical condition(s) apart from the treatment(s);
- Designing a multi-objective neural architecture, called MoMEx, for predicting symptoms and their severity score, based on the interaction between user, treatments, and conditions;
- Conducting extensive evaluation of MoMEx on a real-world dataset, to demonstrate

its effectiveness compared to state-of-the-art baselines and architectural variants.

The remainder of this chapter is organized as follows. We start with conducting an initial analysis on a real world dataset and formally defining our problem statement in Section 5.2. In Section 5.3, we proceed to describe the technical details of our proposed MoMEx framework. Section 5.4 presents the effectiveness of MoMEx in comparison to state-of-the-art baselines and summarizing our contributions and findings in Section 5.5.

5.2 Preliminaries

We first describe the dataset, highlighting different signals and conduct an initial analysis to illustrate the challenges and motivate our approach.

5.2.1 Dataset

We use a public dataset available on Kaggle[2] from the Flaredown (FD) app[3]. The app users can 'check-in' each day to record their treatment(s), and the experienced symptoms along with their severity scores (in the range of 0 to 4). Note that this also includes 'check-in' from users who did not experience any side effects for their treatment(s) and hence their list of side effects is nil and the severity score is 0.

The conditions, treatments and symptoms are pre-defined medical terms in the app, which users need to select from a drop-down list. Treatments are not necessarily prescribed drugs, but could also be alternative medicine or supplements, vitamins, physiotherapy, exercise and so on. For the severity rating, although the app allows users to report severity for each symptom, we assume the maximum reported severity in a 'check-in' to be the representative for the reported set of symptoms. We believe this assumption is reasonable since typically users report many (10 on an average) symptoms in a 'check-in', and might not meticulously note down the severity of each one of them. We filter out those symptoms and treatments which have been mentioned less than 5 times in the

[2] https://www.kaggle.com/flaredown/flaredown-autoimmune-symptom-tracker
[3] http://flaredown.com/

whole dataset. We collect the set of medical conditions mentioned by a user across all her 'check-ins'. Statistics of the dataset are shown in Table 5.2.

Number of treatments	1693
Number of users	3461
Number of unique conditions	1895
Number of unique symptoms	2521
Number of evaluations ('check-in')	14,879

Table 5.2: Statistics of the dataset.

5.2.2 Preliminary Study

To understand the nature of user reported symptoms, we first carry out an initial study to answer a few questions.

Q1. Can all user reported symptoms be substantiated by authoritative medical source as treatment side effects?

We compare the reported symptoms in the FD dataset with those published on the Mayo Clinic portal[4], which contains curated expert information about drugs and their side effects categorized into *common, less common*, and *rare*. For each treatment in the FD dataset, we obtain the set of all its symptoms reported in a 'check-in' across all users. Since a 'check-in' might mention multiple treatments, we associate a symptom to all the treatments mentioned in a 'check-in'. This ensures that even if the symptom occurred solely because of a single treatment, it is still considered as substantiated. Then we match the treatment name to a drug-family in the Mayo Clinic portal and consider the listed side effects as the ground truth.

Table 5.3 shows that only 33.29% of reported symptoms are known *common* side effects of a drug, while 18.76% and 7.60% are *less common* and *rare* side effects respectively. This indicates that comparatively lesser known side effects of a drug are indeed reported by users and their discovery can help augment the existing medical knowledge base. However, we also note that an alarming 40.35% of reported symptoms do not match

[4] mayoclinic.org/drugs-supplements/

with any known side effects of any of the administered drug. This motivates us to further analyze the reported symptoms for potential confounding factors.

Category	Percentage
Common	33.29%
Less Common	18.76%
Rare	7.60%
Unsubstantiated	40.35%

Table 5.3: Percentage breakdown of reported symptoms in the different categories of side effects for a drug.

Q2. Do the pre-existing conditions of a patient have any correlation to the symptoms she reports across drugs?

We analyze whether pre-existing conditions of a user influence the symptoms she experiences. For e.g., a patient suffering from *insomnia* may experience *fatigue* or *drowsiness*, and report them as side effects of her current treatment.

For each reported symptom in the FD dataset, we compare its association with various treatments to its association with various medical conditions. We define three sets of users:

- U_s : Set of users who have reported the symptom s
- U_c : Set of users who suffer from condition c
- U_t : Set of users who have taken treatment t

For each symptom s, we quantify its association with condition c and treatment t, using Jaccard similarity coefficient

$$J(s,c) = \frac{intersection(U_s, U_c)}{union(U_s, U_c)} \tag{5.1}$$

$$J(s,t) = \frac{intersection(U_s, U_t)}{union(U_s, U_t)} \tag{5.2}$$

We consider a symptom s is more correlated with a condition than a treatment, if there exists a condition c, for which $J(s,c) > J(s,t)$ for all $t \in T$, where T is the total number

of treatments in the dataset. We find that around 48.15% of symptoms are more correlated with a condition than with a treatment, indicating that the pre-existing conditions of a user are linked to the symptoms reported.

5.2.3 Problem Formulation

Our preliminary study shows that the reported symptoms could be possible side effects of one of the treatments, or be correlated with some underlying medical conditions of the user. This motivates us to propose an approach towards predicting the symptoms that a patient might report while administering a combination of treatments.

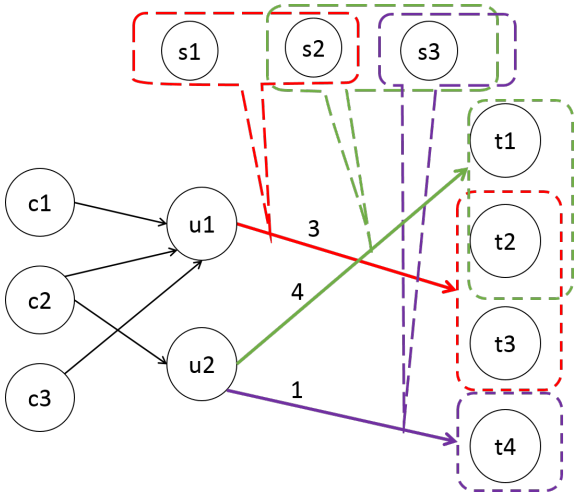


Figure 5.1: Interaction structure between user, her conditions, treatments, reported severity rating and symptoms

A graphical representation of the interactions is shown in Figure 5.1. Users $u1$,$u2$ have conditions $\{c1, c2, c3\}$ and $\{c2\}$ respectively . The check-ins of a user evaluate a set of treatments as shown by the directed edges. Each edge is labeled with a numeric score denoting severity, and a list of reported symptoms among $s1$, $s2$, $s3$. A sample evaluation point in the graph can be interpreted as, user u_1, suffering from conditions $\{c1, c2, c3\}$, has experienced symptoms $\{s1, s2\}$ with a severity of 3, while taking treatments $\{t2, t3\}$.

We formulate the problem as a multi-objective prediction task. For a user u and a set of treatments τ, we predict:

- **Severity of Symptoms:** a numerical rating $r_{u\tau}$, real-valued number in the range $[0, 4]$.

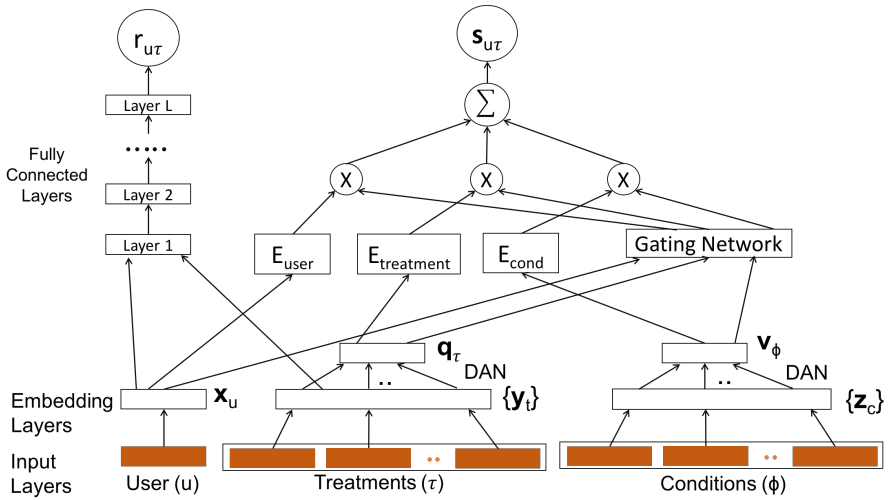


Figure 5.2: Proposed neural network architecture for severity rating and symptom prediction.

- **List of Symptoms:** a sparse S dimensional binary vector $\mathbf{s}_{u\tau}$, indicating the outcome symptoms where S is the total number of unique symptoms.

5.3 Proposed MoMEx Framework

We propose a neural network architecture, called MoMEx (Multi-objective Mixture of Experts), for predicting user reported symptoms along with their severity rating. The input signals to MoMEx are user, a set of treatments and her pre-existing medical conditions, as depicted in Figure 5.2.

We use three separate embeddings to map these inputs to a lower dimensional vectors of dimension k. Let $\mathbf{x_u}$, $\mathbf{y_t}$, $\mathbf{z_c}$ denote the latent feature vectors of user u, treatment t, and condition c respectively. Consider a user u, associated with a set of conditions ϕ, has evaluated a set of treatments τ in a 'check-in'. To encode these sets we employ deep averaging network (DAN) [37], which has proven to be a very effective modeling technique for un-ordered sequences. This helps us capture the dependencies between co-existing conditions (and simultaneous treatments).

We first embed each treatment $t \in \tau$ using treatment embedding to receive a collection of latent vectors $\{\mathbf{y_t}\}$. Then we take an average of the latent vectors of all the treatments in the treatment set (τ) to encode their combination. Thereafter, this vector is passed through

multiple feed-forward layers to capture more abstract representations of the concurrent treatments. The output of the last feed-forward layer gives us a k dimensional vector $\mathbf{q}_\tau$, denoting a latent representation of the combination of treatments. We similarly encode the set of conditions to a k dimensional vector, $\mathbf{v}_\phi$ denoting the set of pre-existing conditions of user u.

Given the latent representations of user, treatments and conditions, we now proceed to describe the prediction tasks.

5.3.1 Predicting Severity of Symptoms

Since the severity rating contains characteristics of both the user and the treatments, we combine the corresponding latent features by concatenating their embedding vectors $\mathbf{x_u}$ and $\mathbf{q}_\tau$. However, a simple concatenation is unable to capture the complex structure implied in the users' historical interactions. We add multiple fully connected layers on the concatenated vector introducing flexibility and non-linearity. The output of the last hidden layer L is transformed to a real valued rating $\hat{r_{u\tau}}$.

$$\hat{r_{u\tau}} = f(\mathbf{W_L h_{L-1}} + b_{L-1}) \tag{5.3}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the weight matrix and bias vectors and f is an activation function for which we use $tanh$. We obtained comparable results with $Relu$ and slightly worse results for $sigmoid$, as activation functions. We formulate this as a regression problem and the loss function is constructed as:

$$\mathcal{L}^r = \sum_{(u,\tau)\in\mathcal{X}} (r_{u\tau} - \hat{r_{u\tau}})^2 \tag{5.4}$$

where $\mathcal{X}$ represents the training set, $r_{u\tau}$ represents the ground truth rating and $\hat{r_{u\tau}}$ represents the predicted severity rating for treatment set τ by user u.

5.3.2 Predicting List of Symptoms

Now, we describe our approach for predicting the list of symptoms $\mathbf{s_{u\tau}}$, reported by user u for treatments τ. This is a sparse binary vector, where the m^{th} entry indicates whether the m^{th} symptom has been reported. We consider this as multiple individual binary classifications, which had been shown as an effective technique in the past [61], where the correlation among labels is exploited by the latent space in the model.

Motivated by our initial study (recall 5.2.2), we realize that the reported symptoms $\mathbf{s_{u\tau}}$, could be due to the treatments τ, or caused by the pre-existing conditions ϕ of the user u. Hence, we learn a model to predict $\mathbf{s_{u\tau}}$ given the embeddings of user, treatment set and a user's conditions i.e. $\mathbf{x_u}$, $\mathbf{q}_\tau$ and $\mathbf{v}_\phi$ respectively.

A plausible approach could be concatenating the vectors and using a multi-layer perceptron to get a binary prediction for each symptom. However, in such a network it will be difficult to rationalize why a certain symptom was predicted - whether it was because of the treatments or condition of the user or a complicated non-linear combination of them.

Inspired from the Mixture of Experts (MoE) approach [38, 42] we develop three simpler local experts namely, $E_{treatment}$, E_{user}, and E_{cond}, taking as input the treatment feature($\mathbf{q}_\tau$), user feature($\mathbf{x_u}$), and condition feature($\mathbf{v}_\phi$) respectively. However, unlike the usual MoE architecture, inputs to our experts are specific to only a single factor giving the experts a semantic meaning for their specialization and makes their decisions interpretable. The predictions from the local experts $E_{treatment}$, E_{user} and E_{cond}, are denoted as $\hat{\mathbf{s}}^{\mathbf{treatment}}_{\mathbf{u}\tau}$, $\hat{\mathbf{s}}^{\mathbf{user}}_{\mathbf{u}\tau}$, and $\hat{\mathbf{s}}^{\mathbf{cond}}_{\mathbf{u}\tau}$ respectively. The m^{th} entry of the vector $\hat{\mathbf{s}}^{\mathbf{treatment}}_{\mathbf{u}\tau}$ denotes the probability of occurrence of the m^{th} symptom according to the treatment expert classifier.

Finally, we need to combine the predictions from these individual experts to output a single prediction $\hat{\mathbf{s}}_{\mathbf{u}\tau}$. One way of doing that could be averaging their predictions, but that does not make sense for our scenario. When we average the output of multiple classifiers and try to match it to a target value, we force each of the classifier to compensate for the combined error made by the other classifiers. However, in our scenario, there are certain

symptoms that can be explained by only a single expert classifier (say, treatment) and we can ignore the results of the other classifiers for that case. This motivates us to develop a gating function where for each input an expert is selected with some probability. This is most similar to an ensembling approach [27], where the final prediction is a weighted average of the local expert predictions. The weights used for combining the expert predictions are the probabilities assigned by the gating function to the experts. The final prediction is a weighted average of the local expert predictions, where the weights are the probabilities assigned by the gating function to the experts.

For a particular input from user u for a set of treatments τ, the gating function takes as input the concatenation of user, treatment and condition vectors ($\mathbf{x_u}, \mathbf{q}_\tau$ and $\mathbf{v}_\phi$), and outputs a probability distribution, $\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{treatment}}$, $\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{user}}$, and $\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{cond}}$ for treatment, user and condition experts respectively. The final prediction is computed as

$$\hat{\mathbf{s}_{\mathbf{u}\tau}} = \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{treatment}} \cdot \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{treatment}} + \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{user}} \cdot \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{user}} + \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{cond}} \cdot \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{cond}} \tag{5.5}$$

where $\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{treatment}}, \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{user}}$, and $\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{cond}}$ are vectors of dimension S, the total number of symptoms. Since they denote probability distributions for weighting the three classifiers, for a particular symptom they sum to one i.e. for symptom $s \in \{1, \cdots, S\}$, we have

$$w_{u\tau}^{treatment}(s) + w_{u\tau}^{user}(s) + w_{u\tau}^{cond}(s) = 1 \tag{5.6}$$

where $w_{u\tau}^{treatment}(s)$ denotes the probability of selecting the prediction of the treatment expert classifier for the s^{th} symptom, others are defined similarly. The gating network helps us in examining our predictions more closely. For a symptom, we can look at the predictions of the three classifiers and their corresponding weights to understand the likely reason for it/

We now need to define the structures of our expert networks and the gating network. We choose similar structures, consisting of a stack of fully connected layers, for all three expert classifiers but with different parameters. The gating network multiplies its input with a trainable weight matrix and applies a $sigmoid$ non-linearity to convert it to a vector

of dimension S. This transforms the input from latent feature space to the symptom dimension. By multiplying this vector with a second trainable weight matrix, we transform the value in each symptom dimension, to a 3-dimensional vector representing the weights for each of the three experts. With $softmax$ activation on these vectors, its elements are converted to values in the range [0, 1] that add up to 1. We train the gating network by back-propagation, along with the rest of the model. Gradients are also back-propagated through the gating network to its inputs. We define this loss function as

$$\mathcal{L}^s = \sum_{(u,\tau)\in\mathcal{X}} \Big(\mathbf{w}_{\mathbf{u}\tau}^{\mathbf{user}} \cdot \text{BCE}(\mathbf{s}_{\mathbf{u}\tau}, \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{user}}) + \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{treatment}} \cdot \text{BCE}(\mathbf{s}_{\mathbf{u}\tau}, \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{treatment}}) + \mathbf{w}_{\mathbf{u}\tau}^{\mathbf{cond}} \cdot \text{BCE}(\mathbf{s}_{\mathbf{u}\tau}, \hat{\mathbf{s}}_{\mathbf{u}\tau}^{\mathbf{cond}}) \Big) \tag{5.7}$$

where $\mathcal{X}$ represents the training set, $\mathbf{s}_{\mathbf{u}\tau}$ represents the ground truth symptom vector of treatments τ by user u and BCE is the binary cross-entropy loss. A loss function like this will encourage specialization, since we are comparing the prediction of each expert separately with the target and then training to reduce the weighted average of all these discrepancies, where the weights are the probabilities of selecting the experts through the gating network.

5.3.3 Multi-Objective Learning

We integrate both the prediction tasks into a unified multi-objective framework with a weighted summation of the losses

$$\mathcal{L} = \sum_{(u,\tau)\in\mathcal{X}} \lambda_r \mathcal{L}^r + \lambda_s \mathcal{L}^s \tag{5.8}$$

where $\mathcal{L}^r$ and $\mathcal{L}^s$ are the losses for severity prediction and symptoms prediction respectively and λ_r and λ_s are the weights. In our experiments, we set them to be equal but one could vary them depending on which task is more important.

5.4 Evaluation

We carry out our experiments to evaluate the effectiveness of the proposed MoMEx framework.

We divide the dataset into training (80%), validation (10%) and test (10%) sets using five fold cross validation. The hyper-parameters are tuned via grid search on the validation set. The embedding dimensions are 64 unless otherwise mentioned. The number of fully connected layers, in the DAN for encoding the treatments and conditions is 2, in the rating predictor component is 3 for encoding the user-treatment interaction, in the local expert models and gating network are 3 and 2 respectively consisting of 500 neurons. We randomly initialized all model parameters with a Gaussian distribution (with mean 0 and standard deviation 0.01). The batch size for mini-batch training is 256 and the network is optimized using Adam[46] optimizer and is implemented using Keras[5]. The learning rate is set to 0.001.

5.4.1 Prediction of Severity Rating

We first evaluate our model on severity rating prediction and use the most popular metrics, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) for error measurement. We compare MoMEx with a number of state-of-the-art rating prediction models, namely, URP [3] , SVD++ [49] and FM [101].

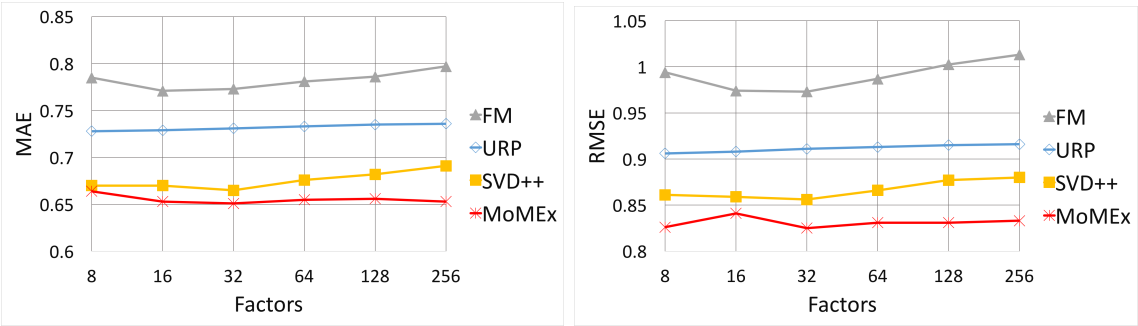
We also compared with other latent factor based models, namely NMF [54] and BPMF [106]. However they performed much worse compared to SVD++. Therefore, we consider SVD++ to be the representative of latent factor models and report its results. We use the librec[6] package for implementation of URP and SVD++. For Factorization Machine we use the libFM implementation [102] by the author.

Note that, in a traditional recommendation setting, a rating is available for a user-item pair. However, in our scenario, the severity rating is not always associated with a single treatment but with a set of treatments that a user has mentioned during the 'check-in'.

[5] `http://keras.io/`
[6] https://www.librec.net/

Therefore, for URP and SVD++, we consider each unique treatment-set to be an item. FM can consider any number of real valued features for making the prediction, therefore its input is similar to MoMEx.

The number of latent factor is important in determining a model's capability. We vary this in the range $\{8, 16, 32, 64, 128, 256\}$ and compute accuracy for competing models. For MoMEx, we vary the dimensions of latent user, treatment vectors as they are similar in spirit with the latent factors of a CF model for predictive capability [33].



(a) Mean Absolute Error

(b) Root Mean Square Error

Figure 5.3: Performance comparison for rating prediction with varying number of latent factors.

Figure 5.3 shows that our method consistently achieves the best performance. This shows that traditional rating prediction systems like SVD++ and URP are not as capable for handling scenarios where a rating is not for a specific entity but for a set of entities. Our framework is able to model that and can learn non-linear interactions between a user and item set through the use of hidden layers in the network. It outperforms the second best method SVD++ with a 4.07% and 3.09% improvement on an average, in terms of MAE and RMSE respectively. Furthermore, it is more robust to variations in number of latent factors, as SVD++ starts over-fitting with higher number of factors.

5.4.2 Prediction of Symptoms

The prediction of the list of symptoms reported by a user is a challenging task as the class distribution is highly skewed. In each 'check-in', only a few symptoms are reported among a huge list of symptoms. We use the standard precision, recall, and F1-score of the

positive class (i.e. of the reported symptoms) as evaluation metrics.

To understand the contribution of each of the input signals, we first perform an ablation study with our MoMEx model. We use different subsets of the three input signals, namely, user, treatment and a user's medical conditions, and note down the model performance. Table 5.4 shows the results. Unsurprisingly, MoMEx achieves the best F1-score when it takes into consideration all the three input signals, instead of taking a subset of them. This proves the necessity of modeling all the three contributing factors in symptoms reporting.

Input Signals for MoMEx	**Precision**	**Recall**	**F1-Score**
user + treatment	0.874	0.739	0.801
treatment + condition	0.836	0.728	0.778
user + condition	0.880	0.764	0.818
user + treatment + condition	**0.901**	**0.794**	**0.843**

Table 5.4: Performance of ablation study using different subset of input signals in MoMEx. MoMEx performs the best when it considers all three input signals.

We also compare MoMEx with the following baselines using other neural architectural variants:

- **Multi-Objective Multi Layer Perceptron(MoMLP) :** We replace the mixture of experts network with Multi Layer Perceptron. We concatenate the user-, treatment-, condition-latent vectors and use MLP layer to predict the list of symptoms. We experimented with $1-3$ number of fully connected layers for the MLP, and reported the best results.

- **Single Objective Mixture of Experts(SoMEx) :** We predict only the symptoms using a single loss function

Table 5.5 shows that using the Mixture of Experts gives superior performance compared to Multi-Layer Perceptron. Furthermore, using a single objective loss function results in a slightly worse performance compared to MoMEx. This indicates that the joint modeling of both the severity rating and symptoms using multi-objective learning benefits the symptom prediction task, as both of them essentially constitute a single 'check-in' by a user and are therefore connected. When a user gives a severity rating of 0, we learn that

the symptom experienced by this user is likely to be nil. On the other hand, when a user gives a high severity rating, the list of symptoms to be predicted is likely to be long.

Method	**Precision**	**Recall**	**F1-Score**
MoMLP	0.854	0.753	0.801
SoMEx	0.879	0.779	0.826
MoMEx	**0.901**	**0.794**	**0.843**

Table 5.5: Comparison among baseline neural architectures. All competitive models use all three input signals. MoMEx outperforms alternate baseline neural architectures.

Finally, we compare MoMEx with Gradient Boosting Machine, K Nearest Neighbour, and Random Forest classifiers. We use the implementations in scikit-learn python package [7] and XGBoost library [8] for the baselines. Table 5.6 shows that MoMEx clearly outperforms these methods.

Method	**Precision**	**Recall**	**F1-Score**
XGBoost	0.291	0.732	0.415
K Nearest Neighbor	0.821	0.580	0.679
Random Forest	0.815	0.660	0.729
MoMEx	**0.901**	**0.793**	**0.843**

Table 5.6: Comparison with state-of-the-art traditional ML classifiers. All competitive models use all three input signals. MoMEx outperforms all comptetitive traditional ML classifiers.

XGBoost achieves a comparable recall but at a very low precision, whereas K Nearest Neighbour and Random Forest suffer in recall due to the highly skewed distribution. MoMEx is able to alleviate this by exploiting the correlation between the label space (symptoms) using the weights of the shared hidden layers.

The labels are not always mutually exclusive but might be correlated with each other. For e.g. the occurrence of one side effect might influence another one as well (e.g. insomnia and headache). Therefore, in MoMEx the loss function is not a single cross-entropy that distributes the probability over the side-effects but individual binary cross-entropies. However, their predictions are not completely independent as they are connected to the outputs from the hidden layers of the expert and gating networks. These hidden layers

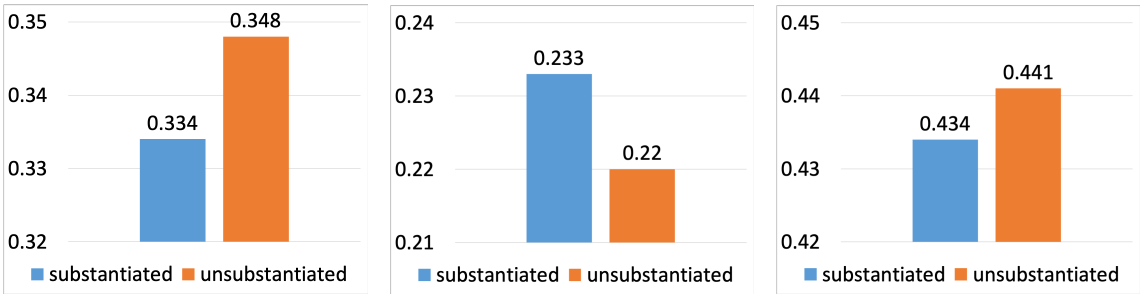
[7] http://scikit-learn.org/stable/index.html
[8] https://github.com/dmlc/xgboost

can learn to adjust the weights of the output according to the correlation among the labels. Using such hidden bottleneck layers and individual binary predictors have shown to be very effective for highly sparse multi-label classification for handling sparsity and correlation in the label space [61, 32].

5.4.3 Case Study

A major advantage of a mixture of experts framework is that the gating network outputs a probability distribution over the local experts, E_{user}, $E_{treatment}$, and E_{cond} built using user, treatment, and condition respectively. The prediction of a symptom for a given user is based on the weighted probabilities of these local experts. This distribution provides insight to the predicted symptoms.

As noted in our preliminary study of the dataset (in Section 5.2.2), while many of the reported symptoms are substantiated side effects of one of the treatments, a significant percentage of them are not substantiated. We first characterize the difference between probability distributions of substantiated versus unsubstantiated side effects. Figure 5.4 shows the average probability with which the predictions of the local expert models are weighted to generate the final prediction for for these two types of symptoms.



(a) E_{user} (b) $E_{treatment}$ (c) E_{cond}

Figure 5.4: Probabilities assigned to E_{user}, $E_{treatment}$ and E_{cond} for substantiated vs. unsubstantiated side effects

Firstly, Figure 5.4 shows that the probabilities assigned to E_{cond} are higher for both types of side effects. This is consistent with our preliminary observation that many of the side effects are correlated to users' medical conditions rather than to the their treatments.

From the weights assigned to $E_{treatment}$ (see Figure 5.4b), we observe that a higher

User	Conditions	Treatments	Predicted Symptoms	Local Experts Probability		
				E_{user}	$E_{treatment}$	E_{cond}
u1	Ehlers-Danlos syndrome, POTS	Bupropin	Out of breath	0.281	**0.373**	0.346
u2	Chronic fatigue syndrome, Crohn's disease, Hashimoto's disease	7-keto-dhea, Prednisolone, Vitamin D	Anxiety	0.285	**0.377**	0.338
u3	Anxiety, Depression, Eating disorders, Migraine	Prozac	Nausea	0.217	**0.572**	0.211
			Skin problems	0.356	0.180	**0.461**
			Pain	0.358	0.183	**0.459**
u4	ADHD, Acne, Depression, Insomnia	Adderall, Running	*Fatigue*	0.355	0.183	**0.462**

Table 5.7: Sample check-ins done by different users, the weights of each local expert networks assigned by the gating network. Symptoms in *italics* are deemed unsubstantiated side effects by expert medical knowledge base. Local experts with maximum probability (in bold) are the likely cause for the predicted symptoms.

probability is assigned in case of substantiated side effects vs. unsubstantiated ones. This is intuitive, since the side effects that are known to be associated with a treatment, will be reported by many users of the treatment and $E_{treatment}$ will be able to predict them reliably. In contrast, unsubstantiated side effects will rarely be reported by many users, resulting in $E_{treatment}$ being unable to model it, and it will be assigned a lower weight by the gating network. Interestingly, the opposite phenomenon is observed for E_{user} and E_{cond} (see Figure 5.4a and 5.4c). This indicates that users report some symptoms that are not associated with the administered treatments, and are more reliably predicted by user features(E_{user}) or her pre-existing conditions(E_{cond}).

Table 5.7 shows a few case studies of the symptoms correctly predicted by MoMEx and the corresponding probabilities of the local experts. We observe that most of the symptoms that are substantiated side effects of one of the treatments correspond to $E_{treatment}$, indicating that the symptoms are due to the treatment. In contrast, the unsubstantiated side effects (in italics) correspond to E_{cond}, suggesting that they are likely to be symptoms of users' pre-existing conditions.

For user $u1$, MoMEx predicted the symptom 'Out of Breath' and assigned the highest weight to $E_{treatment}$. This matches with $u1$'s reported symptom after taking Bupropion in his check-in. Similarly, for user $u2$, MoMEx predicted the symptom 'anxiety' with $E_{treatment}$ having the highest weight. Again, this prediction matches the $u2$'s check-in and

indeed, anxiety is a known side effect of Prednisolone. In contrast, user $u4$ suffers from insomnia and has reported experiencing *'fatigue'*. MoMEx is able to correctly predict this symptom and attribute it to the condition 'Insomnia'.

These demonstrate that analyzing the probability distributions of local experts generated from large scale user data is useful in interpreting reported symptoms, and could be of interest to both the web mining and medical communities.

5.5 Summary

We have systematically investigated the characteristics of user reported symptoms in an online platform. We find that users report diverse symptoms while undergoing similar treatments and a significant percentage of the symptoms could not be substantiated as side effects of the treatment. We study the confounding factor behind the reported symptoms and notice that the side effects experienced by a patient are often more correlated with his/her pre-existing medical conditions than with the treatments. This motivated us to view the symptom prediction problem as a personalized recommendation task tailored to individual users. To this end, we have proposed a novel neural architecture to predict user responses in terms of symptoms and severity rating. In order to predict both the severity rating and symptoms together, we have designed a multi-objective learner with a combined loss function for the two prediction tasks. Our MoMEx framework trains local experts based on user, condition and treatment, and thereafter probabilistically combines their predictions using a gating layer. Experimental evaluation on a real-world dataset shows that our approach is able to outperform state-of-the-art models on both prediction tasks and provide insights into its decisions.

Chapter 6

Improving Usability of Social Media: Detecting Rumor Veracity

6.1 Introduction

In this final chapter we focus on determining information reliability on one of the most widely used mediums for sharing user generated content i.e. social media. With millions of daily active users, social media platforms like Facebook, Twitter, Instagram, Snapchat etc., can provide a wide reach for user generated content within a short amount of time. Given their increasing penetration in our daily lives, uncensored news updates by media and individuals have become our main sources of information. It has been reported that more than 63% of social media users use Facebook and Twitter for their primary source of news [4]. This phenomenon is further reinforced by the ability of people to share and discuss interesting news stories with friends and family via the social media platform.

Whenever an event occurs we see a surge of posts on social media, with people sharing information or expressing their concerns or opinions regarding the event. In recent years we have seen a storm of fake news invading our social media networks during a major political event such as elections [8] or crisis situations like the Las Vegas shooting incident [117], or during natural disasters like Hurricane Florence [95], Hurricane Harvey [96] and so on. During emergency situations people tend to be more vulnerable and tend to retweet unverified posts, with a possible intention of being 'helpful' and sharing 'information'.

Due to the absence of any fact checking mechanism in-place, these posts could get widely circulated and cause panic among thousands of people, even though their reliability is not known.

We refer to such posts as rumors - a circulating piece of story with questionable veracity or truthfulness. There exists rumor debunking websites like `factcheck.org` or `snopes.com` that verify the truthfulness of rumors. These websites rely on social media observers to submit 'tips' on potential rumors which are then fact-checked by manual investigative journalism. This entails a long delay, during which a false story could get widely circulated and become disruptive. Therefore, it is necessary to build a tool that can automatically flag potentially false stories early, before they can affect a large number of people. Stories flagged by the tool can later be further investigated by manual journalism to verify and uncover the propaganda or intent behind spreading the false information.

It is challenging to identify a false story given only the textual content, since they are often created with the purpose of misleading the readers. Additionally, in contrast to our works in previous chapters on fixed domains, the topics of the posts in social media could be very diverse, making it harder. The initial work of [97] focused on identifying controversial posts from Twitter using regular expression based text features and a few network features. Increasingly advanced systems have since been developed using a wide range of hand crafted features [9, 129, 115, 62]. These approaches leverage user features derived from their demography, followers, posting and re-tweeting behavior, textual features from the text of the posts, and external knowledge from shared links to external sources. However, designing and maintaining these wide range of features from the rich and evolving information present in social media is non-trivial.

We propose to employ the wisdom of the crowds i.e. the users of the social media platform to help us identify rumors early. We argue that when a news item starts spreading over social media, peoples' reactions to it contain clues to its truthfulness. Different people react to the same story differently. While a lot of people could be vulnerable and share a story without verifying it, there might be some people who are *skeptical*. On a controversial post, a few people might try to question its authenticity, or ask for

more information or point out discrepancies. These enquiries or skepticism often trigger a discussion that motivates people to look for facts and evidences from external sources to able to either support or deny the story. We aim to model such opinions, and arguments put forward by people in order to resolve the veracity of a rumor.

Figure 6.1 shows a sample conversation on Twitter starting with a tweet mentioning a rumor that spread during the Sydney hostage crisis in 2014. The first tweet in the conversation, hereafter referred to as source tweet, claimed ISIS involvement in the incident. The subsequent tweets *reply* either directly to the source tweet, or to other tweets in the conversation. Each tweet in a conversation can be of type *support, deny, query* or *comment* depending on its stance towards the rumor. There is a veracity label for the whole conversation indicating whether the rumor is *true, false* or *unverified*.

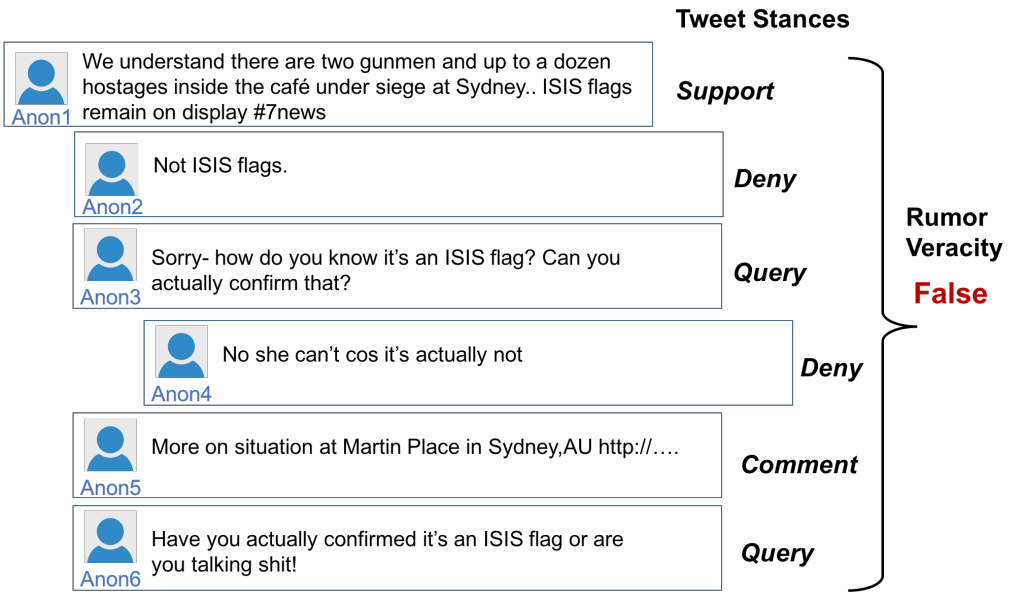


Figure 6.1: Sample tweet conversation structure on a rumor claiming ISIS involvement in an attack in Sydney.

In this chapter we design a two-step solution to detect the veracity of rumors.

1. Identify the *stances* of all the tweets engaging in a conversation about the rumor. To this end, we design a novel neural architecture for predicting the stance of a tweet that considers the conversation tree structure, i.e., the textual content of the target tweet, its timestamp, as well as its context.

2. Determine the veracity of a rumor by aggregate the individual tweet stances of users

using a neural network architecture.

The stance prediction component leverages the discourse around a rumor by detecting how users *react* to it in forms of direct/indirect replies. While a person may render direct *support* or outright *deny* a rumor, often people *comment* on a possible rumor tweet with additional information or ask for more information through *queries*. It is important to correctly identify these stances, since their distributions can be distinctive for different classes of rumors e.g. false rumors tend to evoke a lot more *deny* and *query* tweets than a rumor which is true. To this end, we design a novel neural architecture for predicting the stances that considers the conversation tree structure. To predict the stance of tweet in a conversation tree, the model considers three signals, namely , (1) the textual content of the target tweet, (2) its timestamp, and (3) its context.

To encode the textual content of a tweet, we employ convolutional neural networks inspired by their recent success for natural language processing tasks [45, 131, 69, 28, 140]. We further augment it with attention layer to help the network focus on parts of a tweet that are important for identifying its stance. However, due to the short and conversational nature of tweets, using only the tweet text is often not sufficient to understand its stance, e.g., the tweet "*No she can't cos it's actually not*" in Figure 6.1. Since this is in response to an ongoing conversation, looking at its preceding tweet "*Sorry- how do you know its an ISIS flag? Can you actually confirm that?* ", makes its stance clear. We account for the context of a target tweet by taking into consideration all its preceding tweets in the reply chain of a conversation tree. The sequential nature of conversation is captured through a recurrent neural network (RNN) due to its superiority in handling sequential data. Additionally, we observe that not all tweets preceding a target tweet is equally important in understanding its stance. Therefore, we include a tweet-level attention mechanism to help the RNN focus on the relevant parts of the conversation.

To the best of our knowledge, this is the first work that uses two-level attention over textual content as well as at the tweet-level to encode the conversational nature of a tweet in order to understand its stance and in turn predict rumor veracity.

After predicting the stances of all the tweets in a conversation tree, we aggregate the

predictions along with the textual contents of the tweets to determine the rumor veracity. We analyze several methods of combining the stance prediction component with the veracity prediction step. We optimize the combined network using a transfer learning approach with full fine tuning of the weights learned in the first step. Experimental results on a real-world dataset from Twitter show that our approach significantly outperforms other competitive methods for both stance prediction and veracity classification.

6.2 Preliminaries

We use a real world dataset collected from Twitter and published as part of the PHEME project [17]. This dataset is subsequently used in a SemEval 2017 task [18]. The data contains online conversations on Twitter, each pertaining to a particular event and the rumors around it.

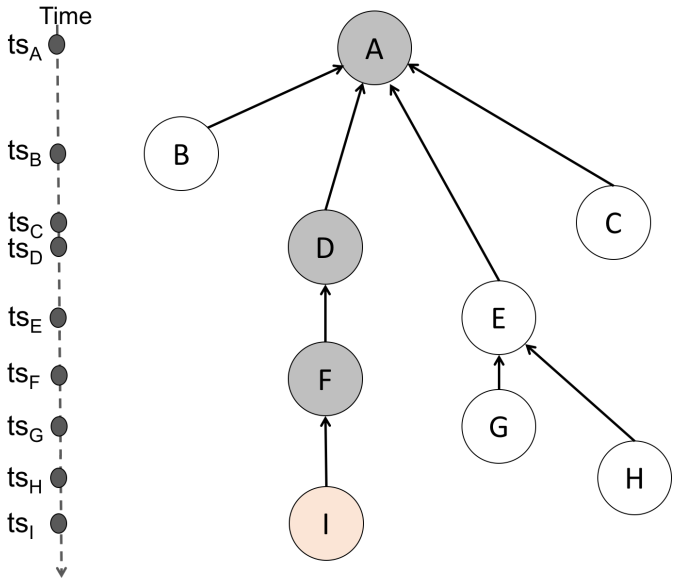


Figure 6.2: Sample tweet conversation tree.

Each conversation in the dataset forms a tree T as shown in Figure 6.2. The root node A is a *source tweet* that initiated the discussion. A directed edge denotes the reply of a tweet to its parent tweet. Each tweet is associated with a timestamp at which it has been posted e.g., tweet C is posted at ts_C. The *conversation sequence* of a tweet is defined by the chain of tweets starting from its parent and going all the way up to the source tweet.,

e.g., the conversation sequence for tweet I is $\{A, D, F\}$.

	False	True	Unverified
Comment	63.26	63.86	65.32
Support	18.93	22.18	18.46
Deny	11.71	5.99	7.52
Query	6.10	7.96	8.70

Table 6.1: Stance distribution of tweets in conversation trees of different types of rumors.

To understand the importance of people's stances in determining the veracity of a rumor, we first look at the distribution of stances of tweets concerning rumors of different veracity. As we can see from Table 6.1, the distribution of stances for different types of rumor are quite discriminating. For example, number of *support* tweets are higher for a true rumor whereas higher number of *deny* tweets are sparked for a rumor which later turned out to be false. Rumors that remained unverified have a greater percentage of query tweets.

6.3 Proposed Solution

Motivated by our observation of the discriminating stance distribution for different types of rumors, we design a two-step solution that takes into consideration the **C**onversation **T**ree structure. The first step predicts the stances of individual tweets via CT-Stance. The second step aggregates the predicted stances via CT-Veracity.

6.3.1 Stance Prediction

We consider three signals for a target tweet: textual content, conversation sequence, and the timestamp that the tweet is posted. Figure 6.3 shows the overall architecture for our stance predictor model CT-Stance.

Each tweet is first encoded by a CNN-based text encoder, and an RNN-based conversation sequence encoder is used to represent the context of the target tweet. The encoded representations of the signals are thereafter used for prediction. Details of the network components are given below.

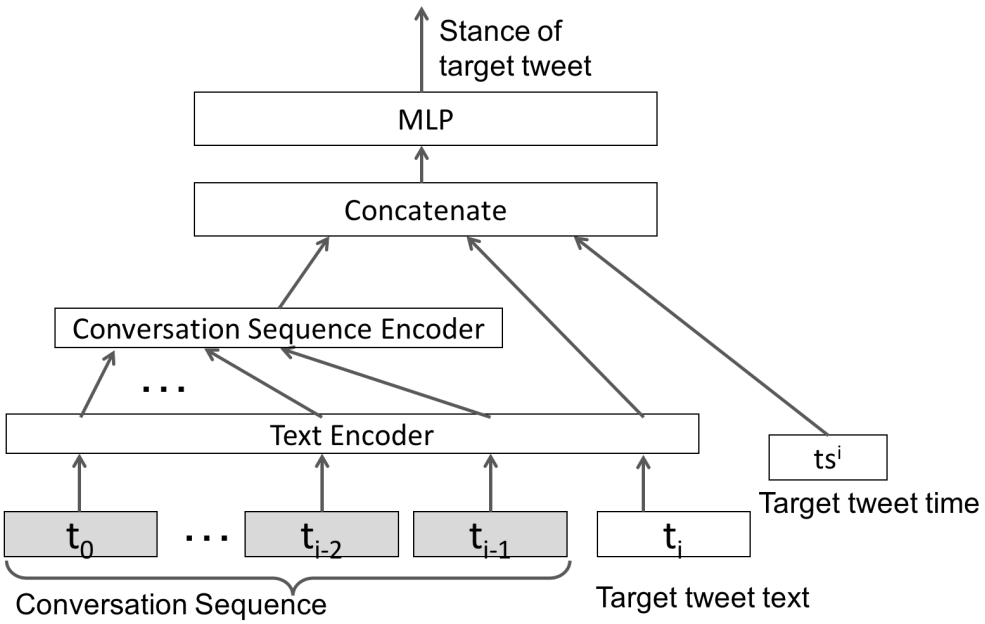


Figure 6.3: Overall architecture of CT-Stance model.

Tweet Text Encoder We first encode the text of an individual tweet t, denoted as a collection of words $t = \{w_1, w_2, \cdots, w_n\}$. Figure 6.4 shows the text encoder.

Each word is embedded in a lower dimensional space so that a tweet is now represented as a sequence of word vectors $\{\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_n}\}$ where $\mathbf{v_i} \in \mathbb{R}^d$. We initialize the word vectors using pre-trained Glove embeddings [84] but tune it during training to capture the intrinsic features of the specific task at hand.

We apply a one dimensional convolution followed by a $tanh$ non-linearity on the sequence of word vectors. The convolutional kernel is parameterized by $\mathbf{W} \in \mathbb{R}^{d \times l}$, $b \in \mathbb{R}$ where d is the dimension of a word and l is the filter length. It processes l consecutive word vectors and maps them to a single output which can be used as a feature. For example, a feature c_i is generated from a window of words $\mathbf{v_{i:i+l-1}}$ by

$$c_i = tanh(\mathbf{W}.\mathbf{v_{i:i+l-1}} + b) \tag{6.1}$$

The kernel slides over the embedded vectors of each *l-gram* and produces a map of features $\mathbf{c} = [c_1, c_2, ..., c_{n-l+1}]$ as the output. The output is padded to make its length the same as the input length i.e. n.

Traditionally, a standard max-over-time pooling operation [15] is performed over

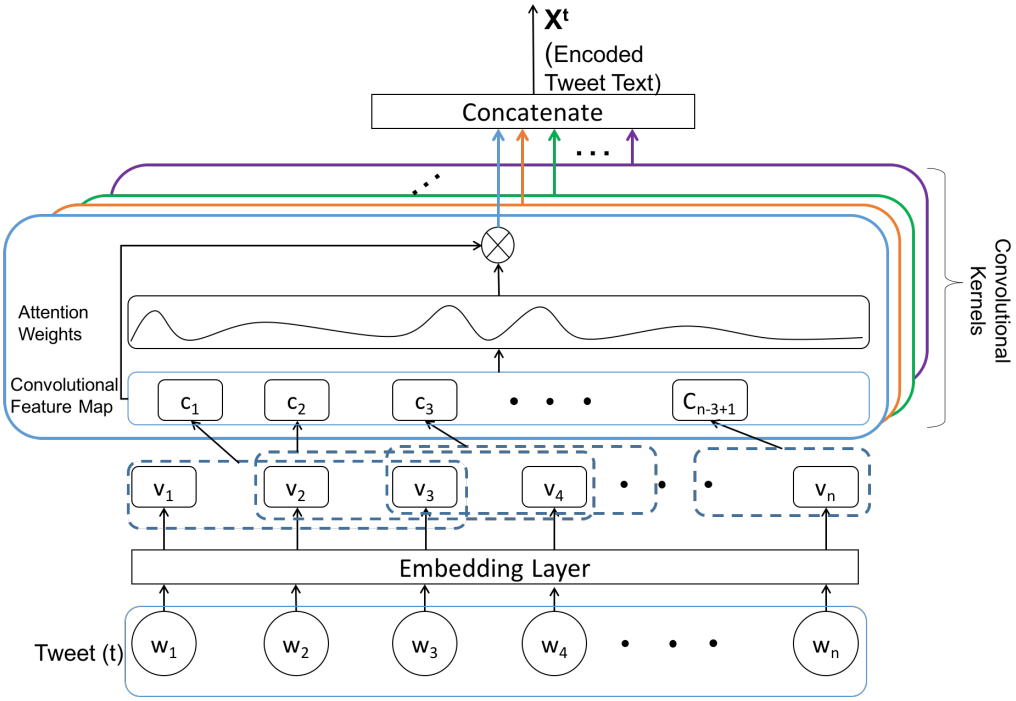


Figure 6.4: Architecture of the Text Encoder component.

the feature map to produce the single most important feature. However, multiple non-consecutive sections of a tweet could be important in understanding its stance, making max pooling insufficient.

In order to identify the parts of a tweet that are important in determining its stance, we use a self-attention [60] mechanism over the output of the convolutional layers. For each *l-gram* $\mathbf{v_{i:i+l-1}}$, we compute a weight a_i to determine the contribution of its corresponding feature vector c_i to the stance of the whole tweet and get an attention vector $\mathbf{a} = \{a_1, a_2, \cdots, a_n\}$ as

$$\mathbf{a} = softmax(tanh(\mathbf{W}.\mathbf{c})) \tag{6.2}$$

The tweet representation for a kernel j is computed as:

$$x_j = \sum_{i}^{n} a_i c_i \tag{6.3}$$

We use three different filter lengths ($l \in \{2, 3, 4\}$) and 128 such kernels for each filter length to detect multiple features and concatenate all extracted features to get the final tweet text representation denoted as $\mathbf{x^t}$.

Conversation Sequence Encoder Next, we encode the preceding tweets in the conversation sequence of a target tweet t by using a bi-directional RNN. The input to the bi-directional RNN is the encoded tweet text representations $\{\mathbf{x^1}, \mathbf{x^2}, \cdots, \mathbf{x^{t-1}}\}$. Figure 6.5 shows the conversation sequence encoder.

The RNN reads the sequence in left to right direction in the forward pass and creates a sequence of hidden states $\{\mathbf{h_f^1}, \mathbf{h_f^2}, \cdots, \mathbf{h_f^{t-1}}\}$, where $\mathbf{h_f^i} = RNN(\mathbf{x^i}, \mathbf{h_f^{i-1}})$ is a function for which we use a GRU [13]. In the backward pass, it reads the input sequence in reverse order and returns a sequence of hidden states $\{\mathbf{h_b^{t-1}}, \mathbf{h_b^{t-2}}, \cdots, \mathbf{h_b^1}\}$. The forward and backward hidden states are then concatenated to create the encoded hidden state of a context tweet $\mathbf{h^i} = [\mathbf{h_f^i}; \mathbf{h_b^i}]$ considering all its surrounding tweets.

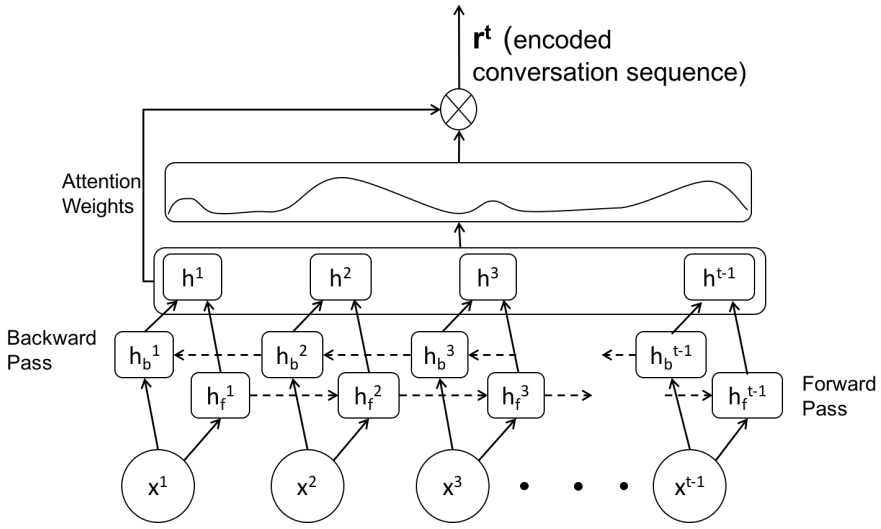


Figure 6.5: Conversation Sequence Encoder

We experimented with replacing GRU by LSTM [34] which resulted in similar performance at the cost of longer training time due to larger number of parameters. We use stacked bi-directional GRUs where the output hidden states of a layer are fed as input sequence to the next layer. The output of the last such layer is considered as the feature vector for the context of the target tweet.

We apply a tweet-level attention over the conversation sequence to focus on the important tweets in the conversation. We compute the context attention weights a_i^c for the feature vector $\mathbf{h_i}$ corresponding to each tweet in the conversation sequence. The attention vector $\mathbf{a^c} = \{a_1^c, a_2^c, \cdots, a_{t-1}^c\}$ is then multiplied with the corresponding features $\mathbf{h_i}$, and

a weighted sum is calculated (similar to Equation 6.3) to get the context representation $\mathbf{r^t}$.

Temporal Signal Encoder. The time elapsed since the source tweet could influence the type of response tweets it generates. For example, when an unverified news emerges, people typically voice their opinions from pre-conceived notions and the limited evidences available at that time to *support* or *deny* the claim. However, as time progresses and more evidences come in, we observe that people try to reason and evaluate the repercussions of the event by *commenting* on earlier tweets with posts like 'why this outrage let's calm down' , 'no one would care if a white kid was shot but now people care because he is black', 'maybe he left his Taser in the car and so he used his gun' and so on.

To study this observation further, we plot the percentage of tweets belonging to the majority two stance classes (*comment* and *support*) arriving within varying time windows. Figure 6.6 shows that as more time elapses since the source tweet, the percentage of reply tweets commenting on the rumor increases while the percentage of support decreases. This motivates us to use the temporal information as a signal in our network. For a target tweet t, we encode its temporal feature ts^t as the difference (in seconds) between the posting time of the source tweet and that of the target tweet.

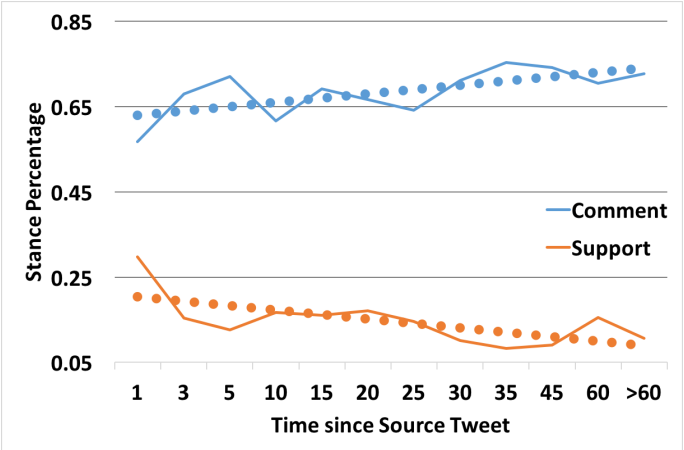


Figure 6.6: Distributions of tweets belonging to *comment* and *support* class over time. Dotted lines show the trends that with time *comment*s increase while percentage of *support* decreases.

CT-Stance Predictor Given a target tweet t, we concatenate its text representation $\mathbf{x^t}$, its context representation $\mathbf{r^t}$, and temporal feature ts^t to form the final tweet representation $\mathbf{z^t} = [\mathbf{x^t}; \mathbf{r^t}; ts^t]$. The vector $\mathbf{z^t}$ is fed through stacked fully connected layers and the

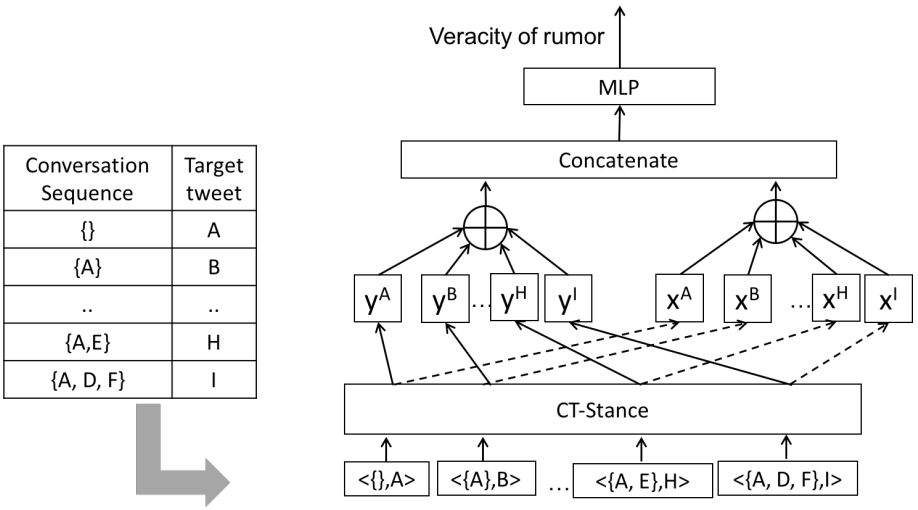


Figure 6.7: Architecture of CT-Veracity. Each row in the table shows the conversation sequence for a target tweet from the conversation tree.

output of the last layer is passed through a *softmax* layer to output a probability distribution over the four stance classes.

$$p(\mathbf{y^t_{stance}}|\mathbf{z^t}) = softmax(\mathbf{W}.\mathbf{z^t} + \mathbf{b}) \tag{6.4}$$

where $\mathbf{y^t_{stance}}$ is the probability distribution over the four stance classes for the tweet t. The model is trained using cross-entropy loss function and optimized with Adam optimizer [47].

6.3.2 Veracity Prediction

In order to classify the veracity of a rumor, we take as input a complete conversation tree T (recall Figure 6.2). Based on the conversation tree, each individual tweet t (its textual content and timestamp) and its conversation sequence is first fed through CT-Stance to obtain the probability distribution over stances for each tweet, denoted as $\mathbf{y^t_{stance}}$. Figure 6.7 shows the architecture of the veracity classification model CT-Veracity.

The probability distribution over four stance classes of individual tweets are then averaged to obtain a probability distribution over stances for the complete tree.

$$\mathbf{y^T_{stance}} = \frac{1}{|T|} * \sum_{t \in T} \mathbf{y^t_{stance}} \tag{6.5}$$

where $|T|$ denotes the number of tweets in T.

Apart from the output stance probability distribution, the stance predictor component learns a tweet text representation $\mathbf{x^t}$ for each tweet t in T. We combine these individual tweet representations to form a textual representation of T by taking an average,

$$\mathbf{x^T} = \frac{1}{|T|} * \sum_{t \in T} \mathbf{x^t} \tag{6.6}$$

Thereafter, the stance distribution and textual representation of the tree are concatenated and fed through a fully connected layer with softmax to predict the veracity of the rumor discussed in the conversation tree.

$$\mathbf{y^T_{veracity}} = softmax(\mathbf{W}.[\mathbf{y^T_{stance}}; \mathbf{x^T}] + \mathbf{b}) \tag{6.7}$$

where ; denotes concatenation operation and $\mathbf{y^T_{veracity}}$ is the probability distribution over three veracity classes.

Now we move on to address the coupling of CT-Stance into the architecture of CT-Veracity model. We first note that the data for the veracity prediction task is considerably smaller than the stance prediction task, since there is a single veracity label for a whole conversation tree in contrast to a label for each tweet stance. To overcome this challenge we adopt a transfer learning approach for training CT-Veracity.

In transfer learning, a base network is trained first, and then the learned features are reused or *transferred* to a second network to be trained on a target task. It has proven to be a powerful learning tool when the target dataset is much smaller compared to the base dataset. For neural networks, the weights of the first n layers from a pre-trained base network are copied to the first n layers of the target network and the remaining layers of the target network are initialized randomly.

Following this principle we pre-train our base network (CT-Stance) and copy the corresponding layer weights to our target network (CT-Veracity). While training CT-Veracity, we backpropagate the error into the transferred features from CT-Stance as well, essentially *fine-tuning* them.

6.4 Experiments

We carry out a comprehensive set of experiments to evaluate our proposed solution. We use the online Twitter conversation dataset of the SemEval 2017 Challenge [18]. The training dataset consists of tweets spanning eight events such as the 'Charlie Hebdo shooting in Paris', 'The Ferguson unrest in the US', and 'The GermanWings plane crash' etc. The test data consists of conversation trees related to some events from the training set as well as two unseen events. We report the results after averaging five runs on the test set.

6.4.1 Evaluation of CT-Stance

We start with evaluating the accuracy of our stance prediction network CT-Stance. For each tweet in the dataset, it outputs a stance among four classes and we use *accuracy* as the metric for evaluation.

We first compare CT-Stance with the following state-of-the-art neural stance prediction models that consider different input signals:

- CNN [11]. This method uses a convolutional neural network on the target tweet text to predict its stance.
- Branch-LSTM [48]. This method uses the entire conversation tree for predicting stances of each of its nodes.
- CT-Stance^{-}. This is the same as CT-Stance except that the temporal signal is not used. In other words, it only considers the target tweet text and the conversation sequence.

Table 6.2 shows the results. We observe that considering the target tweet as well as the conversation sequence is important in understanding the discourse properly and predicting its stance. Incorporating the temporal information helps in boosting the performance further.

We note that, although the branch-LSTM [48] obtains a competitive score, it uses some input signals which might not be available in a real-time system. In order to predict

Model	Input Signals	Accuracy
CNN [11]	Target tweet text	70.06%
Branch-LSTM [48]	Entire conversation tree (includes *future tweets*)	78.4%
CT-Stance$^-$	Target tweet text, conversation sequence	78.02%
CT-Stance	Target tweet text, conversation sequence, time	**79.86%**

Table 6.2: Comparison of Stance Prediction Models that consider different subsets of input signals. Our model CT-Stance achieves the best performance when considering all three realistically available signals.

the stance of a tweet, it looks up *all* the tweets in the tree, including the ones posted in the *future* with respect to the target tweet. On the other hand, our model only uses the preceding tweets in the conversation sequence for predicting stance of a target tweet, which is more realistic. From the results, we can observe that in comparison to branch-LSTM, our model achieves comparable scores using only the realistically available conversation sequence (CT-Stance$^-$) and outperforms using temporal information (CT-Stance).

Next, we investigate the effectiveness of the various components in CT-Stance by implementing the following variants:

- Text Encoder + Concatenation. We use the convolution layers as the text encoder and concatenate the hidden text representations to form the conversation sequence.
- Text Encoder with Attention + Concatenation. We use the convolution layers with attention as text encoder and concatenate the hidden text representations to form the conversation sequence.
- Text Encoder + Conversation Encoder. We use convolution layers as text encoder and use 2 layers of stacked Bidirectional GRU as conversation sequence encoder.
- Text Encoder with Attention + Conversation Encoder with attention. We use the convolution layers with attention as text encoder and use 2 layers of stacked Bidirectional GRU as conversation sequence encoder.

For fair comparison, the final prediction layers and the input signals for all the variants are kept identical.

Table 6.3 shows the results. As we can see from the results, encoding the conversation sequence properly using bidirectional GRUs produces a huge improvement over a simple

Variants of CT-Stance	Accuracy
Text Encoder + Concatenation	72.21%
Text Encoder with Attention + Concatenation	74.35%
Text Encoder + Conversation Encoder	77.50%
Text Encoder with Attention + Conversation Encoder	79.17%
CT-Stance (Text Encoder with Attention + Conversation Encoder with Attention)	**79.86%**

Table 6.3: Performance of architecture variants of CT-Stance. Using a sequence encoder for the conversation greatly improves the accuracy compared to simple concatenation. The model achieves the best scores with the use of attention at both text and tweet levels.

concatenation. This is in line with most of the recent works that have found the efficacy of RNNs in sequence representation across domains. We also note that using attention mechanism further boosts the performance by enabling the model to concentrate on important parts for stance prediction.

6.4.2 Evaluation of CT-Veracity

Veracity is a three class (*true, false, unverified*) classification task and we use accuracy as its performance metric.

We compare CT-Veracity with the following state-of-the-art rumor detection approaches:

- GRU-2 [65]. This uses two stacked GRU layers to encode the sequence of textual contents of tweets being posted about the rumor.
- CAMI [134]. This uses convolutional neural network to encode consecutive tweets of an event.

These works do not consider the proper conversation tree structure of tweets and considers all posts related to a rumor in a linear fashion ordered by their timestamps. We also note that these works have modeled the tweet texts directly for veracity prediction, without considering a tweet's stance, unlike our approach. Therefore, we also design the following baseline to investigate if knowing the ground truth stances of tweets helps improve the accuracy of veracity prediction.

- Bi-GRU-2. This is a baseline that considers only the sequence of ground truth stances for the tweets and use two layers of stacked Bidirectional GRU to encode

it. This baseline demonstrates the rumor detection accuracy achievable by only considering stances of tweets.

Model	**Input Signals**	**Accuracy**
GRU-2 [65]	Tweet texts	45.85%
CAMI [134]	Tweet texts	50.0%
Bi-GRU-2	Tweet stances (ground truth)	50.57%
CT-Veracity	Tweet texts, tweet stances (predicted by CT-Stance)	**57.14%**

Table 6.4: Comparison of Rumor Veracity Prediction Models. This demonstrates the effectiveness of tweet stances in determining a rumor's veracity. Our CT-Veracity model achieves the best performance compared to the state-of-the-art rumor detection methods.

We make two key observations from the results shown in Table 6.4. Firstly, we observe that the ability to detect rumors is greatly benefited by directly considering the stances of tweets compared to only its textual contents as demonstrated by baseline Bi-GRU-2. Secondly, CT-Veracity model outperforms the competitive methods comfortably by considering both the stances as well as the tweet contents.

As the CT-Veracity model considers the predicted stances of tweets, the accuracy of the CT-Stance model and their coupling plays an important role in determining the overall accuracy.

In the next set of experiments, we investigate how the coupling strategy can influence the CT-Veracity model performance by evaluating multiple alternatives.

We consider the following,

- Pipeline model. We train the CT-Stance model first. Thereafter, we use the predicted stances from it, and the encoded text representations of the tweets from the text encoder component as input to CT-Veracity.

- Joint model. We train a single model using a multi-objective loss function that optimizes both the stance prediction and veracity prediction tasks together. In this approach since both the tasks are optimized together they can learn from one another.

- Transfer learning with frozen weights. We train CT-Stance first and copy the weights of the corresponding layers in the complete model. The weights of the text encoder

component are kept frozen during the training of CT-Veracity. This is a prevalent practice for training on smaller datasets, to avoid learning those parameters which are already learnt well in another task in order to avoid overfitting [133].

Method	**Accuracy**
Pipeline model	41.23%
Joint model	44.45%
Transfer learning with frozen weights	50.15%
CT-Veracity (Transfer learning with fine tuning)	**57.14%**

Table 6.5: Performance of Variants of CT-Veracity.

Table 6.5 shows the results. We firstly observe that the transfer learning based approaches outperform both the joint and the pipeline model. This is due to (i) the dependency between stance prediction and veracity prediction tasks, and (ii) imbalance between dataset sizes. For the joint model, the network tries to optimize both objectives together, and learns a sub-optimal stance prediction model possibly due to overfitting on the veracity prediction task. In the pipeline model, since CT-Stance is trained independently, the overall stance prediction accuracy is the best. However, the recall for the under-represented stance classes (*query, deny*) are lower than the majority classes (*comment, support*). This affects the veracity prediction accuracy since they are the most discriminative classes for determining rumors (as shown in Table 6.1).

On the contrary, as the transfer learning with fine tuning approach is able to change the weights in stance prediction component, the overall accuracy of the stance prediction component decreases slightly but recall for the other three classes increase significantly. This helps in achieving high accuracy for veracity prediction. We note that transfer learning with fine tuning outperforms its counterpart with the frozen weight. Similar phenomenon have been observed before in different domains [19]. By freezing the transferred weights, it becomes non-trivial for gradient-descent to optimize a network that has been split in-between. This can be attributed to task-specific co-adaptation of neighboring layers [133].

6.4.3 Case Study

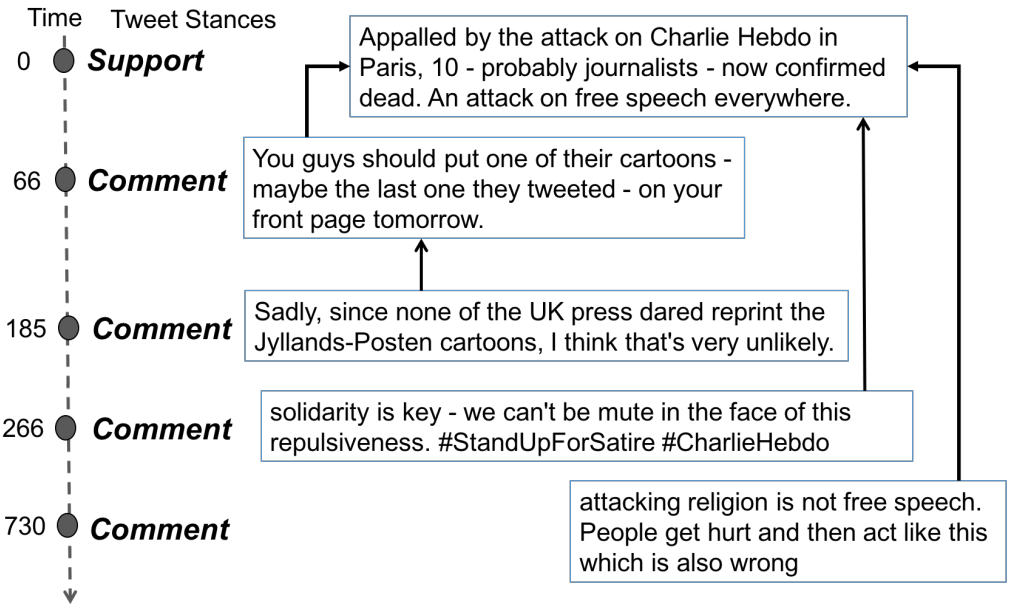
Finally, we present a study for different cases of rumor detection successfully handled by our model. Figure 6.8 shows the conversation trees within the first few minutes for two different types of rumors.

In Figure 6.8(a), a rumor regarding 'Charlie Hebdo shooting in Paris' is presented. We observe that the responding tweets mostly are expressing solidarity or voicing personal opinions, but are not raising questions regarding the event. Hence our model predicted its veracity to be *true*.

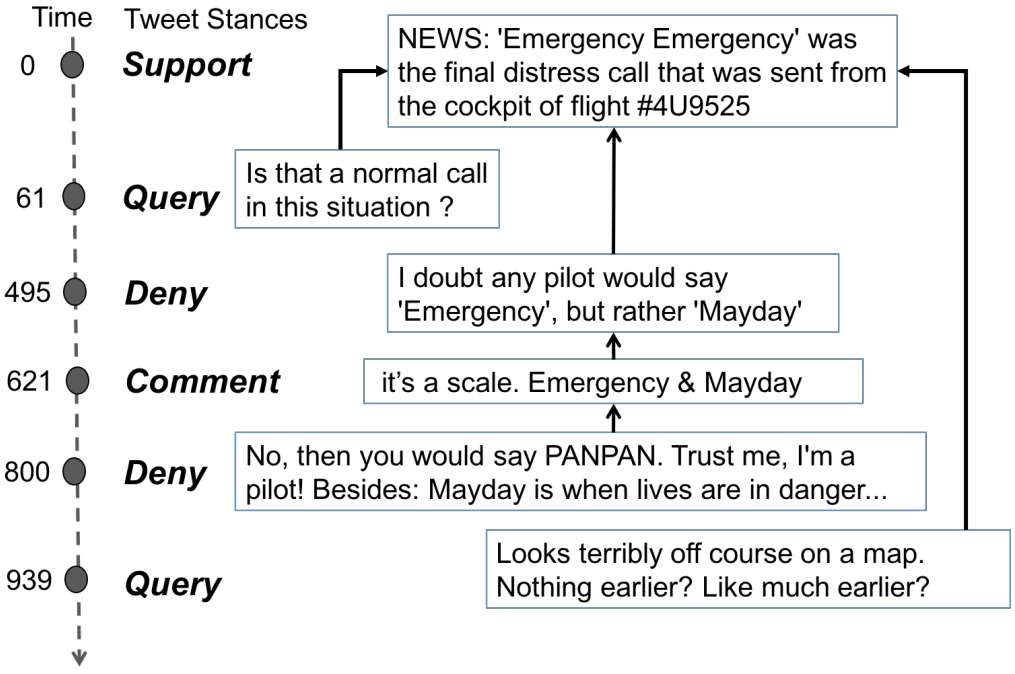
In Figure 6.8(b), a *false* rumor about the 'final distress call from Flight 4U9525' is depicted. We observe that some people respond by expressing doubts regarding the nature of the distress call mentioned in the source tweet. These initiate a conversation where people start pointing out the inconsistencies in the reported information invoking further queries and denials. Considering this conversation structure and the presence of many *deny* tweets our model successfully predicts it to be a false rumor.

6.5 Summary

In this work, we have examined the problem of rumor detection from analyzing the conversations sparked around an event on social media. To this end, we have designed a neural network architecture that captures the stances of peoples' posts towards the rumor and accumulates them in order to predict its veracity. We employ convolution with attention mechanism to encode a tweet's textual content and use RNN with tweet-level attention mechanism to capture the conversation sequence. Experimental results on a real-world Twitter dataset demonstrate that our stance prediction model outperforms state-of the art models. Additionally, coupling the stance prediction model with the veracity classification model using transfer learning with full fine tuning achieves significant improvement over state-of-the-art rumor detection methods.



(a) True Rumor



(b) False Rumor

Figure 6.8: Illustration for conversation trees for two rumors within the first few minutes. The unit of time is in seconds on the time-line.

Chapter 7

Conclusion

7.1 Conclusion

In this thesis, we have studied the problem of reliable use of online user generated content that are present in various forms. While user generated contents are becoming increasingly popular and widespread, there is a need for systematic studies on automatically analyzing and modeling their information content for downstream use.

In our first work, we focus on user ratings for different aspects of an item such as products or hotels or restaurants. We observe that the observed ratings are often obfuscated by the aspect biases of the individual users and do not reflect the true quality of an item. We develop a probabilistic modeling framework to capture these latent aspect biases (or preferences) of users that affect their ratings. We perform experiments on two large real-world datasets to show that such biases indeed exist, and our model is able to capture them well. Knowing these user biases would help in interpreting their ratings better, instead of relying only on the observed ratings which are often conflicting for different users and can confuse a person trying to make a purchase decision.

In our second work, we tackle the issue of subjectivity at a finer level of reviews, which are also influenced by individual aspect biases. As opposed to explicit ratings, people express their opinions in free text with varying vocabulary when writing reviews. We develop a probabilistic graphical model to capture opinions expressed in reviews as a combination of aspect, topic, and sentiments for each sentence in the review. We further

develop a similarity measure for opinions in order to find and rank supporting opinions for a review. This framework would help a user going through diverse reviews of a product to search for consensus around a particular opinion and thus verify its reliability.

The third contribution of this thesis deals with personalising user reported drug side effects. For the same drug, different people may experience different symptoms or side effects depending on their pre-existing medical conditions or other concurrent drug usage. This limits the generalization power of self-reported symptoms for being used as a crowd-sourced knowledge base of drug side effects. We develop a neural network architecture to model the possible symptoms a patient might experience and their severity score, given her existing medical conditions and a set of treatments. Knowing about the possible side effects would help people make an informed decision when choosing between alternative treatments. Additionally, this will reduce the anxiety people might feel before taking a drug while looking at the long list of side effects reported online, even though most of them may not apply to her.

The fourth contribution of the thesis is to detect unreliable information in a domain independent, generic social media platform like Twitter. We develop a neural network model to automatically mine opinions expressed in people's posts and detect whether a story being circulated is a false rumor or not. Due to the open nature of social media and its huge network connectivity, it is crucial to detect and stop the spread of such rumors early, before they get disruptive and mislead millions of people. We believe our proposed framework is a step towards achieving that efficiently.

7.2 Future Directions

There exist multiple directions in which our proposed models could be extended, some of which are outlined below.

The probabilistic graphical model proposed for modeling latent user biases behind ordinal aspect ratings (described in Chapter 3) makes a fundamental contribution regarding alleviating Gaussian-Categorical non-conjugacy with introduction of auxiliary variable

augmentation. This mathematical construction of the model is generic and presents new possibilities for modeling such data in a wide-range of domains. The Author-ATS model (described in Chapter 4) proposed for capturing opinions in reviews, could be extended to examine whether the choice of aspects discussed in a review, the verbosity of a review are dependent on author's preferences as well. We have specialized our MoMEx framework (discussed in Chapter 5) for the use of symptom prediction in this thesis, but we believe this model is general in nature and could be applicable to other scenarios involving users, items and multiple interaction targets. The proposed rumor detection model (in Chapter 6) currently does not use user demographic information (e.g. age, gender, location, profession etc.). It would be interesting to explore in the future whether those attributes can help us identify a person's stance towards an issue more accurately.

An important dimension not considered in any of our work but could be explored in the future is the temporal dimension. A user's aspect biases may change over time changing their opinion about an item or issue. A person's demography or medical conditions could also vary over time, resulting in changes in her drug reactions. Future work should include modeling temporal behavior of users to better understand her online feedbacks.

Bibliography

[1] J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American statistical Association*, 1993.

[2] C.-S. Atodiresei, A. Tănăselea, and A. Iftene. Identifying fake news and fake users on twitter. *Procedia Computer Science*, 126:451–461, 2018.

[3] N. Barbieri. Regularized gibbs sampling for user profiling with soft constraints. In *Advances in Social Networks Analysis and Mining (ASONAM)*, 2011.

[4] M. Barthell, E. Shearer, J. Gottfried, and A. Mitchell. The evolving role of news on twitter and facebook. `http://www.journalism.org/2015/07/14/the-evolving-role-of-news-on-twitter-and-facebook/`, 2015.

[5] S. E. Baumgartner and T. Hartmann. The role of health anxiety in online health information search. *Cyberpsychology, Behavior, and Social Networking*, 14(10):613–618, 2011.

[6] Bazaarvoice. Bazaarvoice and the center for generational kinetics release new study on how millennials shop, 2012. [Online; last accessed 20-Feb-2018].

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *Journal of Machine Learning Research*, 2003.

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

[9] C. Castillo, M. Mendoza, and B. Poblete. Information credibility on twitter. In *WWW*, 2011.

[10] J. Chen, J. Zhu, Z. Wang, X. Zheng, and B. Zhang. Scalable inference for logistic-normal topic models. In *Advances in Neural Information Processing Systems*, 2013.

[11] Y.-C. Chen, Z.-Y. Liu, and H.-Y. Kao. Ikm at semeval-2017 task 8: Convolutional neural networks for stance detection and rumor verification. In *Workshop on Semantic Evaluation (SemEval-2017)*, 2017.

[12] Z. Chen, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh. Exploiting domain knowledge in aspect extraction. In *Proc. of EMNLP*, 2013.

[13] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *Proceedings of NIPS Deep Learning and Representation Learning Workshop*, 2014.

[14] R. J. Cline and K. M. Haynes. Consumer health information seeking on the internet: the state of the art. *Health education research*, 16(6):671–692, 2001.

[15] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 2011.

[16] H.-J. Dai, M. Touray, J. Jonnagaddala, and S. Syed-Abdul. Feature engineering for recognizing adverse drug reactions from twitter posts. *Information*, 7(2):27, 2016.

[17] L. Derczynski and K. Bontcheva. Pheme: Veracity in digital social networks. In *UMAP Workshops*, 2014.

[18] L. Derczynski, K. Bontcheva, M. Liakata, R. Procter, G. W. S. Hoi, and A. Zubiaga. Semeval-2017 task 8: Rumoureval: Determining rumour veracity and support for rumours. *arXiv preprint arXiv:1704.05972*, 2017.

[19] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

[20] L. Du, W. Buntine, and H. Jin. Sequential latent dirichlet allocation: Discover underlying topic structures within a document. In *Proc. of IEEE ICDM*, 2010.

[21] L. Du, J. K. Pate, and M. Johnson. Topic segmentation with an ordering-based topic model. In *Proc. of AAAI*, 2015.

[22] D. Egger, F. Uzdilli, M. Cieliebak, and L. Derczynski. Adverse drug reaction detection using an adapted sentiment classifier. In *Proc. of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing*, 2016.

[23] O. Enayet and S. R. El-Beltagy. Niletmrg at semeval-2017 task 8: Determining rumour and veracity support for rumours on twitter. In *Workshop on Semantic Evaluation (SemEval-2017)*, 2017.

[24] I. I. for Healthcare Informatics. Engaging patients through social media. *Report*, 2014.

[25] S. Fox and M. Duggan. Health online 2013. *Washington, DC: Pew Internet & American Life Project*, 2013.

[26] A. Friggeri, L. A. Adamic, D. Eckles, and J. Cheng. Rumor cascades. In *ICWSM*, 2014.

[27] E. Garmash and C. Monz. Ensemble learning for multi-source neural machine translation. In *COLING*, 2016.

[28] Y. Gong and Q. Zhang. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*, 2016.

[29] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer. Fake news on twitter during the 2016 u.s. presidential election. *Science*, 363(6425):374–378, 2019.

[30] A. Gruber, Y. Weiss, and M. Rosen-Zvi. Hidden topic markov models. In *International Conference on Artificial Intelligence and Statistics*, 2007.

[31] S. Gupta, S. Pawar, N. Ramrakhiyani, G. K. Palshikar, and V. Varma. Semi-supervised recurrent neural network for adverse drug reaction mention extraction. *CoRR*, 2017.

[32] K. Halder, L. Poddar, and M.-Y. Kan. Cold start thread recommendation as extreme multi-label classification. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, 2018.

[33] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[34] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[35] T. Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 2003.

[36] E. H. Hovy. Automated discourse generation using discourse structure relations. *Artificial intelligence*, 63(1):341–385, 1993.

[37] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *ACL*, 2015.

[38] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[39] Y. Jo and A. H. Oh. Aspect and sentiment unification model for online review analysis. In *ACM International Conference on Web Search and Data Mining*, 2011.

[40] J. Jonnagaddala, T. R. Jue, and H. Dai. Binary classification of twitter posts for adverse drug reactions. In *Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing, Big Island, HI, USA*, pages 4–8, 2016.

[41] M. I. Jordan. An introduction to probabilistic graphical models, 2003.

[42] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[43] M. E. Khan, S. Mohamed, B. M. Marlin, and K. P. Murphy. A stick-breaking likelihood for categorical data analysis with latent gaussian models. In *AISTATS*, 2012.

[44] S. Kim, J. Zhang, Z. Chen, A. H. Oh, and S. Liu. A hierarchical aspect-sentiment model for online reviews. In *Proc. of AAAI*, 2013.

[45] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[46] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2014.

[47] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[48] E. Kochkina, M. Liakata, and I. Augenstein. Turing at semeval-2017 task 8: Sequential approach to rumour stance classification with branch-lstm. *arXiv preprint arXiv:1704.07221*, 2017.

[49] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[50] Y. Koren and J. Sill. Collaborative filtering on ordinal user feedback. In *IJCAI*, 2013.

[51] S. Kumar, J. Cheng, J. Leskovec, and V. Subrahmanian. An army of me: Sockpuppets in online discussion communities. In *Proceedings of the 26th International Conference on World Wide Web*, pages 857–866. International World Wide Web Conferences Steering Committee, 2017.

[52] S. Kwon, M. Cha, and K. Jung. Rumor detection over varying time windows. *PloS one*, 2017.

[53] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang. Prominent features of rumor propagation in online social media. In *ICDM*, 2013.

[54] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[55] K. Lee, A. Qadir, S. A. Hasan, V. Datla, A. Prakash, J. Liu, and O. Farri. Adverse drug event detection in tweets with semi-supervised convolutional neural networks. In *Proceedings of the 26th International Conference on World Wide Web*, pages 705–714. International World Wide Web Conferences Steering Committee, 2017.

[56] Y. Li, N. Du, C. Liu, Y. Xie, W. Fan, Q. Li, J. Gao, and H. Sun. Reliable medical diagnosis from crowdsourcing: Discover trustworthy answers from non-experts. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 253–261. ACM, 2017.

[57] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han. A survey on truth discovery. *ACM Sigkdd Explorations Newsletter*, 17(2):1–16, 2016.

[58] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han. On the discovery of evolving truth. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 675–684. ACM, 2015.

[59] C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *Proc. of ACM CIKM*, 2009.

[60] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. *ICLR*, 2017.

[61] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124. ACM, 2017.

[62] X. Liu, A. Nourbakhsh, Q. Li, R. Fang, and S. Shah. Real-time rumor debunking on twitter. In *CIKM*, 2015.

[63] Y. Lu, C. Zhai, and N. Sundaresan. Rated aspect summarization of short comments. In *Proc. of WWW*, 2009.

[64] M. Lukasik, P. Srijith, D. Vu, K. Bontcheva, A. Zubiaga, and T. Cohn. Hawkes processes for continuous time sequence classification: an application to rumour stance classification in twitter. In *ACL*, 2016.

[65] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*, 2016.

[66] J. Ma, W. Gao, and K.-F. Wong. Detect rumors in microblog posts using propagation structure via kernel learning. In *ACL*, 2017.

[67] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[68] B. M. Marlin. Modeling user rating profiles for collaborative filtering. In *Advances in neural information processing systems*, 2003.

[69] F. Meng, Z. Lu, M. Wang, H. Li, W. Jiang, and Q. Liu. Encoding source language with convolutional neural network for machine translation. In *ACL-IJCNLP*, 2015.

[70] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*, 2013.

[71] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum. Optimizing semantic coherence in topic models. In *Proc. of EMNLP*, 2011.

[72] A. Mnih and R. R. Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

[73] S. Moghaddam and M. Ester. Ilda: interdependent lda model for learning latent aspects and their ratings from online product reviews. In *Proc. of SIGIR*, 2011.

[74] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, and R. Ghosh. Spotting opinion spammers using behavioral footprints. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 632–640. ACM, 2013.

[75] A. Mukherjee and B. Liu. Aspect extraction through semi-supervised modeling. In *Proc. of ACL*, 2012.

[76] S. Mukherjee, G. Weikum, and C. Danescu-Niculescu-Mizil. People on drugs: credibility of user statements in health communities. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 65–74. ACM, 2014.

[77] S. Muthukumarana and T. B. Swartz. Bayesian analysis of ordinal survey data using the dirichlet process to account for respondent personality traits. *Communications in Statistics-Simulation and Computation*, 2014.

[78] A. T. Nguyen, A. Kharosekar, M. Lease, and B. Wallace. An interpretable joint graphical model for fact-checking from crowds. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[79] B. Ofoghi, S. Siddiqui, and K. Verspoor. Read-biomed-ss: Adverse drug reaction classification of microblogs using emotional and conceptual enrichment. In *Proc. of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing*, 2016.

[80] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 309–319. Association for Computational Linguistics, 2011.

[81] M. Park. *HealthTrust: Assessing the Trustworthiness of Healthcare Information on the Internet*. PhD thesis, University of Kansas, 2013.

[82] J. Pasternack and D. Roth. Latent credibility analysis. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1009–1020. ACM, 2013.

[83] M. Paul and R. Girju. A two-dimensional topic-aspect model for discovering multi-faceted topics. In *Proc. of AAAI*, 2010.

[84] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[85] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach. In *Uncertainty in artificial intelligence*, 2000.

[86] G. Peterson, P. Aslani, and K. A. Williams. How do consumers search for and appraise information on medicines on the internet? a qualitative study using focus groups. *JMIR*, 5(4), 2003.

[87] C. Pierre. Balancing prediction and recommendation accuracy: hierarchical latent factors for preference data. In *SIAM*, 2012.

[88] V. Plachouras, J. L. Leidner, and A. G. Garrow. Quantifying self-reported adverse drug events on twitter: signal and topic analysis. In *Proceedings of the 7th 2016 International Conference on Social Media & Society*, page 6. ACM, 2016.

[89] L. Poddar, W. Hsu, and M. L. Lee. Author-aware aspect topic sentiment model to retrieve supporting opinions from reviews. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 472–481, 2017.

[90] L. Poddar, W. Hsu, and M. L. Lee. Quantifying aspect bias in ordinal ratings using a bayesian approach. In *Proc. of IJCAI*, 2017.

[91] L. Poddar, W. Hsu, M. L. Lee, and S. Subramaniyam. Predicting stances in twitter conversations for detecting veracity of rumors: A neural approach. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 65–72. IEEE, 2018.

[92] L. Poddar, W. Hsu, M. L. Lee, and S. Subramaniyam. Predicting user reported symptoms using a gated neural network. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019.

[93] N. G. Polson, J. G. Scott, and J. Windle. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American statistical Association*, 2013.

[94] K. Popat, S. Mukherjee, J. Strötgen, and G. Weikum. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 1003–1012. International World Wide Web Conferences Steering Committee, 2017.

[95] H. Post. Hoaxes run amok on social media as hurricane florence moves inland, 2018. [Online; last accessed 14-July-2019].

[96] W. Post. https://www.washingtonpost.com/news/the-intersect/wp/2017/08/28/no-the-shark-picture-isnt-real-a-running-list-of-harveys-viral-hoaxes, 2017. [Online; last accessed 14-July-2019].

[97] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei. Rumor has it: Identifying misinformation in microblogs. In *EMNLP*, 2011.

[98] M. M. Rahman and H. Wang. Hidden topic sentiment model. In *Proc. of World Wide Web*, 2016.

[99] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proc. of EMNLP*, 2009.

[100] M. Rastegar-Mojarad, R. K. Elayavilli, Y. Yu, and H. Liu. Detecting signals in noisy data-can ensemble classifiers help identify adverse drug reaction in tweets. In *Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing*, 2016.

[101] S. Rendle. Factorization machines. In *ICDM*, 2010.

[102] S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 2012.

[103] P. Resnik and E. Hardisty. Gibbs sampling for the uninitiated. Technical report, Maryland Univ College Park Inst for Advanced Computer Studies, 2010.

[104] P. E. Rossi, Z. Gilula, and G. M. Allenby. Overcoming scale usage heterogeneity: A bayesian hierarchical approach. *Journal of the American Statistical Association*, 2001.

[105] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[106] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, 2008.

[107] H. Samuel and O. Zaiane. Medfact: Towards improving veracity of medical information in social media using applied machine learning. In *Advances in Artificial Intelligence: 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, May 8–11, 2018, Proceedings 31*, pages 108–120. Springer, 2018.

[108] G. P. Schoenherr and R. W. White. Interactions between health searchers and search engines. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 143–152. ACM, 2014.

[109] V. Singh, S. Narayan, M. S. Akhtar, A. Ekbal, and P. Bhattacharyya. Iitp at semeval-2017 task 8: A supervised approach for rumour evaluation. In *Workshop on Semantic Evaluation (SemEval-2017)*, 2017.

[110] K. Sjölander, K. Karplus, M. Brown, R. Hughey, A. Krogh, I. S. Mian, and D. Haussler. Dirichlet mixtures: a method for improved detection of weak but significant protein sequence homology. *Computer applications in the biosciences: CABIOS*, 1996.

[111] M. D. Smucker, D. Kulp, and J. Allan. Dirichlet mixtures for query estimation in information retrieval. *Center for Intelligent Information Retrieval*, 2005.

[112] A. Srivastava, G. Rehm, and J. M. Schneider. Dfki-dkt at semeval-2017 task 8: rumour detection and classification using cascading heuristics. In *Workshop on Semantic Evaluation (SemEval-2017)*, 2017.

[113] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of the 18th international conference on World wide web*, 2009.

[114] K. Sugiyama, M.-Y. Kan, K. Halder, et al. Treatment side effect prediction from online user-generated content. In *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*, pages 12–21, 2018.

[115] S. Sun, H. Liu, J. He, and X. Du. Detecting event rumors on sina weibo automatically. In *Asia-Pacific Web Conference*. Springer, 2013.

[116] M. Swan. Crowdsourced health research studies: an important emerging complement to clinical trials in the public health research ecosystem. *Journal of medical Internet research*, 14(2), 2012.

[117] N. Y. Times. After las vegas shooting, fake news regains its megaphone, 2017. [Online; last accessed 14-July-2019].

[118] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *Proc. of WWW*, 2008.

[119] I. Titov and R. T. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proc. of ACL*, 2008.

[120] A. Trabelsi and O. R. Zaiane. Mining contentious documents using an unsupervised topic model based approach. In *Proc. of IEEE ICDM*, 2014.

[121] S. Virtanen and M. Girolami. Ordinal mixed membership models. In *International Conference on Machine Learning*, 2015.

[122] F. Wang, M. Lan, and Y. Wu. Ecnu at semeval-2017 task 8: Rumour evaluation using effective features and supervised ensemble models. In *Workshop on Semantic Evaluation (SemEval-2017)*, 2017.

[123] H. Wang and M. Ester. A sentiment-aligned topic model for product aspect rating prediction. In *EMNLP*, 2014.

[124] H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis on review text data: a rating regression approach. In *Proc. of SIGKDD*, 2010.

[125] H. Wang, Y. Lu, and C. Zhai. Latent aspect rating analysis without aspect keyword supervision. In *Proceedings of the 17th ACM SIGKDD*, 2011.

[126] R. W. White, R. Harpaz, N. H. Shah, W. DuMouchel, and E. Horvitz. Toward enhanced pharmacovigilance using patient-generated data on the internet. *Clinical Pharmacology & Therapeutics*, 96(2):239–246, 2014.

[127] Wired. Facebook is changing news feed (again) to stop fake news, 2019. [Online; last accessed 04-July-2019].

[128] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 153–162. ACM, 2016.

[129] F. Yang, Y. Liu, X. Yu, and M. Yang. Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, 2012.

[130] Z. Yang, A. Kotov, A. Mohan, and S. Lu. Parametric and non-parametric user-aware sentiment topic models. In *Proc. of SIGIR*, 2015.

[131] S. W.-t. Yih, X. He, and C. Meek. Semantic parsing for single-relation question answering. In *ACL*, 2014.

[132] X. Yin, J. Han, and S. Y. Philip. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering*, 20(6):796–808, 2008.

[133] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.

[134] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan. A convolutional approach for misinformation identification. In *IJCAI*, 2017.

[135] W. Zhang and J. Wang. Prior-based dual additive latent dirichlet allocation for user-item connected documents. In *Proc. of IJCAI*, 2015.

[136] Z. Zhang, J. Nie, and X. Zhang. An ensemble method for binary classification of adverse drug reactions from social media. In *Proc. of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing*, 2016.

[137] T. Zhao, C. Li, Q. Ding, and L. Li. User-sentiment topic model: refining user's topics with sentiment information. In *ACM SIGKDD Workshop on Mining Data Semantics*, 2012.

[138] Z. Zhao, P. Resnick, and Q. Mei. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *WWW*, 2015.

[139] S. Zhi, Y. Sun, J. Liu, C. Zhang, and J. Han. Claimverif: a real-time claim verification system using the web and fact databases. In *Proceedings of the 2017 ACM on*

Conference on Information and Knowledge Management, pages 2555–2558. ACM, 2017.

[140] L. Zhining, G. Xiaozhuo, Z. Quan, and X. Taizhong. Combining statistics-based and cnn-based information for sentence classification. In *ICTAI*. IEEE, 2016.