DECEPTIVE REVIEW IDENTIFICATION VIA REVIEWER NETWORK REPRESENTATION LEARNING

by

Shih-Feng Yang

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Department of Computer and Information Technology

West Lafayette, Indiana

December 2021

I am dedicating this thesis to my beloved family.

ACKNOWLEDGMENTS

It has been a fascinating journey during my days at Purdue University. This is my first time studying abroad, living in a different country, and shoveling snow in winter. Things end but memories will last forever.

I would like to express my appreciation to my advisor, Dr. Julia Rayz, for your patience, guidance, and support. I will never forget your sincerity and encouragement over the years. Also, thank you to my committee members, Dr. Clifton Bingham, Dr. John Springer, and Dr. Ida Ngambeki. Your encouraging words and thoughtful, detailed feedback have been very important to me.

Thank you to all members of the AKRaNLU lab, especially Kanishka Misra and Xiaonan Jing. I enjoyed all the brainstormings that we had, and I have learned so much. Those precious moments, I will remember for a long time.

I sincerely appreciate Purdue CERIS (The Center for Environmental and Regulatory Information Systems), which supported me financially throughout my Master's and Ph.D. Thank you to all the team members, especially Michael Hill. You are always so supportive and kind to me when I worked as a graduate assistant.

Special thanks to Dr. Yi-Yu Lai for supporting and encouraging me to pursue my Ph.D. study. Also, thank you to all my Taiwanese friends in our West Lafayette Dining Group and Cumberland Ave Group. Without your company, I couldn't have gone through these difficult and unusual years and accomplished my long-term achievements.

Last but not least, thank you to my family for their selfish love, endless support (both emotionally and financially), and sacrifices for me to pursue my dream. Thank you, Mom and Dad, for being healthy and supportive. Thank you, my younger brother, for taking care of our parents in past years.

My most heartfelt thanks to each and every one of you. You complete me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

With the growth of the popularity of e-commerce and mobile apps during the past decade, people rely on online reviews more than ever before for purchasing products, booking hotels, and choosing all kinds of services. Users share their opinions by posting product reviews on merchant sites or online review websites (e.g., Yelp, Amazon, TripAdvisor). Although online reviews are valuable information for people who are interested in products and services, many reviews are manipulated by spammers to provide untruthful information for business competition. Since deceptive reviews can damage the reputation of brands and mislead customers' buying behaviors, the identification of fake reviews has become an important topic for online merchants. Among the computational approaches proposed for fake review identification, network-based fake review analysis jointly considers the information from review text, reviewer behaviors, and production information. Researchers have proposed network-based methods (e.g., metapath) on heterogeneous networks, which have shown promising results.

However, we've identified two research gaps in this study: 1) We argue the previous network-based reviewer representations are not sufficient to preserve the relationship of reviewers in networks. To be specific, previous studies only considered first-order proximity, which indicates the observable connection between reviewers, but not second-order proximity, which captures the neighborhood structures where two vertices overlap. Moreover, although previous network-based fake review studies (e.g., metapath) connect reviewers through feature nodes across heterogeneous networks, they ignored the multi-view nature of reviewers. A view is derived from a single type of proximity or relationship between the nodes, which can be characterized by a set of edges. In other words, the reviewers could form different networks with regard to different relationships. 2) The text embeddings of reviews in previous network-based fake review studies were not considered with reviewer embeddings.

To tackle the first gap, we generated reviewer embeddings via MVE (Qu et al., 2017), a framework for multi-view network representation learning, and conducted spammer classification experiments to examine the effectiveness of the learned embeddings for distinguishing spammers and non-spammers. In addition, we performed unsupervised hierarchical clustering to observe the

clusters of the reviewer embeddings. Our results show the clusters generated based on reviewer embeddings capture the difference between spammers and non-spammers better than those generated based on reviewers' features.

To fill the second gap, we proposed hybrid embeddings that combine review text embeddings with reviewer embeddings (i.e., the vector that represents a reviewer's characteristics, such as writing or behavioral patterns). We conducted fake review classification experiments to compare the performance between using hybrid embeddings (i.e., text+reviewer) as features and using text-only embeddings as features. Our results suggest that hybrid embedding is more effective than text-only embedding for fake review identification. Moreover, we compared the prediction performance of the hybrid embeddings with baselines and showed our approach outperformed others on fake review identification experiments.

The contributions of this study are four-fold: 1) We adopted a multi-view representation learning approach for reviewer embedding learning and analyze the efficacy of the embeddings used for spammer classification and fake review classification. 2) We proposed a hybrid embedding that considers the characteristics of both review text and the reviewer. Our results are promising and suggest hybrid embedding is very effective for fake review identification. 3) We proposed a heuristic network construction approach that builds a user network based on user features. 4) We evaluated how different spammer thresholds impact the performance of fake review classification. Several studies have used the same datasets as we used in this study, but most of them followed the spammer definition mentioned by Jindal and Liu (2008). We argued that the spammer definition should be configurable based on different datasets. Our findings showed that by carefully choosing the spammer thresholds for the target datasets, hybrid embeddings have higher efficacy for fake review classification.

CHAPTER 1. INTRODUCTION

With the growth of the popularity of e-commerce and mobile apps during the past decade, people rely on online reviews more than ever before for purchasing products, booking hotels, and choosing various services. Users share their opinions by posting product reviews on merchant sites or online review websites (e.g., Yelp, Amazon, Tripadvisor). A local consumer reviewer survey (Rosso & Cagnina, 2017) found that 68% of consumers have reinforced their decision to purchase a product or service by reading positive online reviews, and 92% of consumers read online reviews to judge a local business or a product. Although online reviews are valuable information for people who are interested in products and services, many reviews have been found to be manipulated by spammers to provide untruthful information for business competition. Since deceptive reviews can damage the reputation of brands and mislead customers' buying behaviors, the identification of fake reviews has become an important topic for online merchants. For example, Mayzlin, Dover, and Chevalier (2014) discuss how undetectable fake reviews influence consumers to make suboptimal choices when purchasing products. Also, the presence of biased reviews may lead consumers to mistrust reviews. In 2009, the Federal Trade Commission in the United States updated its guidelines governing endorsements and testimonials to include online reviews (FTC, 2009). Furthermore, in February 2012, the UK Advertising Standards Authority ruled that the travel review website TripAdvisor must stop claiming that it offers "honest, real, or trusted" reviews from "real travelers" since too many fake reviews were manipulated to mislead people. Since more attention has been drawn to the dangers of fake reviews, fake review identification has become more and more important. In the next sections, we discuss the development of fake review identification in the past years, which sets the context for the research questions and experiments addressed by this work.

1.1 Definitions

To reduce the ambiguity caused by the usages of terms, we define the following terms:

Fake reviews: Reviews written and manipulated for purposes other than providing the truthful opinions of reviewers. Since the terms “fake review”, “opinion spam”, and “deceptive review” are often used to refer to the same concept in previous literature, we use them interchangeably throughout this work

Network embeddings: Low-dimensional representations (usually vectors) of vertices in networks that capture and preserve the network structure (D. Wang, Cui, & Zhu, 2016). Such vector representations are able to preserve the proximities between nodes, which can be used as features for network analysis (Qu et al., 2017)

View: A view is derived from a single type of proximity or relationship between nodes, which can be characterized by a set of edges.

Reviewer embeddings: Network embeddings that capture and preserve the characteristics (e.g., writing or behavioral patterns of reviewers) of a reviewer network structure

Spammers: Reviewers who wrote at least one fake review (Jindal & Liu, 2008)

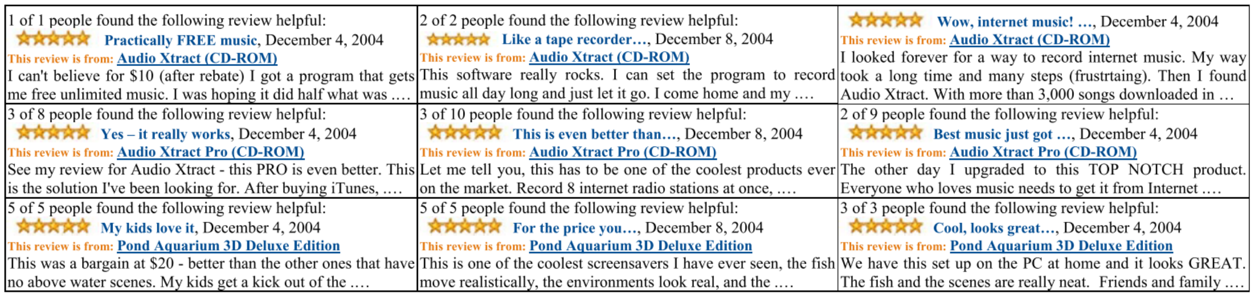
1.2 Background of Fake Review Identification

According to Jindal and Liu (2007), there are three common types of fake reviews: 1) Untruthful opinions, which deliberately mislead readers or opinion mining systems by giving undeserving positive reviews to promote objects or by giving malicious negative reviews to deteriorate the reputations of objects; 2) reviews on brands only, which comment on the brands and manufacturers rather than products; 3) non-reviews, which are advertisements or random text irrelevant to the products.

To identify type 2 and type 3 fake reviews, researchers usually recruit annotators to (manually) label spam and non-spam reviews. Since these two types are more recognizable (i.e., type 2 reviews are only related to brands but not the target product; type 3 reviews contain advertisements that are irrelevant to the target product), annotators can more easily reach a consensus on whether a review is type 2 or type 3 (Jindal & Liu, 2007). After collecting labels for spam and non-spam reviews, researchers explore useful features and design predictive models to identify whether a given review is fake or genuine. However, labeling type 1 fake reviews is relatively difficult. First, untruthful reviews can be deliberately crafted and are always not even composed by the user who posts the reviews. Spammers sometimes get incentives to imitate real reviewers' opinions. Second, reviewers can intentionally fabricate reviews to indicate features that are nonexistent or to give misleading opinions about the target products, brands, and services. Even if we carefully compare the details (e.g., product specification, service description, store location) mentioned in the review with the actual properties of the target product, it is still hard to tell if a review is deceptive or not. Third, according to the findings of Jindal, Liu, and Lim (2010) and Mukherjee et al. (2012), the behavioral patterns of a reviewer (or a group of reviewers) can play a more important role in fake review identification than the content of a review. Due to the above reasons, labeling type 1 reviews is more complicated, and annotators tend to agree less. Because this is an ongoing problem that requires a robust solution, we focus our investigation on this type of review.

It is also important to distinguish between opinion spam and other types of spam. Jindal and Liu (2007) distinguished opinion spam from web spam, email spam, and recommendation system attacks. Gyongyi and Garcia-Molina (2005) defined web spam as actions intended to mislead search engines into ranking some pages higher than they deserve. There are two types of web spam: link spam and content spam. Link spam is a type of spam where a web page contains advertisement URL links, which do not have a clear overlap with opinion spam. Content spam is a type of spam in which certain words, usually irrelevant to the content, are added to a web page. The nature of opinion spam is very different from content spam because a fake review is not usually formed by irrelevant spam words. Email spam refers to unsolicited commercial advertisements in email. Although commercial advertisements also exist in reviews, advertisements in reviews are not as frequent as those in email. Besides, the major challenge of email spam is to effectively identify advertisements, whereas the major challenge for the domain of opinion spam is detecting untruthful reviews. A recommendation system attack refers to actions whereby attackers manipulate data (e.g., ratings, feedback) to mislead a recommendation system. This kind of attack is often done by adding malicious user profiles to a recommendation system. This type of attack is different from opinion spam because researchers do not consider reviewer profiles for opinion spam detection. Instead, reviewer behavioral patterns and reviewer group indicators have been used in previous studies of opinion spam (Jindal et al., 2010; Mukherjee et al., 2012).

Figure 1.1 contains examples of suspicious reviews that show the characteristics of spam reviews and spammers. The reviews were written by three different reviewers; however, the following suspicious patterns can be noted: 1) The three users all reviewed the same three products and gave all of them five-star ratings. 2) The posted reviews are within a small time window of four days. 3) The reviewers only reviewed these three products. 4) The reviewers were among the earliest reviewers of the products after their release.



1 of 1 people found the following review helpful: ★★★★★ **Practically FREE music**, December 4, 2004 This review is from: Audio Xtract (CD-ROM) I can't believe for $10 (after rebate) I got a program that gets me free unlimited music. I was hoping it did half what was ….	2 of 2 people found the following review helpful: ★★★★★ **Like a tape recorder…**, December 8, 2004 This review is from: Audio Xtract (CD-ROM) This software really rocks. I can set the program to record music all day long and just let it go. I come home and my ….	★★★★★ **Wow, internet music! …**, December 4, 2004 This review is from: Audio Xtract (CD-ROM) I looked forever for a way to record internet music. My way took a long time and many steps (frustrtaing). Then I found Audio Xtract. With more than 3,000 songs downloaded in …
3 of 8 people found the following review helpful: ★★★★★ **Yes – it really works**, December 4, 2004 This review is from: Audio Xtract Pro (CD-ROM) See my review for Audio Xtract - this PRO is even better. This is the solution I've been looking for. After buying iTunes, ….	3 of 10 people found the following review helpful: ★★★★★ **This is even better than…**, December 8, 2004 This review is from: Audio Xtract Pro (CD-ROM) Let me tell you, this has to be one of the coolest products ever on the market. Record 8 internet radio stations at once, ….	2 of 9 people found the following review helpful: ★★★★★ **Best music just got …**, December 4, 2004 This review is from: Audio Xtract Pro (CD-ROM) The other day I upgraded to this TOP NOTCH product. Everyone who loves music needs to get it from Internet ….
5 of 5 people found the following review helpful: ★★★★★ **My kids love it**, December 4, 2004 This review is from: Pond Aquarium 3D Deluxe Edition This was a bargain at $20 - better than the other ones that have no above water scenes. My kids get a kick out of the ….	5 of 5 people found the following review helpful: ★★★★★ **For the price you…**, December 8, 2004 This review is from: Pond Aquarium 3D Deluxe Edition This is one of the coolest screensavers I have ever seen, the fish move realistically, the environments look real, and the ….	3 of 3 people found the following review helpful: ★★★★★ **Cool, looks great…**, December 4, 2004 This review is from: Pond Aquarium 3D Deluxe Edition We have this set up on the PC at home and it looks GREAT. The fish and the scenes are really neat. Friends and family ….

Figure 1.1. Examples of fake reviews on Amazon (Mukherjee et al., 2012)

The above examples showed that many features of reviews and reviewers can be used to determine whether a review is fake or genuine. In the next section, we introduce the features that were used in previous studies.

1.2.1 Features

In previous research, many linguistic features have been considered for analyzing deceptive reviews, including statistical features such as N-grams (Cagnina & Rosso, 2015; S. Feng, Banerjee, & Choi, 2012; S. Feng, Xing, Gogar, & Choi, 2012; Fusilier, Montes-y Gómez, Rosso, & Cabrera, 2015b; Ott, Cardie, & Hancock, 2013; Ott, Choi, Cardie, & Hancock, 2011), part-of-speech tags (Lai, Xu, Lau, Li, & Jing, 2010; Ott et al., 2013, 2011), word embeddings (Aghakhani, Machiry, Nilizadeh, Kruegel, & Vigna, 2018; Ren & Ji, 2017; W. Zhang, Du, Yoshida, & Wang, 2018), and psycho-linguistic features (Cagnina & Rosso, 2015; Ott et al., 2013, 2011). Although linguistic features are effective for text analysis such as authorship attribution (Juola, 2008; Koppel, Schler, & Argamon, 2009; Stamatatos, 2009) and sentiment analysis (Nasukawa & Yi, 2003; Taboada, 2016), it is difficult to only use linguistic properties to determine whether a review is deceptive or not (Mukherjee, Venkataraman, Liu, & Glance, 2013). Thus, researchers have incorporated various perspectives, such as textual features, reviewer behaviors, and spammer group indicators, to identify untruthful reviews. Reviewer behaviors reflect whether a reviewer displays a pattern with respect to promoting and/or demoting products from certain brands (Jindal et al., 2010). Spammer group indicators utilize common patterns between different reviewers to evaluate if there are spammer groups (Mukherjee, Liu, Wang, Glance, & Jindal, 2011). A more detailed review of the use of features will be described in Chapters 2 and 3.

Since the features are extracted from real-world data, we need to understand the datasets investigated in previous research. In the next section, we survey the datasets built and used for fake review identification.

1.2.2 Datasets

Many initial studies constructed datasets based on the meta-information of reviewers and reviews, unusual review patterns, and indicators of spam groups (Mukherjee et al., 2011). Jindal and Liu (2007) and Jindal and Liu (2008) collected more than 5 million product reviews from Amazon and other online stores. They designed heuristic rules of fake review annotation and recruited annotators to label a small portion of the collected data based on the provided rules. After the small datasets were labeled, they used them to infer the labels of a larger amount of data. However, the resulting datasets were not convincing enough and were not widely used in other studies. Later, after crowdsourcing resources such as Amazon Mechanical Turk (AMT) became more popular, researchers started to hire workers to build human-annotated datasets. Ott et al. (2011) collected real hotel reviews from multiple sources (e.g., TripAdvisor, Expedia, Hotels.com, Orbitz, Priceline, and Yelp) and marked them as truthful. They also collected deceptive reviews by recruiting AMT workers and asking them to write reviews based on provided instructions. J. Li, Ott, and Cardie (2013) built the FOUR-CITIES dataset, which consists of deceptive reviews of eight hotels in four cities (i.e., Chicago, New York, Los Angeles, and Houston) from AMT as well as a set of truthful reviews from TripAdvisor. While these datasets were 100% built by people, the reviews written by AMT workers were not written by actual customers. In addition, as the AMT workers followed rules provided by the researchers to fabricate the reviews, it is questionable whether the data is similar to real-world data.

A turning point in the development of fake review datasets came with the Yelp datasets, which were collected by Yelp.com and were first used by Mukherjee, Venkataraman, Liu, and Glance (2013). In these datasets, Yelp labeled "filtered" and "unfiltered" reviews by their filtering algorithm. Reviews labeled as "filtered" were treated as fake or suspicious by Yelp's filtering algorithm. However, its filtering algorithm is a trade secret. In this study, we experiment with Yelp's filtered and unfiltered reviews to find out what Yelp's filter might be doing. It is important to note that Yelp had a filtering algorithm to identify recommended and filtered reviews. The filtered reviews were made public. Although the Yelp anti-fraud filter was not perfect, according to the analysis done by Mukherjee, Venkataraman, Liu, and Glance (2013), fake review classification under a balanced class distribution (i.e., the same amount of fake reviews and genuine reviews) had an accuracy rating of 67.8%, which was significantly higher than random guessing (50%).

Several studies (Mukherjee, Venkataraman, Liu, & Glance, 2013; Rayana & Akoglu, 2015, 2016) have discussed Yelp's filtering algorithm using its filtered and recommended reviews. The researchers have since released three different datasets:

- The YelpChi dataset contains 67,395 reviews (13.23% of filtered reviews) from 38,063 reviewers (20.33% of spammers) across 201 hotels and restaurants in Chicago.
- The YelpNYC dataset contains 359,052 reviews (10.27% of filtered reviews) from 160,225 reviewers (17.79% of spammers) across 923 hotels and restaurants in New York City.
- The YelpZIP dataset contains 608,598 reviews (13.22% of filtered reviews) from 260,277 reviewers (23.91% of spammers) across 5,044 restaurants. The reviews were collected and organized by zip code in a continuous region of the U.S., including NJ, VT, CT, and PA.

Each review was labeled as filtered (fake) or recommended (non-fake) by Yelp.com and has product and user information, a timestamp, ratings, and review content. Compared to the datasets built before, the Yelp datasets contain genuine reviews written by real users. Moreover, the size of the datasets is much larger than previous ones, and the datasets are accessible to the public.

In order to automatically extract features and identify fake reviews from the datasets, computational approaches based on machine learning have been widely adopted over the past decade. We next discuss these computational methods used for fake review identification.

1.2.3 Computational Approaches

In the early stages, many studies made rule-based assumptions to distinguish untruthful and genuine reviews. For example, Wu, Greene, and Cunningham (2010) defined several predefined criteria and calculated the correlations between those criteria. The correlations were treated as “expected rules”. For each review, the authors computed the deviation between the expected rules and the review to indicate ”suspiciousness” or ”unexpectedness” (Jindal et al., 2010). Due to a lack of labeled fake reviews, the researchers considered the suspiciousness value as the label of review. The next step was to train logistic regression (LR) or support vector machine (SVM) models with the small dataset labeled by the rule-based suspiciousness value. The models were then used to predict fake reviews or spammers.

After the release of gold-standard annotated datasets, many researchers were able to adopt various computational methods on fake review detection and treated this as a classification problem. Among traditional supervised classification methods, the best performance reported in these models was about 91% in accuracy and in F1 (S. Feng, Banerjee, & Choi, 2012; S. Feng, Xing, et al., 2012). Utilizing deep learning for fake review identification has also recently attracted a lot of attention. These models can achieve better performance on classification, and they preserve the sparse and high dimensional features in low-dimensional vectors on latent space. The best performance among the neural network approaches reached around 95% in F1 W. Zhang et al. (2018).

Moreover, in order to tackle the problem that most real-world datasets do not contain labeled fake reviews, many Positive and Unlabeled learning (PU-learning) and collective classification approaches have been used. PU-learning enables models to learn from a smaller labeled training set but still achieves high performance. The key idea is to learn a classification model from a small portion of a dataset and then to use the model to infer labels for the rest of unlabeled data, known as "pseudo labels". The classification model then refits the whole dataset with real and pseudo labels, which is used for predicting unknown data. The best performance among the PU-learning approaches has achieved over 80% in F1 when using 25% of training data as known data and 75% as unknown data, and over 85% in F1 when using 50% of training data as known data and 50% as unknown data Cagnina and Rosso (2015).

Another interesting direction is network-based fake review identification. Researchers focus on performing classification on heterogeneous networks (H. Li et al., 2014). By inferencing the "metapath" (Yuan et al., 2018) between nodes in heterogeneous networks, the nodes in different types of networks can be correlated. Figure 1.2 is an example of a heterogeneous network, which contains edges that connect different types of nodes (e.g., review, reviewer, and IP address). Figure 1.3 is an example of a metapath that walks across different types of nodes (review, reviewer, and feature).

Network-based methods attempt to learn embeddings, usually vectors, to represent nodes in a network by approximating the likelihood between the original network and the node embeddings. That is, if two nodes are very similar or connected in a network, the distance of their node embeddings should be close. For instance, reviewers 1 and 2 share the same feature ACS in Figure 1.3, so the distance between their node embeddings is more likely to be small.

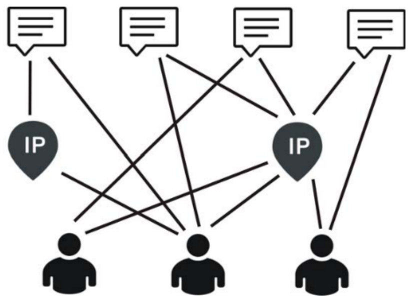


Figure 1.2. Heterogeneous network (H. Li et al., 2014)

In this study, we are especially interested in network-based approaches since these studies provide a more comprehensive perspective across different types of information networks. After our investigation, we identified two major research gaps in the literature. We discuss the gaps in the next section.

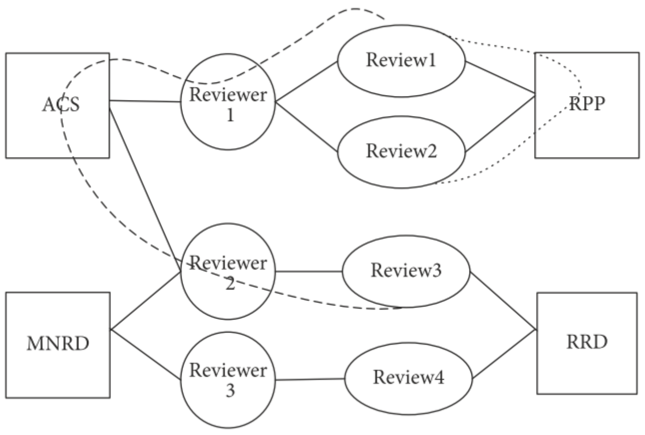


Figure 1.3. Sample network of users, reviews, and features (Yuan et al., 2018)

1.3 Research Gaps

The first research gap is that previous studies of fake review approaches cannot effectively capture and preserve the relationship between reviewers. The first reason is that, although previous network-based methods, such as metapath, can correlate two reviewers if they share the same features or have posted reviews about the same products, those methods failed to preserve the relationship of two reviewers if there was no direct edge connecting them. Based on our observation, the previous network-based methods capture only the first-order proximity of reviewers but cannot preserve the second-order proximity between reviewers (J. Tang et al., 2015). First-order proximity estimates the observed links in the local structure of the networks, while second-order proximity captures the extent to which the neighborhood structures of two vertices overlap. As illustrated in Figure 1.4, the first-order proximity between vertices 6 and 7 is high because they are connected to each other. On the other hand, it is low for vertices 5 and 6 because there is no edge between them. However, the second-order proximity of vertices 5 and 6 is high because it preserves the characteristic that the two vertices share a similar neighborhood even though they are not connected. J. Tang et al. (2015) showed their LINE model outperformed the graph factorization approach (Ahmed, Shervashidze, Narayanamurthy, Josifovski, & Smola, 2013) and DeepWalk (Perozzi, Al-Rfou, & Skiena, 2014) in experiments on node classification and link prediction by considering the second-order proximity of networks. The second reason for not effectively capturing and preserving the relationships between reviewers is that previous studies of fake review identification did not learn review/reviewer representations with multiple views. A view is derived from a single type of proximity or relationship between the nodes, which can be characterized by a set of edges. Considering multiple types of views for node representation learning is important because multiple views exist in many real-world networks. For instance, social media sites like Twitter consist of multiple types of proximities that can be induced by the following-followee, reply, retweet, and mention relationships. Each proximity defines a view of a network, and multiple proximities yield a network with multiple views. According to Qu et al. (2017), since each individual view is usually sparse and biased, the node representations learned from multi-view networks are more comprehensive and robust than those learned by single-view networks.

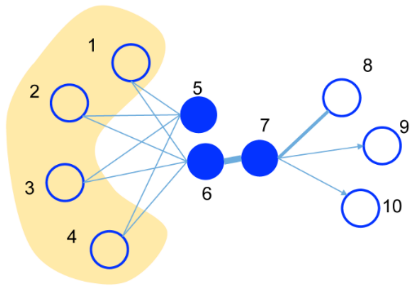


Figure 1.4. First-order proximity and second-order proximity (J. Tang et al., 2015)

The second research gap is that, to the best of our knowledge, the text embedding of a review was not jointly considered with its reviewer embedding in previous literature. Since different reviewers, especially spammers, have different behavioral patterns or writing styles (Jindal et al., 2010), it is necessary to consider the text embedding of a review along with its reviewer's profile.

In order to fill these gaps, we set up two main research questions and corresponding research methods. The details are illustrated in the next section.

1.4 Research Questions

To recap, we have identified the following two research gaps: 1) Previous graph-based fake review studies did not consider the characteristics of second-order proximity and multiple proximities of a reviewer network. 2) The text embedding of a review was not considered with its reviewer embedding in previous literature. We, therefore, proposed the following two research questions to address these gaps.

The first research question is: *Do reviewer embeddings learned via multi-view network representation approaches preserve the characteristics of spammers?* This research question is tied to the first research gap. As we've discussed earlier, if reviewer features are not created by considering second-order proximity and multiple proximities, they cannot effectively capture the relationship between reviewers. In order to verify this, we adopted a multi-view network representation, MVE (Qu et al., 2017), because of its strength to learn reviewer embeddings in consideration of second-order proximity and multiple proximities. To evaluate whether the reviewer embeddings are effective as the features of spammer classification, we conducted spammer classification experiments and used the reviewer embeddings as features. In addition, we applied a clustering algorithm to group reviewer embeddings into two clusters and compared the difference between the distribution of clusters and the distribution of spammer labels.

The second research question is: *Is the hybrid embedding of a review more sensitive at distinguishing whether a review is fake compared to the review's text embedding?* This question is tied to the second research gap. Since the text embedding of a review has not been jointly considered with its reviewer embedding for fake review classification, we compare the performance between having and not having reviewer embeddings. Moreover, we aim to explain how reviewer embeddings contribute to the performance difference. To perform the comparison, and thus fill this gap, we built hybrid embeddings that consist of each review's text embeddings and its reviewer's embedding. Next, we performed fake review classification experiments and evaluated whether using the hybrid embeddings as features was more effective than using the text embeddings as features. Finally, we conducted a feature ablation study to examine each of the features used to build reviewer embeddings.

The contributions of this study are four-fold: 1) We adopt a multi-view representation learning approach for reviewer embedding learning and analyze the efficacy of the embeddings used for spammer classification and fake review classification. 2) We propose a hybrid embedding that considers the characteristics of both the review text and the reviewer. Our results are promising and suggest hybrid embedding is very effective for fake review identification. 3)

We propose a heuristic network construction approach that builds a user network based on user features. 4) We evaluate how different spammer thresholds impact the performance of fake review classification. Several studies have used the same datasets as we used in this study, but most of them followed the spammer definition mentioned by Jindal and Liu (2008). We argue that the spammer definition should be configurable based on different datasets. Our findings show that by carefully choosing the spammer thresholds for the target datasets, hybrid embeddings have higher efficacy for fake review classification.

1.5 Assumptions

The following are the assumptions of the study:

- The fake/filtered review labels provided by Yelp in the datasets used in this study were treated as the ground truth of fakeness.
- The third-party libraries used in our experiments had no significant defects that may lead to incorrect experiment results.

1.6 Limitations

The following are the limitations of the study:

- The study only considers reviews written in English. Reviewers who had no English reviews were removed
- Studies based on crowdsourcing data may yield low-quality answers due to the openness of crowdsourcing (i.e., many crowdsourced reviews were not well-formed or not written under critical thinking) (Zheng, Li, Li, Shan, & Cheng, 2017). However, since this is not the focus of this study, we do not discuss whether the issue exists in the Yelp datasets.

1.7 Delimitations

The delimitations for this study include:

- Findings and observations in this study were based on the reviews and reviewers collected in the YelpChi, YelpNYC, YelpZIP datasets.

CHAPTER 2. REVIEW OF RELEVANT LITERATURE

2.1 Fake Review Identification

Data annotation for untruthful reviews is quite difficult. Before the release of large, annotated datasets, researchers proposed various unsupervised approaches for identifying effective features and abnormal patterns in untruthful reviews. However, after the release of several gold-standard datasets, researchers were able to conduct supervised learning and had a more reliable evaluation for this task.

To the best of our knowledge, Jindal and Liu (2007) and Jindal and Liu (2008) were the first studies to propose unsupervised approaches for spam reviews. The authors explicitly defined three types of review spam (i.e., untruthful reviews, reviews on brands only, and non-reviews) and developed a product review dataset from Amazon. They built three types of features from the dataset: 1) review-centric features, 2) reviewer-centric features, and 3) product-centric features. To detect type 2 and type 3 spam reviews, they manually labeled 470 spam reviews and trained a logistic regression (LR) model for fake review detection. Next, the authors conducted the following steps to detect type 1 spam reviews. First, they analyzed the dataset and identified duplicated reviews by content comparison (i.e., by computing whether the Jaccard distance of the bigram union of two reviews was over 90%) (B. Liu, 2007). They assumed that duplicated reviews were spam reviews while others were non-spam reviews. Moreover, they also assumed "outlier reviews" (i.e., reviews whose ratings largely deviated from the average rating but were not duplicated) were spam reviews. Second, the authors trained an LR model to learn from the duplicated reviews and found their model was predictive of both duplicated reviews and outlier reviews. They, therefore, suggested that the model was predictive of type 1 spam reviews if their assumption (i.e., duplicated reviews and outlier reviews are very likely to be type 1 spam reviews) held. In terms of evaluation metrics, the Area under ROC curve (AUC) was employed to evaluate the classification results. The best classification result in AUC values reported for type 2 and 3 spam reviews were 98.7%, while for duplicated reviews, it was 78%. For the detection of outlier reviews, the authors reported visualized lift curves, and they split outlier reviews into five categories: 1) Negative deviation, 2) positive deviation, 3) negative deviation on the same brand, 4) positive deviation on the same brand, and 5) bad product with average reviews. Based on the lift curves, they suggested that their LR model was predictive of four types of outlier reviews, with the exception being the positive deviation type.

In a subsequent paper (Jindal et al., 2010), the authors identified unusual reviewer patterns by finding unexpected rules that indicate spam activities. They used the same Amazon dataset as Jindal and Liu (2007) and Jindal and Liu (2008) and defined a series of unexpectedness factors based on class association rules (Ma & Liu, 1998). They also evaluated the confidence unexpectedness of their association rules for the dataset. The authors did not conduct experiments on spammer identification but provided a case study to show how their unexpected rules could identify suspicious reviewers. Their rules suggested that a reviewer could be a spammer if the person wrote negative reviews about all the products of a brand but wrote positive reviews about a competing brand.

Many studies followed the study of Jindal and Liu (2008) and proposed unsupervised approaches. Lai et al. (2010) proposed a hybrid approach to detect untruthful reviews by an unsupervised probabilistic language model that estimates the similarity between any pairs of reviews and to detect non-review by a supervised SVM model. Their unsupervised probabilistic language model was inspired by web spam detection approaches (Martinez-Romo & Araujo, 2009; Mishne, Carmel, Lempel, et al., 2005), which were based on Kullback-Leibler (KL) divergence. KL divergence is a measure commonly used to estimate the distance between two probability distributions. For each pair of reviews, the authors computed the KL divergence between two reviews and treated them as spam if they are similar. For the supervised model, the authors adopted Joachims (1998)'s SVM package and extracted syntactical, lexical, and stylistic features to detect non-review. For collecting part-of-speech (POS) tags, they developed a POS tagger based on WordNet lexicon (Miller, Beckwith, Fellbaum, Gross, & Miller, 1990) and WordNet API. In their evaluation, they followed Jindal and Liu (2007) and constructed the Amazon dataset. The authors adopted an evaluation approach, TREC Spam Track (Macdonald, Ounis, & Soboroff, 2007), used in email spam detection. Their system automatically scanned through a collection of consumer reviews to identify the spam. The classification results were then compared with the gold standard (i.e., the labels generated by Jindal and Liu (2007)'s approach). The authors reported four effectiveness measures of TREC Spam Track based on the confusion matrix as well as a measure called ”Area Above the ROC Curve” (1-AUC). They compared their unsupervised model with a LR-based baseline (Jindal & Liu, 2008) and a vector space model baseline (Dillon, 1983). According to the performance reported in the paper, their unsupervised approach achieved the best (lowest) 0.1416% in (1-AUC)% compared to baselines. In addition, the authors compared their supervised SVM model with a LR-based baseline and also achieved a better 1.8059% in (1-AUC)%.

Lin et al. (2014) proposed six time-sensitive features to highlight fake reviews in review sequences based on review contents and reviewer behaviors. They devised supervised solutions and an unsupervised threshold-based solution for spotting fake reviews. The six time-sensitive features include: 1) personal content similarity, 2) similarity with reviews on a product, 3) similarity with reviews on other products, 4) the review frequency of a reviewer, 5) the review frequency of a product, and 6) the repeatability measure. The former three features modeled the review contents while the latter three modeled the reviewer behaviors. The authors applied two supervised models, LR and support vector machine (SVM), to predict the labels of reviews. However, the authors noted that fake reviews usually occupy a small proportion of all reviews and are difficult to collect for training. Thus, they devised a threshold-based approach without labeled samples. The idea is to calculate the summation score of weighted feature scores for each review and compare it with a predetermined threshold to determine if a review is spam or non-spam. The authors reconstructed the Amazon dataset and followed the process used by Jindal and Liu (2007) to treat duplicated/near-duplicated reviews as fake reviews. The evaluation metrics were precision, recall, and F-measure (F1). For supervised approaches, the best result of SVM was 92.3% in F1 while for LR, it was 85.9% in F1. Their best performance for their unsupervised approach was 85.1% in F1.

Some studies worked on defining or exploring useful features for spam detection. Wu et al. (2010) proposed suspiciousness criteria to be a feature set comprised of several different dimensions of features: 1) Proportion of positive singletons, 2) concentration of positive singletons, 3) reactive positive singletons, 4) review weighted rating, 5) contribution weighted rating, 6) truncated rating, 7) sentiment shift, and 8) positive review length difference. The authors applied the Single Value Decomposition (SVD) and Unsupervised Hedge (UH) algorithms (Tan & Jin, 2004) for projecting the high dimensional features into a suspiciousness score. They collected a review dataset of 741 hotels on TripAdvisor and recruited 55 users to label the reviews of 46 hotels. They conducted a case study to compare the suspiciousness ranks of hotels labeled by human with the ranks calculated by their SVD method and the hedge algorithm method. Their analysis suggested that the proportion of positive singletons, reactive positive singletons, and truncated rating were the strongest features compared to others. In addition, SVD outperformed UH in their feature aggregation.

Mukherjee et al. (2011) focused on spam group identification and proposed eight indicators: 1) Time window, 2) group deviation, 3) group content similarity, 4) member content similarity, 5) early time frame, 6) ratio of group size, 7) group size, and 8) support count. The authors used the dataset built by Jindal and Liu (2008). The first step was to apply a frequency pattern mining model to find candidate groups. Second, they ranked the candidate groups with the proposed indicators and applied SVM rank (Joachims, 2002) to produce the final ranking of the candidate spammer groups. In the experiment, they showed 300 out of 2,273 groups ranked by the proposed method to three human judges. The 300 groups consisted of three types: 100 top-ranked, 100 middle-ranked, and 100 bottom-ranked. The human judges labeled each of the 300 groups as spam or non-spam. The evaluation metrics of the inter-rater agreement were based on Cohen's Kappa (Landis & Koch, 1977). The reported Cohen's Kappa scores for the three types of groups were all above 0.8, which indicates the proposed indicators and human judges had a high level of agreement regarding spam group identification.

Xie, Wang, Lin, and Yu (2012) proposed a singleton review (SR) spam detection model to identify unusual temporal patterns for singleton reviews (i.e., the only review written by a reviewer). The authors observed that a normal reviewers' arrival pattern is stable and uncorrelated to their rating pattern temporarily, while spam attacks are usually bursty and are either positively or negatively correlated to their rating pattern. Based on correlated abnormal patterns, the authors indicated there is a high likelihood of SR spam attacks. To apply the analysis in multidimensional time series, they first detected bursts in single time series. Second, they identified correlated abnormal patterns in multidimensional time series (e.g., time series of ratings, number of reviews, and ratio of SR). Finally, they pinpointed suspicious periods with higher time resolution (i.e., the window size of fluctuations on a time series). In the experiment, the authors used a snapshot of a review website (www.resellerratings.com) on October 6th, 2010. They recruited three human judges to label the suspiciousness of 53 resellers as well as 147 SRs. They conducted three case studies regarding the multidimensional time series of several stores, spam attacks caused by similar product competition between resellers, and reviews with direct evidence that the stores hired reviewers to provide positive singleton reviews.

As the gold-standard datasets were released by Ott et al. (2011) and Ott et al. (2013), many supervised learning methods were proposed for fake review identification. The dataset consists of truthful and deceptive reviews of 20 Chicago hotels: 400 truthful positive reviews (promoting the products) from TripAdvisor, 400 deceptive positive reviews from Amazon Mechanical Turk (AMT), 400 truthful negative reviews (damaging the reputation of products) from Expedia, Hotels.com, Orbitz, Priceline, TripAdvisor, and Yelp, and 400 deceptive negative reviews from AMT.

Ott et al. (2011) considered three automated approaches for detecting positive reviews: 1) Genre identification. Since the frequency distribution of POS tags is dependent on the genre of the text (Douglas, 2010; Rayson, Wilson, & Leech, 2002), they extracted POS features and evaluated if they were predictive at identifying truthful or deceptive reviews. 2) Psycho-linguistic deception detection. The authors used the Linguistic Inquiry and Word Count (LIWC) software (Pennebaker, Chung, Ireland, Gonzales, & Booth, 2007) to extract occurrence features about psycho-linguistic keywords for deception detection. 3) Text categorization. Word-level unigrams, bigrams, and trigrams were used to detect deception. They trained Naive Bayes and SVM classifiers with the above features for supervised deception detection.

The results showed that while standard n-gram based text categorization was the best individual detection approach (Bigram, 88.7% in F1 for truthful class, 89.0% in F1 for deceptive class), a combination approach using psycho-linguistic features and n-gram features performed slightly better (LIWC with bigram, 89.8% in F1 for truthful class, 89.8% in F1 for deceptive class). In a subsequent study (Ott et al., 2013), they applied the same approaches but focused on negative deceptive review detection. The best model reported reached approximately 86% in accuracy.

S. Feng, Banerjee, and Choi (2012) and S. Feng, Xing, et al. (2012) extended Ott et al. (2011) n-gram features and introduced deep syntax features (i.e., syntactic production rules derived from probabilistic context-free grammar parse trees.) Their experimental results achieved 91.2% accuracy on the Tripadvisor datasets used by Ott et al. (2011) and showed that deep syntactic patterns are helpful in discriminating deceptive writing. In a subsequent study (V. W. Feng & Hirst, 2013), the authors further extended S. Feng, Banerjee, and Choi (2012) method and incorporated profile compatibility features. They considered a product's profile (e.g., description about location, price, and service) and measured the extent to which a profile is compatible with the aspects mentioned in reviews. The results achieved scores of 91.3% in accuracy and 91.4% in F1. Although the above studies (S. Feng, Banerjee, & Choi, 2012; S. Feng, Xing, et al., 2012; V. W. Feng & Hirst, 2013; Ott et al., 2013, 2011) demonstrate that fake review detection can achieve scores of over 90% in accuracy and F1, those approaches are still not applicable in reality because it is difficult to collect such large quantities of labeled truthful and deceptive reviews for training. For example, the gold-standard dataset contains 50% labeled truthful reviews and 50% deceptive reviews, and S. Feng, Banerjee, and Choi (2012), S. Feng, Xing, et al. (2012), V. W. Feng and Hirst (2013) utilized 80% of reviews as a training set and 20% as a testing set. However, for real-world datasets, we are only able to collect a small portion of labeled data. The classifiers can only rely on a small training set but need to achieve a decent predictive performance for a large testing set.

In order to be able to learn from a small portion of positive reviews and unlabeled samples, H. Li et al. (2014) proposed a collective classification algorithm called "Multi-typed Heterogeneous Collective Classification" and then extended it to Collective Positive and Unlabeled learning (PU-learning) by leveraging the dependencies among reviews, users, and IP addresses. The authors treated the problem as one of collective classification in heterogeneous networks (as illustrated in Figure 1.1). They developed a dataset from Dianping, a Chinese review hosting site, which consists of 3,523 filtered (fake) reviews and 6,242 unfiltered (unlabeled) reviews from 500 restaurants in Shanghai, China. In the training phase, they considered the unlabeled reviews as unlabeled, but in the testing phase, they considered the unlabeled reviews as negative samples. The comparison baselines were 1) a logistic regression model; 2) PU-LEA (Hernández, Guzmán, y Gomez, & Rosso, 2013), another PU-learning algorithm that iteratively removes positive data instances from unlabeled ones, but does not consider network relations; 3) Spy-EM and Spy-SVM (B. Liu, Dai, Li, Lee, & Philip, 2003), a series of classifiers for learning from positive and unlabeled data; 4) Heterogeneous Collective Classification (Kong, Yu, Ding, & Wild, 2012), a heterogeneous classifier based on meta-path (Sun, Han, Yan, Yu, & Wu, 2011). In the experiments, the proposed model had the best performance and achieved around 59% in F1 when using 10% of the training data, about 70% in F1 when using 40% of the training data, and eventually about 75% in F1 when using 90% of the training data.

Ren, Ji, and Zhang (2014) proposed a semi-supervised PU-learning approach on the datasets collected by Ott et al. (2013). First, the authors identified reliable negative examples from the unlabeled dataset. Second, they generated both positive and negative examples with Latent Dirichlet Allocation (LDA) (Blei, Ng, & Jordan, 2003). Third, for the remaining unlabeled examples, two similarity weights were calculated regarding the probability of an unlabeled example belonging to the positive class and the negative class. They trained an SVM classifier for incorporating the unlabeled examples and their similarity weights. Their best results reached 86% in accuracy when using 40% of the training data.

Fusilier, Montes-y Gómez, Rosso, and Cabrera (2015a) proposed a PU-learning approach for the hotel review datasets gathered by Ott et al. (2013). The authors used only a hundred examples of deceptive opinions for training and achieved 80% in F1 for positive reviews and 70% in F1 for negative reviews. Moreover, the authors analyzed the role of opinion polarity in the detection of deception. Their results showed that negative spam reviews were more difficult to detect than positive ones. The authors also showed that using a single classifier for analyzing both kinds of opinions was better than using two separate classifiers. This implied that there were common characteristics in the way people write positive and negative fake reviews. In a subsequent study (Fusilier et al., 2015b), the authors considered detection as a stylish classification. They used character n-grams as features to capture the characteristics of lexical content as well as stylistic information. In their experiments, they compared character n-grams with word n-grams. They analyzed the features with Naive Bayes and SVM classifiers. The best results of character n-grams (5-grams) achieved 90% in F1 on the positive class and 88% on the negative class, which outperformed the best setting of word n-grams (i.e., unigram and bigram) on both positive class (88.2% in F1) and negative class (85.4% in F1). Furthermore, character n-grams also reached 80% in F1 when using 25% of the training data and 85% in F1 when using 50% of the training data.

Cagnina and Rosso (2015) extended the work of Fusilier et al. (2015b) and jointly utilized character n-grams, emotion-based features (Burgoon, Blair, Qin, & Nunamaker, 2003; Hancock, Curry, Goorha, & Woodworth, 2007; Newman, Pennebaker, Berry, & Richards, 2003), and LIWC-based features (Pennebaker et al., 2007). When applying character n-grams, the authors considered character n-grams in tokens rather than traditional n-grams (i.e., they only considered n-grams with n non-space characters) to reduce the dimensions of the n-gram features. The authors applied the features and trained SVMs. They used the datasets collected by Ott et al. (2013, 2011). In the experiments, the best results for both positive reviews and negatives were using character 4-grams with LIWC (89% in F1 for the positive class, 86% in F1 for the negative class). They also compared their work with Fusilier et al. (2015b) and showed their approach achieved the same level of performance but used only 3% of the feature dimensions for the positive class and 5% of the feature dimension for the negative class.

Recently, several studies (Aghakhani et al., 2018; Dong et al., 2018; H.-H. Huang, Wen, & Chen, 2017; Luo et al., 2017; Ren & Ji, 2017; C.-C. Wang, Day, Chen, & Liou, 2018; X. Wang, Liu, & Zhao, 2017a; W. Zhang et al., 2018) have applied various neural networks for supervised fake review detection because of the popularity and high performance of deep learning. Many neural networks, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) (LeCun, Bengio, & Hinton, 2015), Long Short-Term Memory (LSTM), Graph RNN (GRNN), Autoencoder decision forests, Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), and attention-based CNNs, were adopted in the detection approaches. A large portion of these studies focused on either adopting state-of-the-art neural networks or designing deep neural networks. In order to keep our focus on fake review identification rather than the variety of neural networks, we do not introduce the detailed designs of the neural networks in this paper.

Ren and Ji (2017) introduced a bi-directional average GRNN model for fake review detection. The model learned word embeddings as features using the CBOW model (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013) from an Amazon review corpus. The authors compared the proposed network with other state-of-the-art models (i.e., CNN, RNN, GRNN, average GRNN) on the dataset developed by Ott et al. (2013, 2011) and H. Li et al. (2014) and showed their model achieved the best performance, 83.6% in accuracy and 83.4% in F1.

H.-H. Huang et al. (2017) compared the performances of SVM, CNN, and Dependency-based CNN (DCNN) on an advertisement dataset crawled by the authors. They used the Chinese Word Segmenter and the parser of Stanford CoreNLP to build their features. The training data of the positive class was from the department of health of Taipei City government, which published cases of illegal advertisements monthly in the government's open data collection. The training data of the negative class consisted of sentences in legal advertisements. In the experiments, DCNN consistently achieved the best performance in three product domains (86.77% in F1 for cosmetics; 93.18% in F1 for food; 87.46% in F1 for drugs).

Luo et al. (2017) proposed a multi-aspect based neural network model to identify fake reviews. Three aspects of review features were considered: metadata, similarity, and sentiment. The metadata features were built based on human knowledge and expert experience; the similarity-based features were developed by a combination of TF-IDF and SimHash (Sadowski & Levin, 2007) methods to convert each review into a 64-bit feature word; the review sentiment features were built in four aspects: positive rate, negative rate, average positivity, and average negativity (Ott et al., 2011). Their dataset consisted of 3,000 comments and was manually labeled through 4,672 questionnaires (4,391 reclaimed). Each comment received about 30 annotated results. They adopted a feedforward neural network (FNN) to classify fake reviews. Their best result reached 83.1% in precision and 87.2% in recall.

Aghakhani et al. (2018) proposed FakeGAN, a semi-supervised GAN model for detecting deceptive reviews. Unlike the typical applications of GANs, where the ultimate goal is to have a strong generator, FakeGAN leverages GAN to create a well-trained discriminator. The authors utilized word2vec vector embeddings (Mikolov, Sutskever, et al., 2013) as input features. They applied FakeGAN to the dataset collected by Ott et al. (2011). The final model achieved 89.1% in accuracy.

Dong et al. (2018) presented an end-to-end trainable unified model to leverage the appealing properties from Autoencoder and random forest. A stochastic decision tree model was implemented to guide the global parameter learning process. The authors utilized several reviewer behavioral features and review features. They used the Amazon dataset collected by He and McAuley (2016) as well as ground-truth spamming reviews (Mukherjee et al., 2011). Their best model achieved 95.11% in F1.

W. Zhang et al. (2018) proposed a Deceptive Review Identification by Recurrent Convolutional Neural Network (DRI-RCNN) model to identify deceptive reviews by using word contexts and deep learning. The model learned two sets of word embeddings using the Skip-gram model (Mikolov, Sutskever, et al., 2013) on deceptive reviews and truthful reviews. They applied their model to the spam dataset collected by Ott et al. (2011) and the deception dataset collected by Ott et al. (2013, 2011) and H. Li et al. (2014). Compared to the baselines, DRI-RCNN consistently achieved better performances in F1 when using above 70% of the training data for the two datasets (spam dataset: 84.58% in F1 when using 70% of the training data; deception dataset: 82.42% in F1 when using 70% of the training data).

X. Wang, Liu, and Zhao (2017b) proposed a neural network model to detect review spam for the cold-start problem by learning to represent the reviews written by new reviewers with jointly embedded textual and behavioral information. The model learned embeddings for reviews, review ratings, and average product ratings. They followed the idea of TransE (Bordes, Usunier, Garcia-Duran, Weston, & Yakhnenko, 2013), which encodes the graph structure and represents the nodes and edges (head, relation, tail) in low dimension vector space. They treated products as the head part of TransE, reviewers as the relation part, and reviews as the tail part. The authors then used the learned embeddings as features and trained SVM classifiers. They applied their model to the Yelp dataset collected by Mukherjee, Venkataraman, Liu, and Glance (2013) and Rayana and Akoglu (2015) and showed the combination of their learned embeddings was more predictive of fake review detection than traditional linguistic features and behavioral features discussed by Mukherjee, Venkataraman, Liu, and Glance (2013). Their best model achieved 67.2% in F1 for hotels and 67.5% in F1 for restaurants.

X. Wang, Liu, and Zhao (2017a) presented an attention-based neural network to detect deceptive review spam by using linguistic and behavioral features. The authors considered several effective behavioral features proposed by Mukherjee, Venkataraman, Liu, and Glance (2013) and Rayana and Akoglu (2015) and X. Wang, Liu, He, and Zhao (2016) as the inputs of the Multi-Layer Perceptron (MLP) hidden layer and took the output vectors as behavioral feature vectors. In addition, they trained a bigram word embedding model as the inputs of a CNN and took the output vectors as linguistic feature vectors. They applied their model to the Yelp dataset collected by Mukherjee, Venkataraman, Liu, and Glance (2013) and Rayana and Akoglu (2015).

Their results consistently outperformed the baselines. For hotel reviews, their best model with the attention mechanism reached the best performance of 87.3% in F1 when incorporating the behavioral features used by Mukherjee et al. (2011) and 88.9% in F1 when incorporating the behavioral features used by X. Wang et al. (2016). For restaurant reviews, their best model with the attention mechanism reached the best performance of 87.9% in F1 when incorporating the behavioral features used by Mukherjee et al. (2011) and 91.2% in F1 when incorporating the behavioral features used by X. Wang et al. (2016).

2.2 Network Representation Learning

In the early stages of network representation learning, there were several unsupervised learning approaches that attempted to capture the “manifolds”, which are also called ”embeddings”. Manifold learning typically learns a low-dimensional vector for a node in a network. Most of the initial studies were based on nearest-neighbor graphs (Goodfellow, Bengio, & Courville, 2016). Belkin and Niyogi (2002, 2003) computed low-dimensional representations that preserved local neighborhood information for network nodes based on the graph Laplacian. Their approach used the properties of the Laplace Beltrami operator to construct invariant embedding maps. The algorithm constructs adjacency graphs for nodes, weights the edges, and computes the eigenvalues and eigenvectors as low-dimensional representations of nodes. The researchers showed node embeddings can reduce the influence of outliers and form clear clusters in different types of datasets through several case studies and data visualizations.

Perozzi et al. (2014) proposed DeepWalk, which utilizes local information from truncated random walks to learn node embeddings. Their approach first performed random walk to sample random paths for nodes. Second, the authors treated the random paths as textual sentences and applied the SkipGram algorithm (Mikolov, Chen, Corrado, & Dean, 2013), a language model that maximizes the co-occurrence probability among words that appear within a window in a sentence, to update the representations for nodes. In order to improve the efficiency of training time, they apply Hierarchical Softmax (Mnih & Hinton, 2009; Morin & Bengio, 2005) to reduce the computation of posterior distribution. They compared Deep Walk with five baseline methods: SpectralClustering (L. Tang & Liu, 2011), EdgeCluster (L. Tang & Liu, 2009a), Modularity (L. Tang & Liu, 2009b), wvRN (Macskassy & Provost, 2003), and Majority (i.e., choose most frequent labels). They performed a multi-label classification task on the BlogCatalog, Flickr, and YouTube datasets. The experiments showed DeepWalk performed better than four (out of five) baseline methods in F1 scores (42% for BlogCatalog, 38.7% for Flickr, 43.05% for YouTube). Also, when less than 60% of labeled nodes were provided, DeepWalk outperformed all baseline methods.

J. Tang et al. (2015) developed the LINE model, which preserves first-order proximity and second-order proximity separately when learning network embeddings. First-order proximity refers to the local pairwise proximity between the vertices in the network, while second-order proximity captures the "context" of a node (i.e., the neighborhood nodes of a vertex) and preserves the context proximity between different vertices. LINE introduces an edge sampling approach to solve the gradient divergence problem when edge weights have a high variance. The researchers treated weighted edges as multiple binary edges (e.g., converting an edge whose weight is five to five binary edges) and randomly sampled edges with the sampling probabilities proportional to the original edge weights. They compared LINE with a graph factorization approach (Ahmed et al., 2013) and DeepWalk (Perozzi et al., 2014) and performed experiments on language networks (Wikipedia), social networks (Flickr, YouTube), and citation networks (DBLP). For language network datasets, LINE outperformed others in the word analogy experiment in F1 scores (66.1%) and in the document classification experiment (83.74%). For social network datasets, LINE also outperformed others in the multi-label classification task for the Flickr dataset (64.74%) and for the YouTube dataset (46.43%). For citation network datasets, LINE outperformed others in the multi-label classification task for the DBLP AuthorCitation network dataset (66.05%) and for the DBLP PaperCitation network dataset (62.80%).

Node2Vec (Grover & Leskovec, 2016) maximizes the likelihood of preserving network neighborhoods of nodes when mapping nodes to a low-dimensional space of features. The researchers designed a flexible notion for a node's network neighborhood to facilitate Node2Vec in learning richer node representations. Compared to DeepWalk (Perozzi et al., 2014) and LINE (J. Tang et al., 2015), which were also inspired by the Skip-gram model (Mikolov, Chen, et al., 2013) and analogized a network to a document, Node2Vec introduced a second-order random walk for path sampling rather than the traditional random walk used by DeepWalk and LINE. The second-order random walk used a return parameter p and an in-out parameter q to control the likelihood of whether the path traversal stays near the starting node or goes out further to reach far nodes. In the experiments, Node2Vec was compared with three baselines: Spectral Clustering (L. Tang & Liu, 2011), DeepWalk (Perozzi et al., 2014), and LINE (J. Tang et al., 2015) on a multi-label classification task and a link prediction task. For the multilabel classification task, Node2Vec outperformed baselines when tested on the BlogCatalog dataset (25.81% in F1 score), the PPI dataset (17.91%), and the Wikipedia dataset (15.51%) with 50% of the nodes labeled for training. For the link prediction task, the researchers evaluated four different binary operators (i.e., average, Hadamard, weighted-L1, weighted-L2) to form an edge vector with two node embeddings. The result shows that the edge vectors built using the Hadamard operator achieved better performances on link predictions for Node2Vec. Node2Vec outperformed baselines when tested on the Facebook dataset (96.8% in Area Under Curve score), the PPI dataset (77.19%), and the arXiv dataset (93.66%) by using the Hadamard operator as the edge forming operator.

Cao, Lu, and Xu (2015) proposed GraRep, which defines k-step loss functions to capture k-step local relational information between different vertices of a graph. This further reveals useful global structural information associated with graphs. To obtain k-step relational information, GraRep manipulates different global transition matrices over the graph. LINE (J. Tang et al., 2015) also considers higher-order relationships among vertices, but it cannot be easily extended to tackle k-order (k ¿ 2) relationships. Moreover, GraRep introduced an Enhanced SGNS (E-SGNS) method for sampling nodes and sequences from weighted graphs. It uses transition probabilities to measure the relationship between nodes. In the experiments, the authors compared GraRep with LINE (J. Tang et al., 2015), DeepWalk (Perozzi et al., 2014), E-SGNS (an enhanced Skip-gram model), and Spectral Clustering (L. Tang & Liu, 2011). In the clustering task for the 20-NewsGroup dataset, GraRep consistently outperformed other baselines and achieved over 81% in normalized mutual information (NMI). In the classification task (via logistic regression) for the Blogcatalog dataset, GraRep also outperformed others and achieved 44.24% in micro-F1 score. In another classification task for the DBLP Network dataset, GraRep performed better (1.0070) than others in the reported Kullback-Leibler divergence.

Several different learning methods were proposed to improve the quality and scalability of node embeddings. Most of them were compared with the methods we've discussed above concerning the link prediction task and the multi-label classification task on different datasets. SDNE (D. Wang et al., 2016) is a semi-supervised model that simultaneously optimizes first-order and second-order proximity when learning node representations. Compared to LINE (J. Tang et al., 2015), SDNE jointly optimizes first-order and second-order proximity so that it can preserve both the local and global network structure and become more robust on sparse networks. Ou, Cui, Pei, Zhang, and Zhu (2016) proposed the HOPE model, which preserves high-order proximity as well as asymmetric transitivity for node embeddings. They generalized their approach to high-order proximities and found a theoretical upper bound on the approximation of their HOPE model. Moreover, they argued that the asymmetric transitivity of directed graphs was not well-considered in previous approaches. They also claimed asymmetric transitivity can be preserved by approximating high-order proximity, and they evaluated these aspects in experiments. X. Wang, Cui, et al. (2017) proposed a Modularized Nonnegative Matrix Factorization (M-NMF) model that considers both the microscopic structure (first-order and second-order proximity) and the mesoscopic structure (communities within a network). The node embeddings are learned through a jointly optimized model.

With the growth in popularity of deep neural networks, network representation learning via deep neural networks has gained increasing attention recently. Pan, Wu, Zhu, Zhang, and Wang (2016) presented TriDNR, a tri-party deep network representation model based on DeepWalk (Perozzi et al., 2014). The model learns from random walk sequences and also combines with neural networks to learn from the node-content relations. TriDNR was compared with several baselines on node classification and achieved 77.7% in F1 score for the Citeseer-M10 network dataset and 74.4% in F1 score for the DBLP network dataset.

Kipf and Welling (2016) introduced the Graph Convolutional Networks (GCN) model for semi-supervised learning on graph-structured data. Instead of using a deep neural network as a complement in node representation learning, the GCN model constructs fast approximate convolutions on graphs directly. The authors focused on solving the semi-supervised node classification problem with multi-layer GCN models. With limited node labels, GCN learns the node characteristics from both graph structures and labels and uses the vectors of hidden layers as node embeddings. The authors compared GCN with ManiReg (Belkin, Niyogi, & Sindhwani, 2006), SemiEmb (Weston, Ratle, Mobahi, & Collobert, 2012), LP (Zhu, Ghahramani, &

Lafferty, 2003), DeepWalk (Perozzi et al., 2014), ICA (Getoor, 2005), and Planetoid (Z. Yang, Cohen, & Salakhudinov, 2016) regarding the task of node classification and showed GCN outperformed all other models in classification accuracy on four datasets (70.3% for Citeseer; 81.5% for Cora; 79% for Pubmed; and 66% for NELL).

2.3 Multi-view Network Representation

Multi-view Network Representation has gained more attention recently since many real-world network datasets contain multiple types of edges between nodes. A view of a network is derived from a single type of proximity or relationship between the nodes, which can be characterized by a set of edges. Qu et al. (2017) proposed MVE, multi-view network representation learning. Since the networks in real-world data are usually complex, the edges in a network can vary when describing different views. To simultaneously consider the complexity of multiple views, MVE learns node representations for a multi-view network and infers robust node representations across different views. For each individual view, MVE follows LINE (J. Tang et al., 2015) to jointly preserve first-order proximity and second-order proximity for optimizing view-specific node embeddings. Next, MVE integrates those view-specific node embeddings by assigning voting weights for robust node representations. The voting weights are learned by the back-propagation of an attention-based weight learning mechanism. The objective functions of the attention mechanism vary according to different predictive tasks. The authors demonstrated that MVE outperformed single-view approaches in F-1 scores on a node classification task for DBLP (74.85%), Flickr (58.95%), and PPI (26.96%) datasets and a link prediction task for YouTube (94.01%) and Twitter (84.98%) datasets.

Ni et al. (2018) proposed a deep-learning multi-view network representation learning method, DMNE, for social and information networks. DMNE is capable of simultaneously learning from multiple networks of different sizes. The learning process is based on cross-network relationships between nodes in different networks. DMNE learns single-network embeddings to capture the network structure and uses cross-network regularization to penalize embedding disagreement (ED) and proximity disagreement (PD) for nodes in different networks. The authors compared DMNE with eight baselines on a multi-label classification task for two news group datasets (i.e., 6-NG and 9-NG), DP-NET (Hwang et al., 2012), DBIS (J. Tang et al., 2008), and CiteSeer-M10 (K. W. Lim & Buntine, 2015). The results showed DMNE consistently outperformed other baselines. Also, among the five datasets, DMNE regularized with ED performed best in the DP-NET dataset, and DMNE regularized with PD performed best in the other four datasets.

MVN2VEC (Shi et al., 2018) learns multi-view network embeddings via a preservation objective and a collaboration objective. The authors introduced MVN2VEC-CON, which reflects collaboration by enforcing constraints on context node embeddings and sharing parameters across different views. They also introduced MVN2VEC-REG, which reflects preservation by regularizing embeddings across different views. The authors compared the two proposed models with MVE (Qu et al., 2017) and DMNE (Ni et al., 2018) on a link prediction task for YouTube, Twitter, and Snapchat datasets. MVN2VEC-CON and MVN2VEC-REG outperformed other baselines and MVN2VEC-REG performed better in most experiments.

Sun, Bui, Hsieh, and Honavar (2018) proposed a single view network embedding approach (SVNE) and extended it to a multi-view version (MVNE) by using graph factorization clustering (GFC) and an objective function that maximizes the agreement between views based on both the local and global structure of the underlying multi-view graph. They compared SVNE with three single-view approaches, LINE (J. Tang et al., 2015), DeepWalk (Perozzi et al., 2014), and Node2Vec (Grover & Leskovec, 2016) on a node label prediction task for Blogcatalog, PPI, Wikipedia datasets. They also compared MVNE with multi-view learning methods, Co-RegSC (Kumar, Rai, & Daume, 2011), MultiNMF (J. Liu, Wang, Gao, & Han, 2013), and MVE (Qu et al., 2017) for Flickr and Last.fm datasets. The results showed both SVNE and MVNE constantly outperform their baselines.

F. Huang et al. (2018) proposed VAE, a multi-view correlation learning based variational auto-encoder. VAE learns node embeddings that consider the link information and multi-modal contents simultaneously. The authors proposed a visual-textual model that uses LSTM with the attention mechanism to learn from pictures and their captions. They compare VAE with four methods that combine content and network structure: SDNE (D. Wang et al., 2016), HNE (Chang et al., 2015), TADW (C. Yang, Liu, Zhao, Sun, & Chang, 2015), and VAEBNR (H. Li, Wang, Yang, & Odagaki, 2017). They conducted a multi-label classification experiment and showed VAE outperformed other baselines for PASCAL (Everingham, Van Gool, Williams, Winn, & Zisserman, 2010), MIR (Huiskes & Lew, 2008), NUS-WIDE (Chua T et al., 2009) datasets. Also, the results suggested that the VAE variant that combined the visual-textual attention model and the multi-view VAE of the mixed mode performed best.

Sun, Wang, Hsieh, Tang, and Honavar (2019) presented MEGAN, a generative adversarial network (GAN) for multi-view network embedding. The authors designed a GAN architecture that included a multi-view generator and a node-pair discriminator. They conducted a node classification task and a link prediction task, and they compared MEGAN with GraphGAN (H. Wang et al., 2017), Node2Vec (Grover & Leskovec, 2016), MVE Qu et al. (2017), DRNE (Tu, Cui, Wang, Yu, & Zhu, 2018), and MNE (H. Zhang, Qiu, Yi, & Song, 2018). The Last.fm and Flickr datasets were used in the experiments. The results showed MEGAN slightly outperformed other baselines in both tasks for the two datasets.

CHAPTER 3. FRAMEWORK AND METHODOLOGY

We first introduce the Yelp datasets used in the experiments as well as the reviewer features used for network construction. Secondly, we discuss how we adopted multi-view network representation learning to generate reviewer embeddings for the first research question: *Do reviewer embeddings learned via multi-view network representation approaches preserve the characteristics of spammers?* We conducted spammer classification experiments and performed clustering analysis on the learned reviewer embeddings. Thirdly, we present the hybrid embeddings that jointly consider review text and the reviewer, which were used in the fake review classification experiments to answer the second research question: *Is the hybrid embedding of a review more sensitive at distinguishing whether a review is fake than the review's text embedding?* The overview of our methodology is illustrated in Figure 3.1, with each of the steps outlined above represented: orange arrows corresponding to the first research question and the green arrows corresponding to the second one.

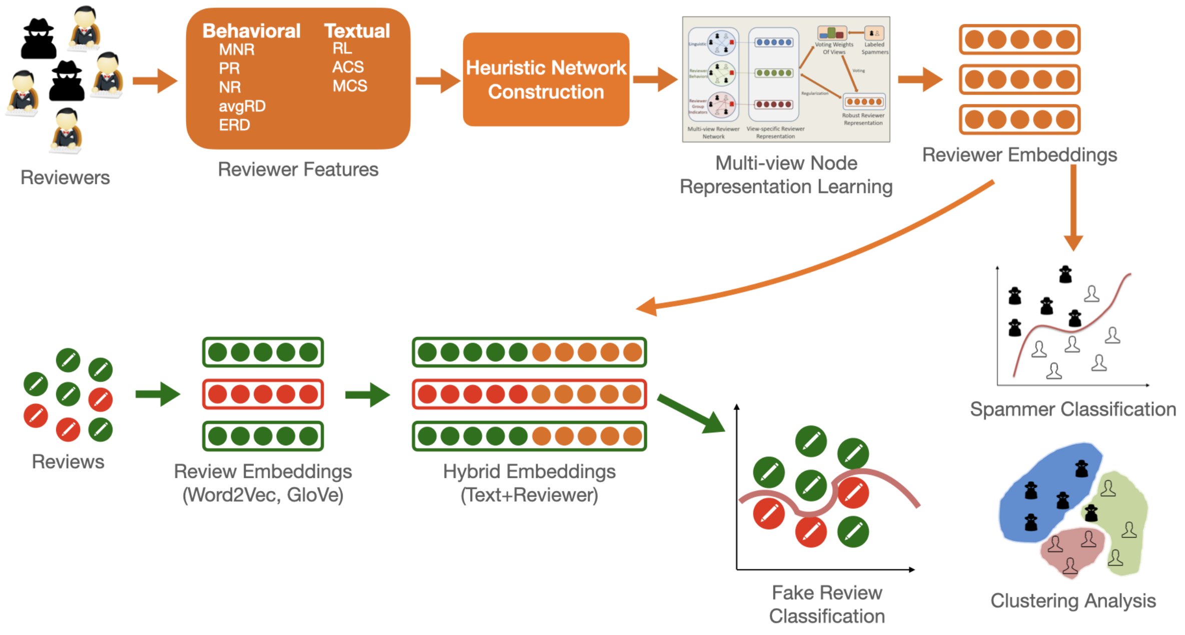

Reviewers
Behavioral
MNR
PR
NR
avgRD
ERD
Textual
RL
ACS
MCS
Reviewer Features
Heuristic Network Construction
Multi-view Node Representation Learning
Reviewer Embeddings
Spammer Classification
Clustering Analysis
Reviews
Review Embeddings (Word2Vec, GloVe)
Hybrid Embeddings (Text+Reviewer)
Fake Review Classification

Figure 3.1. Methodology Overview

3.1 Datasets

We used the datasets YelpChi (Mukherjee, Venkataraman, Liu, & Glance, 2013; Rayana & Akoglu, 2015, 2016), YelpNYC (Rayana & Akoglu, 2015, 2016), and YelpZIP (Rayana & Akoglu, 2015, 2016) to train the reviewer embeddings and perform reviewer/review classification tasks. There are several reasons why we chose these datasets: 1) The three Yelp datasets have been broadly used in many previous studies that were discussed in the literature review. 2) Since our study focuses on learning reviewer representation and uses them to enhance fake review identification, the Yelp datasets are useful because they contain not only the review information but also information about reviewers, which are not provided in many other datasets.

The statistics of the datasets are described in Table 3.1. Each review has *product id*, *user id*, *timestamp*, *rating*, *review content*, and a *filtered* label. The *filtered* labels are given by Yelp based on their filtering algorithm, which has evolved over the years since their launch in 2005 to filter shill and fake reviews.

Table 3.1. *The statistics of the Yelp datasets*

Dataset	**# of reviews**	**# of reviewers**
YelpChi	67,395 (13.23% of filtered reviews)	38,063 (20.33% of spammers)
YelpNYC	359,052 (10.27% of filtered reviews)	160,225 (17.79% of spammers)
YelpZIP	608,598 (13.22% of filtered reviews)	260,277 (23.91% of spammers)

3.2 Reviewer Features

We followed Rayana and Akoglu (2015) and selected eight features that are related to a reviewer's behavioral and textual patterns (see Table 3.2). The behavioral statistical features are comprised of the maximum number of reviews written in a day (MNR), the ratio of positive reviews with 4 or 5 stars (PR), the ratio of negative reviews with 1 or 2 stars (NR), the average rating deviation of a user (avgRD), and the entropy of the rating distribution of a user's reviews (ERD). On the other hand, the textual features are comprised of the average review length in number of words (RL), the average content cosine similarity between any pairs of users (ACS), and the maximum content similarity between any pairs of users (MCS). Those features have been used in many studies of opinion spam detection (Fei et al., 2013; Jindal & Liu, 2008; E.-P. Lim, Nguyen, Jindal, Liu, & Lauw, 2010; Mukherjee et al., 2012; Mukherjee, Venkataraman, Liu, & Glance, 2013; Rayana & Akoglu, 2015).

Table 3.2. *Reviewer Features*

Name	**Description**
MNR	Maximum number of reviews written in a day
PR	Ratio of positive reviews (4-5 star)
NR	Ratio of negative reviews (1-2 star)
avgRD	Average rating deviation of user *i*'s reviews
ERD	Entropy of rating distribution of user's reviews
RL	Average review length in number of words
ACS	Average content similarity-pairwise cosine similarity among user's reviews, where a review is represented as a bag-of-bigrams
MCS	Maximum content similarity—maximum cosine similarity among all user pairs

Tables 3.3, 3.4, and 3.5 show the mean and standard deviation of reviewer features of YelpChi, YelpNYC, and YelpZIP. The distributions of most features are similar among the three datasets.

Table 3.3. *The statistics of reviewer features of YelpChi*

YelpChi	MNR	PR	RL	NR	avgRD	ERD	ACS	MCS
mean	1.136	0.751	126.533	0.138	0.175	0.473	0.008	0.013
std	0.466	0.393	107.468	0.321	0.364	0.837	0.021	0.031

Table 3.4. *The statistics of reviewer features of YelpNYC*

YelpNYC	MNR	PR	RL	NR	avgRD	ERD	ACS	MCS
mean	1.212	0.78	100.411	0.122	0.186	0.579	0.008	0.015
std	0.648	0.37	89.574	0.302	0.366	0.987	0.02	0.037

Table 3.5. *The statistics of reviewer features of YelpZIP*

YelpZIP	MNR	PR	RL	NR	avgRD	ERD	ACS	MCS
mean	1.252	0.742	100.025	0.163	0.21	0.601	0.008	0.017
std	0.76	0.391	89.926	0.34	0.402	1.013	0.024	0.044

3.3 Reviewer Network Construction

Since the Yelp datasets do not contain reviewer networks, we proposed a heuristic reviewer network construction to build edges between reviewer nodes. Given a reviewer set R and the reviewers' feature values F regarding a target feature, the following are the steps of the process:

1. Initialize an empty reviewer network N.
2. For each reviewer $r_a \in R$ and its feature value f_a, where $f_a \in F$, calculate the mean absolute deviation $d_{a*} = \frac{\Sigma|f_a - f_k|}{|R|-1}, \forall f_k \in F$ and $r_k \neq r_a$.
3. For each reviewer pair (r_a, r_b), where $r_a \neq r_b$, if $\frac{1}{d_{ab}+1} \geq \frac{1}{d_{a*}+1}$, add an edge (r_a, r_b) with the weight $\frac{1}{d_{ab}+1}$ into N.
4. Return N as the reviewer network regarding the target feature.

We explain the network construction in a running example. Let's say we want to build a reviewer network for four reviewers, r_a, r_b, r_c, r_d, and they all have a feature f. r_a has $f_a = 2$, r_b has $f_b = 3$, r_c has $f_c = 1$, and r_d has $f_d = 4$. Firstly, we initialize an empty reviewer network N. Secondly, we calculate the mean absolute deviations $d_{a*} = \frac{4}{3}$, $d_{b*} = \frac{4}{3}$, $d_{c*} = 2$, $d_{d*} = 2$. Thirdly, we add the following edges into N:

- Edge (r_a, r_b) with the weight $\frac{1}{2}$, since $\frac{1}{d_{ab}+1} = \frac{1}{2} \geq \frac{3}{7} = \frac{1}{d_{a*}+1}$.
- Edge (r_a, r_c) with the weight $\frac{1}{2}$, since $\frac{1}{d_{ac}+1} = \frac{1}{2} \geq \frac{3}{7} = \frac{1}{d_{a*}+1}$.
- Edge (r_b, r_d) with the weight $\frac{1}{2}$, since $\frac{1}{d_{bd}+1} = \frac{1}{2} \geq \frac{3}{7} = \frac{1}{d_{b*}+1}$.

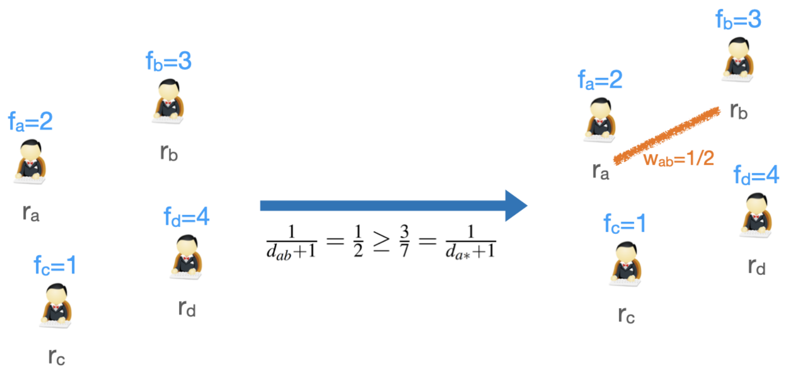


Figure 3.2. A running example of Reviewer Network Construction: build edge (r_a, r_b)

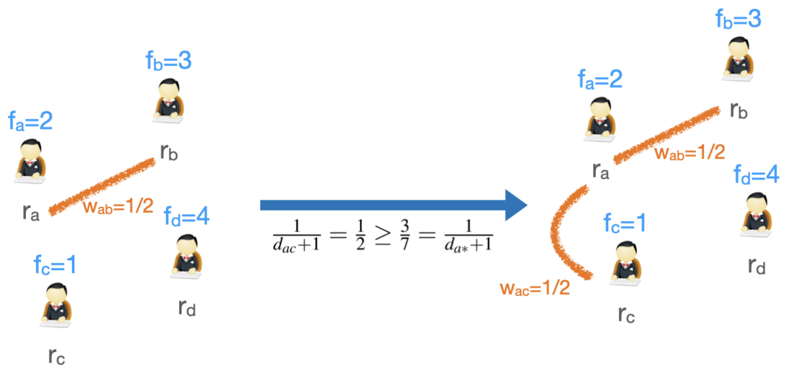


Figure 3.3. A running example of Reviewer Network Construction: build edge (r_a, r_c)

The other node pairs do not form any edges because their weights are smaller than the mean absolute deviation of nodes. Lastly, we return $N = \{(r_a, r_b) : \frac{1}{2}, (r_a, r_c) : \frac{1}{2}, (r_b, r_d) : \frac{1}{2}\}$ as the

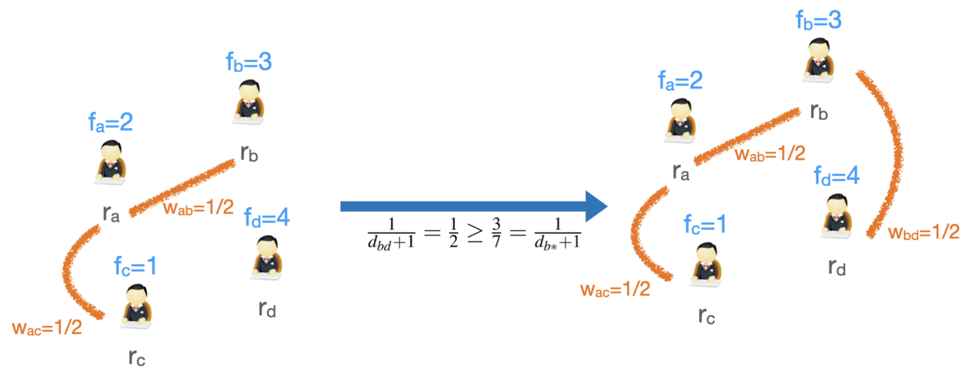


Figure 3.4. A running example of Reviewer Network Construction: build edge (r_b, r_d)

reviewer network regarding the target feature f. The network construction method builds weighted edges for reviewers based on the deviation of the feature values between reviewers; meanwhile, it reduces the number of edges by keeping only edges with higher weights (i.e., reviewer pairs that have smaller deviations).

3.4 Multi-view Reviewer Representation Learning

As illustrated in Figure 3.5, we learn reviewer embeddings with regard to a reviewer's textual and behavioral characteristics via MVE (Qu et al., 2017) for reviewer networks.

For each feature view, we firstly conduct the reviewer network construction process introduced in the previous section. Secondly, we adopt MVE to learn view-specific embeddings. Thirdly, we use MVE to learn the voting weights for those views. The edge weights are calculated by the deviation-based scores introduced in the reviewer network construction. Lastly, the robust reviewer embeddings are obtained by the linear combination of weighted view-specific reviewer embeddings.

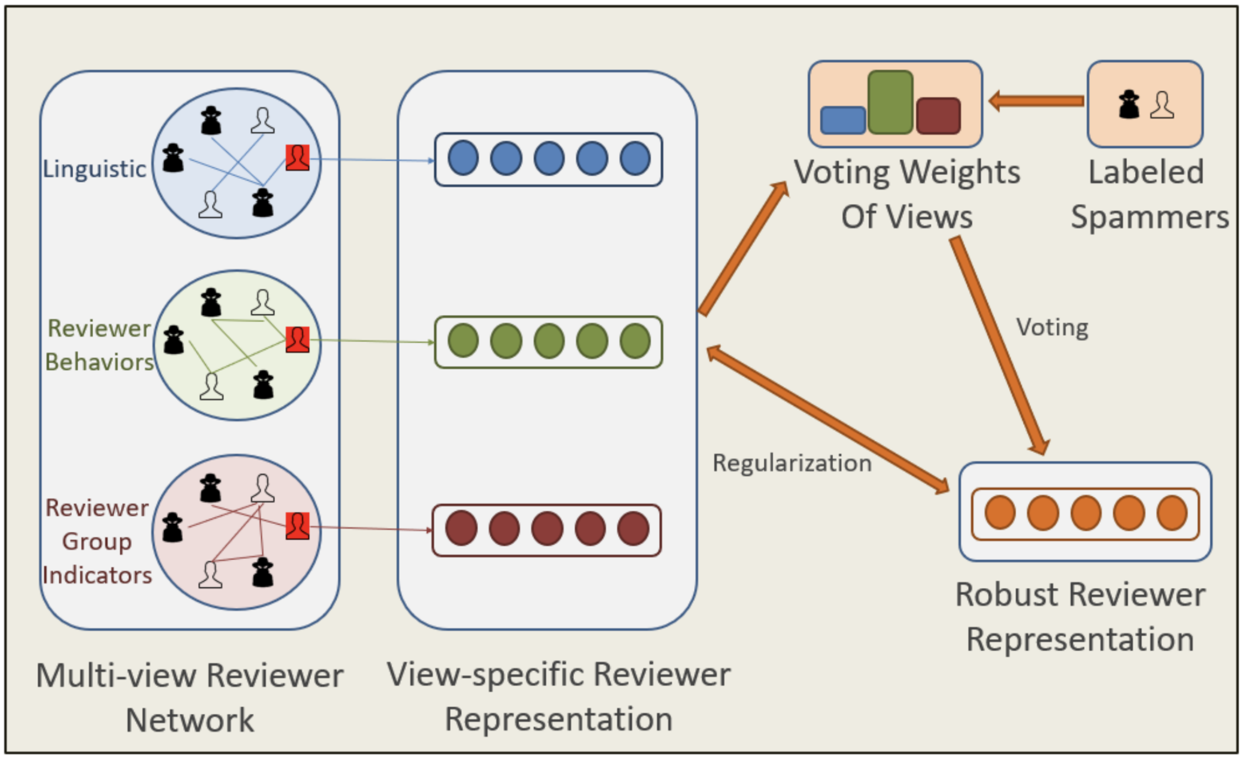

Linguistic
Reviewer Behaviors
Reviewer Group Indicators
Multi-view Reviewer Network
View-specific Reviewer Representation
Voting Weights Of Views
Labeled Spammers
Voting
Regularization
Robust Reviewer Representation

Figure 3.5. Multi-view Reviewer Network

Furthermore, we explore the different parameters of MVE on reviewer representation learning: 1) The different sizes of MVE embeddings; 2) the different numbers of training epochs of MVE.

<u>3.5 Word Embedding</u>

Word2vec is the language model introduced by Mikolov, Sutskever, et al. (2013). It is effective at generating word embeddings (i.e., vectors) that capture the characteristics of words based on the co-occurrences of words used in the corpus. The learned embeddings can be used to infer the relationships between words by vector arithmetic. The Word2Vec models are shallow, two-layer neural networks. Word2vec takes a large corpus of text as the input and generates a low-dimensional (e.g., 300) vector space. Each word (but not each sense of a word) is represented as a unique vector in the vector space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located close to one another in the space. For instance, we know Paris is the capital of France and London is the capital of England. In many news articles, the text pattern of word occurrences between ‘Paris‘ and ‘France‘ is similar to that between ‘London‘ and ‘England‘. Thus, if we train a Word2Vec model that learns from a large number of news articles, and we take the word embeddings of ‘Paris‘, ‘France‘, and ‘London‘ to do a vector arithmetic ($‘Paris‘ + ‘France‘ - ‘London‘$), the result would be very close to the word embedding of ‘England‘. There are more examples of this kind of word analogy in Figure 3.6.

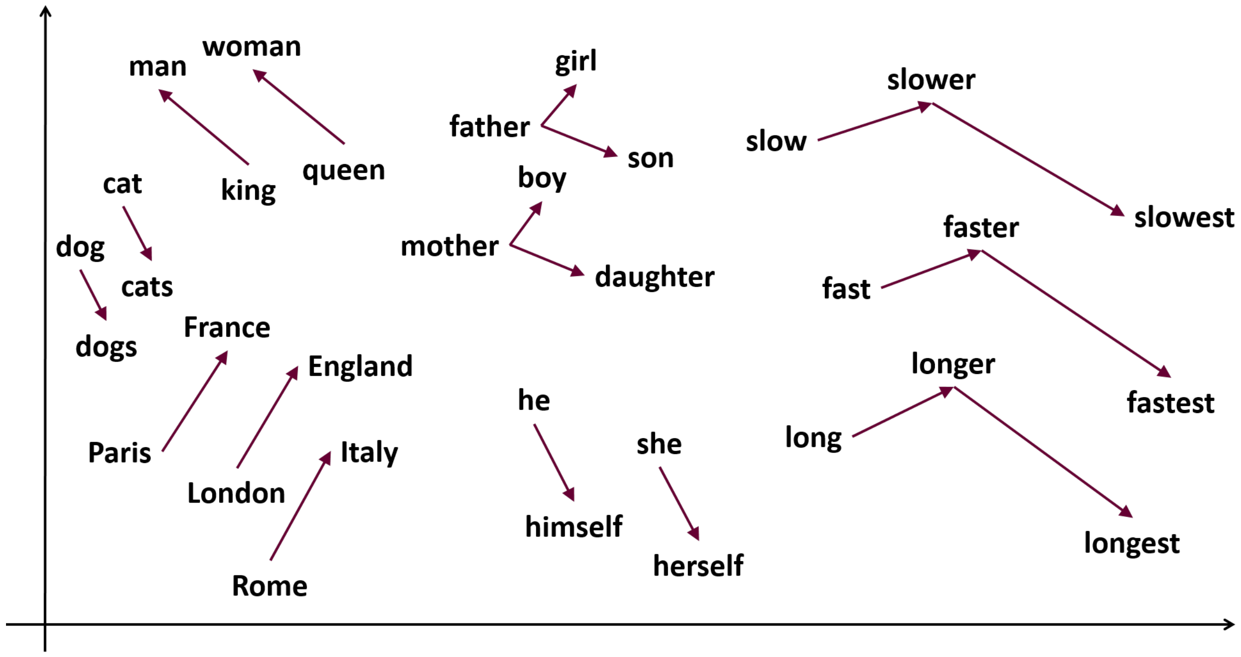


Figure 3.6. Word2Vec example (*NLP with gensim (word2vec)*, 2021)

GloVe, which was proposed by Pennington, Socher, and Manning (2014), learns a specific weighted least squares model that trains on global word-word co-occurrence counts. GloVe combines the advantages of two major model families for learning word vectors: 1) global matrix factorization methods, such as latent semantic analysis (LSA) (Dumais, 2004), and 2) local context window methods, such as Word2Vec. According to the experimental results of Pennington et al. (2014), GloVe outperformed Word2Vec models and other baselines using single value decomposition (SVD) (Van Loan, 1976) with 75% accuracy on the word analogy dataset. However, when compared to Word2Vec in various publications, each of the models can outperform another depending on the task and corpus.

Recently, contextual text embedding approaches, such as BERT (Devlin, Chang, Lee, & Toutanova, 2018), have drawn great attention because they are much more effective in many NLP tasks compared to static text embeddings (e.g., Word2vec and GloVe). In this study, we consider only static text embeddings in our experiments. There are two reasons for this: 1) Our goal is to examine whether the multi-view reviewer embedding of a review's reviewer can complement any review's text embedding on fake review identification. We do not focus on comparing which type of text embedding method performs best, but we consider static embeddings to perform suitably well for our task. 2) The training and fine-tuned processes of contextual approaches are, computationally, extremely expensive. Even though contextual embedding methods might achieve better performance than static ones, this does not increase the value of the research since the research does not pursue the highest performance of classification. Based on these two reasons, we consider only Word2vec and GloVe in our experiments.

3.6 Hybrid Embedding

Since it is difficult to distinguish whether a review is deceptive or not without considering the reviewer's previous review patterns, we are interested in generating hybrid representations by jointly considering textual embeddings as well as multi-view reviewer embeddings. First, we compute the review embeddings created by pre-trained Word2Vec (Mikolov, Sutskever, et al., 2013) and GloVe (Pennington et al., 2014) respectively. Second, for each review, we find its reviewer and concatenate the corresponding reviewer embedding and the review's text embedding as the hybrid embedding. Third, we treat the hybrid embedding as the features and conduct fake review classification experiments. We compare the performance of models learned by the text embeddings only (Word2Vec-only, GloVe-only) with those learned by the hybrid embeddings (Word2Vec+MVE, GloVe+MVE).

3.7 Data Analysis

3.7.1 Classification Analysis

The goal of classification is to predict the target class (e.g., the filtered labels in the Yelp datasets) accurately for each sample of data. The following metrics are widely used in classification tasks:

- Precision: The fraction of true positive predictions among the instances that were predicted as positive. The equation is illustrated as Equation 3.1.

$$Precision = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{3.1}$$

- Recall: The fraction of true positive predictions among the truly positive instances. The equation is illustrated as Equation 3.2.

$$Recall = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{3.2}$$

- F1 score: The harmonic mean of precision and recall. The equation is illustrated as Equation 3.3.

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} \tag{3.3}$$

Here is a detailed explanation of the relationship between precision and recall. When a classification model is designed to be stricter on predicting positive cases, it tends to predict fewer false positives but more false negatives. On the contrary, if a model is less strict, it predicts more false positives but fewer false negatives. For instance, suppose we have a test kit to evaluate whether a person caught the flu. If the test kit were designed to predict positive cases only for patients with multiple apparent symptoms so that the hospital doesn't waste medical resources on false alarms, the test results would be higher in precision (i.e., fewer false positives) but lower in recall (i.e., more false negatives). Conversely, if the test kit were designed to predict positive cases even when the patients have minor symptoms, because the hospital doesn't want to miss any infected people, the test results would be lower in precision (i.e., more false positives) but higher in recall (i.e., fewer false negatives). The trade-off between precision and recall highly depends on the goal we want to achieve, and there is no exact answer for which setting is better. Therefore, choosing the metrics for model evaluation is extremely critical. To jointly consider both precision and recall, the F1 score is a harmonic mean of both and is a more comprehensive measure when precision and recall are equally important to the classification task.

3.7.2 Cluster Analysis

We analyze clusters in the node classification task discussed in previous sections. In terms of cluster analysis, there is no best measure for evaluating cluster quality. However, considering a mix of internal and external quality criteria can provide a comprehensive view for evaluating clustering approaches (Bhatnagar & Ahuja, 2010). Therefore, we adopt two widely used metrics: purity (Zhao & Karypis, 2001) as an external criterion and normalized mutual information (NMI) (Strehl & Ghosh, 2002) as an internal criterion to evaluate the quality of the clustering results.

Purity measures the extent to which the documents in a cluster are from primarily one specific class. Given that there are k clusters formed by a total of n documents and that each document was labeled by one of I classes, the purity of the r-th cluster S_r with size n_r is defined by Equation 3.4.

$$P(S_r) = \frac{1}{n_r} max_i(n_r^i), \forall i \in 1, 2, ..., |I| \tag{3.4}$$

n_r^i is the number of documents of the i-th class that were assigned to the r-th cluster. The overall purity of the clustering solution is obtained as a weighted sum of the individual cluster purities and is given by Equation 3.5. In general, the larger the purity value, the better the clustering solution is.

$$Purity = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \tag{3.5}$$

NMI provides an indication of the shared information between a pair of clusters. Given X and Y are two random variables described by two different cluster labelings, let $I(X,Y)$ denotes the mutual information between X and Y, $H(X)$ and $H(Y)$ denote the entropy of X, Y. The equation for NMI is shown in Equation 3.6.

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)H(Y)}} \tag{3.6}$$

The value of NMI is a fraction between 0 and 1, with 0 indicating that the two clusters do not share the same information and 1 indicating that the two clusters are exactly the same. In general, high mutual information means a large reduction in the difference between two clusters; low mutual information means a small reduction; and zero mutual information between two clusters means the two distributions are independent and do not share mutual information.

CHAPTER 4. EXPERIMENTS

We evaluate the multi-view reviewer embeddings on two tasks: spammer classification and reviewer clustering. The two experiments address our first research question, *"Do reviewer embeddings learned via multi-view network representation approaches preserve the characteristics of spammers"*. Next, we evaluate the hybrid embeddings (i.e., text+reviewer) on a fake review classification task. Also, we evaluate each feature's contribution to fake review classification by a feature ablation study. The experiments address our second research question, *"Is the hybrid embedding of a review more sensitive at distinguishing whether a review is fake than the review's text embedding"*

Moreover, we investigate the effectiveness of the reviewer network construction approach proposed in this study. Lastly, we examine whether setting the spammer threshold at a different level impacts the reviewer embeddings when they were used in fake review classification. The setup of the experiments is described in the following sections.

4.1 Experimental Setup

As discussed in the previous chapter, we conducted experiments on the datasets YelpChi, YelpNYC, and YelpZIP. Eight features were considered, including MNR, PR, NR, avgRD, ERD, RL, ACS, and MCS. For each feature, we built a view (i.e., a network derived from a single type of proximity or relationship between the nodes) with edges and weights based on the reviewer network construction approach introduced in chapter 3. Since the size of spammers and the size of non-spammers are not balanced in the original datasets (as described in Table 3.1), we randomly sampled the same amount of spammers and non-spammers. In addition, we investigated the spammer threshold parameter X, which defines a user as a spammer if the user has written more than X filtered reviews. We set the spammer threshold as 4 to separate spammers and non-spammers based on our experiments described in section 4.5. Next, we followed the best practices provided by Qu et al. (2017) to choose the parameters for MVE: the number of negative examples for negative sampling was set at 5, the depth of random walk was set at 1, and the number of training samples was set at 3 million in an epoch. The embedding size and training epochs vary for different tasks.

4.2 Spammer Classification

In the spammer classification experiments, we investigated to what extent the learned reviewer embeddings capture the characteristics of spammers. The reviewer embeddings were taken as features, and the labels of reviewers were the prediction target. For parameter tuning, we tested a list of embedding sizes [3, 10, 20, 50, 100] and a list of MVE training epochs [1, 5, 10, 20]. Since the best parameter may vary from different datasets, thus, instead of exploring all possible options to find the best one, we focused on evaluating how different levels of embedding sizes and training epoch numbers influence the performance of spammer classification. We used the GridSearchCV packages (scikit-learn version 0.24.2) in the scikit-learn toolkits. We adopted the random forest classifier as the classification method. We chose the random forest classifier because it outperformed other classifiers (i.e., logistic regression, SVM) in preliminary tests. For each dataset, we sampled a balanced set with the same number of spammers and non-spammers. The balanced set is then randomly split into a 80% training set and a 20% testing set based on spammer labels. Thus, one reviewer does not exist in both the training and testing sets. The best models were selected through 10-fold cross validation based on the parameter settings ’n estimators’: [100,200], ’min samples leaf’: [1,2], ’max features’: [”auto”, ”sqrt”, ”log2”]. Precision, Recall, and F1 scores are reported as the comparison metrics in Table 4.1, 4.2, and 4.3.

Table 4.1. *Spammer classification with different embedding sizes (Precision)*

embedding size	3	10	20	50	100
YelpChi	0.923	0.722	0.923	0.929	0.929
YelpNYC	0.838	0.871	0.807	0.866	0.841
YelpZIP	0.804	0.837	0.845	0.859	0.871

According to the results in Table 4.3, the larger embedding sizes have better performance on reviewer classification in general (as shown in Figure 4.1). For YelpChi, the best performance was .929 in F1 score when the embedding size was 50 or 100; for YelpNYC, the best performance

Table 4.2. *Spammer classification with different embedding sizes (Recall)*

embedding size	3	10	20	50	100
YelpChi	0.857	0.929	0.857	0.929	0.929
YelpNYC	0.912	0.941	0.956	0.949	0.934
YelpZIP	0.935	0.941	0.966	0.966	0.947

Table 4.3. *Spammer classification with different embedding sizes (F1)*

embedding size	3	10	20	50	100
YelpChi	0.889	0.813	0.889	**0.929**	**0.929**
YelpNYC	0.873	**0.905**	0.875	**0.905**	0.885
YelpZIP	0.865	0.886	0.901	**0.909**	0.907

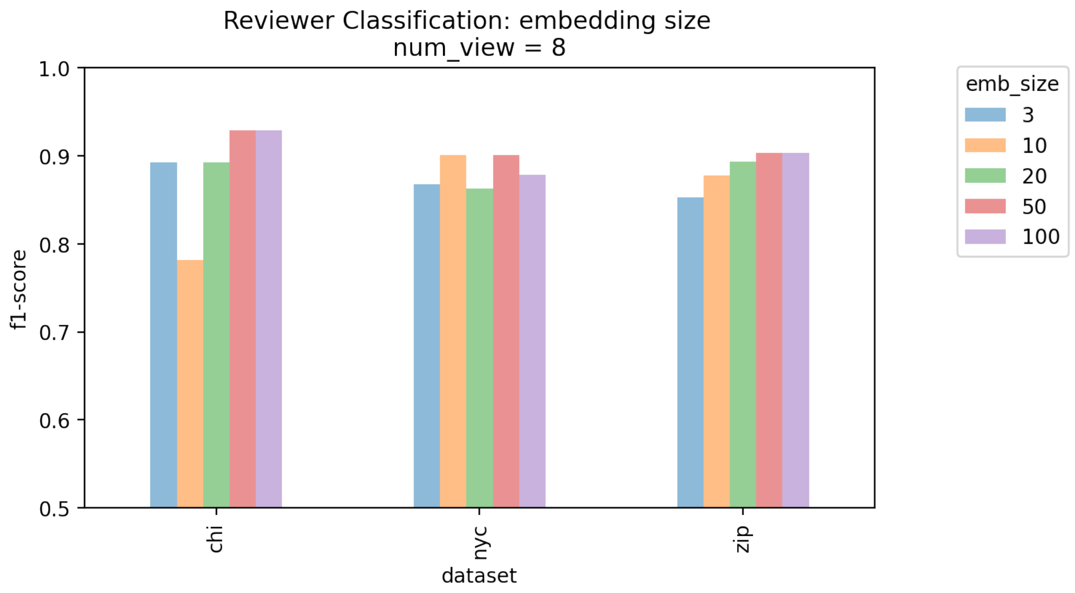


Figure 4.1. Spammer classification with different training sizes (F1)

was .905 when the embedding size was 10 or 50; and for YelpZIP, the best performance was .909 when the embedding size was 50.

Next, we evaluated the impact of different numbers of training epochs on reviewer classification. According to the results in Table 4.6, for YelpChi, the best performance was 1.00 in F1 score when the training epochs was 10; for YelpNYC, the best performance was .915 when the training epochs was 10; and for YelpZIP, the best performance was .907 when the training epochs

Table 4.4. *Spammer classification with different training epochs (Precision)*

training epochs	1	5	10	20
YelpChi	0.875	0.933	1.000	0.929
YelpNYC	0.859	0.813	0.854	0.841
YelpZIP	0.857	0.837	0.856	0.871

Table 4.5. *Spammer classification with different training epochs (Recall)*

training epochs	1	5	10	20
YelpChi	1.000	1.000	1.000	0.929
YelpNYC	0.941	0.926	0.985	0.934
YelpZIP	0.896	0.885	0.963	0.947

Table 4.6. *Spammer classification with different training epochs (F1)*

training epochs	1	5	10	20
YelpChi	0.933	0.966	**1.000**	0.929
YelpNYC	0.898	0.866	**0.915**	0.885
YelpZIP	0.901	0.892	0.906	**0.907**

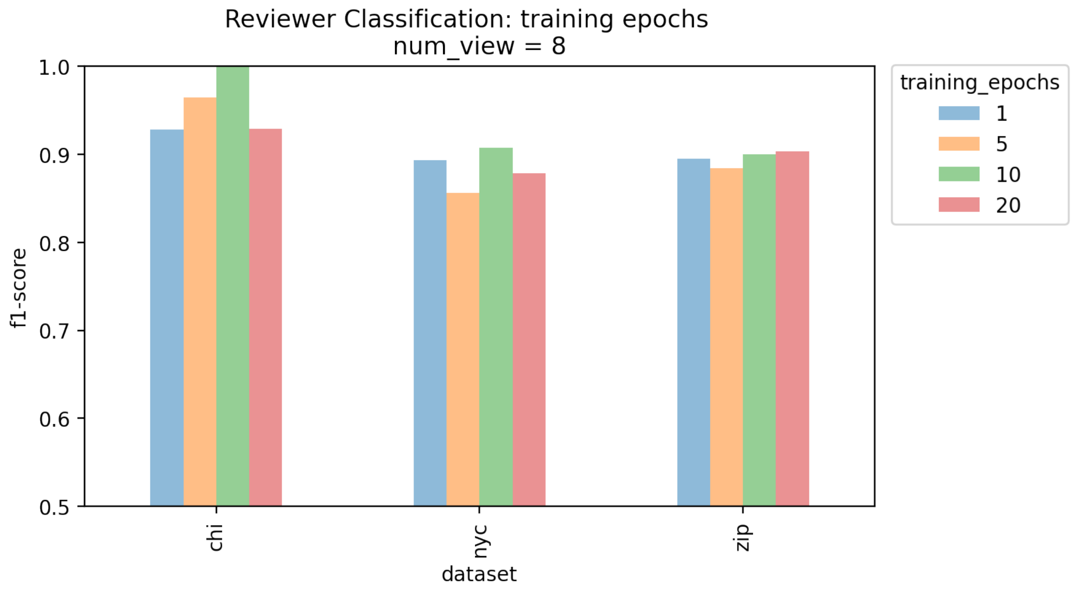


Figure 4.2. Spammer classification with different training epochs (F1)

was 20. As shown in Figure 4.2, there is no consistent choice for the best number of training epochs.

4.2.1 Summary

In summary, the reviewer embeddings capture the characteristics of spammers well. The experiments achieved higher than 0.9 in F1 scores in the three datasets. Furthermore, the results suggest that the larger embedding sizes perform better, but the performance does not improve much for the datasets after the size is greater than 50. On the other hand, it is interesting to see that increasing training epochs does not play a crucial role after 5 to 10 epochs, which suggests that the MVE training process is efficient enough to learn from the Yelp datasets through a small number of training epochs.

This experiment partially answers the first research question, as we have shown reviewer embeddings are sensitive at classifying spammers and non-spammers with over .9 in F1 scores. The numbers are promising; however, is this actually because of the effectiveness of reviewer embeddings, or is it due to the strength of the classification algorithms? To answer this question, we took a further step and investigated the clusters formed by reviewer embeddings in the next experiment.

4.3 Reviewer Clustering Analysis

Instead of spammer classification, we investigated clusters formed by reviewer embeddings in the second experiment. Firstly, we conducted an unsupervised hierarchical clustering of the reviewer embeddings by grouping the embeddings into two clusters. We then compared the difference between the assigned cluster and the spammer label of the reviewers. In the hierarchical clustering experiments, we adopted the agglomerative clustering in scikit-learn and set the number of clusters to two. Agglomerative clustering is a bottom-up algorithm that builds nested clusters by minimizing the variance of the clusters being merged recursively.

We compared the clusters built by using 1) the reviewer's multi-view embeddings and 2) the reviewer's eight original features as the features respectively. Based on the results in Tables

4.3 and 4.6, we used different embedding sizes and training epochs for different datasets (YelpChi: embedding size=100, training epochs=10; YelpNYC: embedding size=50, training epochs=10; YelpZIP: embedding size=50, training epochs=20). The results of Purity and NMI scores are as shown in Table 4.7 for multi-view reviewer embeddings versus original features. The Purity scores reflect the degree to which the spammers and non-spammers are "correctly" grouped into two clusters based on the reviewer's embedding, while the mutual information scores indicate the degree of reduction in uncertainty. Using multi-view reviewer embeddings as features resulted in better purity scores (.674 for YelpChi, .702 for YelpNYC, .741 for YelpZIP) and NMI scores (.114 for YelpChi, .146 for YelpNYC, .189 for YelpZIP) than using the original reviewer features. The results suggest that both Purity and NMI are enhanced when using multi-view reviewer embeddings rather than using the original features directly.

Table 4.7. *Reviewer Embeddings vs. Reviewers' Original Features*

dataset/metrics	Purity (Multi-view)	Purity (Original)	NMI (Multi-view)	NMI (Original)
YelpChi	**0.674**	0.645	**0.114**	0.075
YelpNYC	**0.702**	0.571	**0.146**	0.049
YelpZIP	**0.741**	0.542	**0.189**	0.030

Next, we performed dimensional reduction on the reviewer embeddings and visualized the points by using PCA (Wold, Esbensen, & Geladi, 1987) and t-SNE (Van der Maaten & Hinton, 2008). The experiments bring a different perspective than scores and illustrate how multi-view reviewer embeddings can be used to distinguish spammers and non-spammers. Concerning the configuration of the visualization, we set the dimensions to two for plotting the 2-D figures. We used the reviewer embeddings generated with an embedding size of 100 and a training epoch size of 20. The PCA plots of the three datasets are shown in Figures 4.3, 4.4, and 4.5 and t-SNE plots are shown in Figures 4.6, 4.7, and 4.8. Through the two different visualization approaches, we found that the multi-view reviewer embeddings can clearly distinguish spammers and non-spammers.

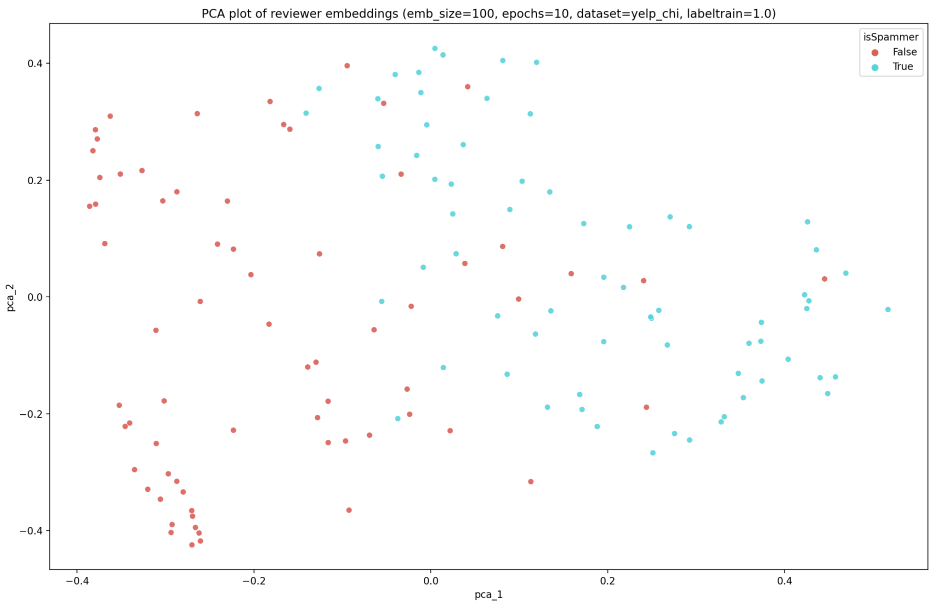


Figure 4.3. PCA plot of reviewer embeddings of YelpChi

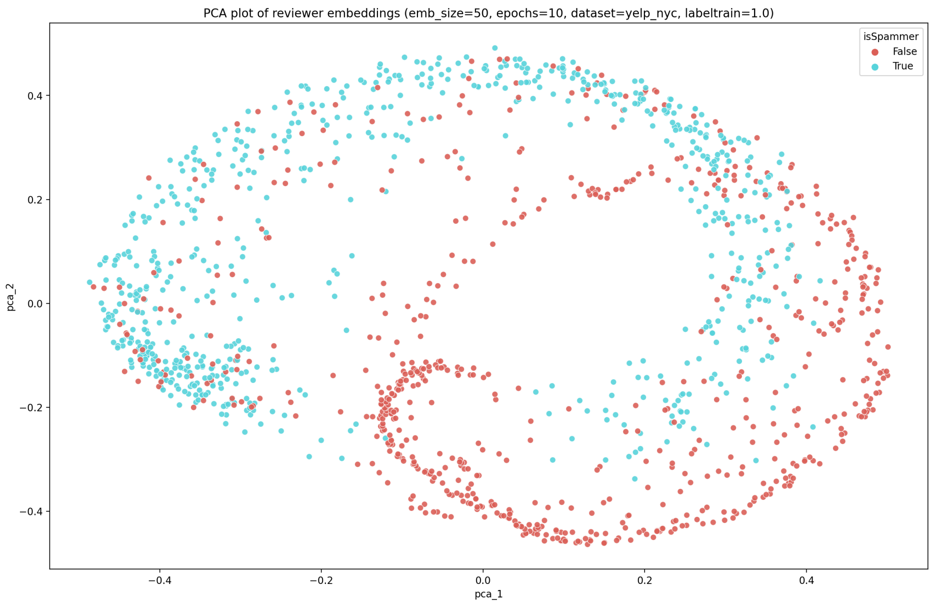


Figure 4.4. PCA plot of reviewer embeddings of YelpNYC

4.3.1 Summary

In summary, the experiments showed that multi-view reviewer embeddings preserve the characteristics of spammers through unsupervised approaches. Firstly, we compared the Purity and NMI scores between the clusters generated from multi-view reviewer embeddings and from

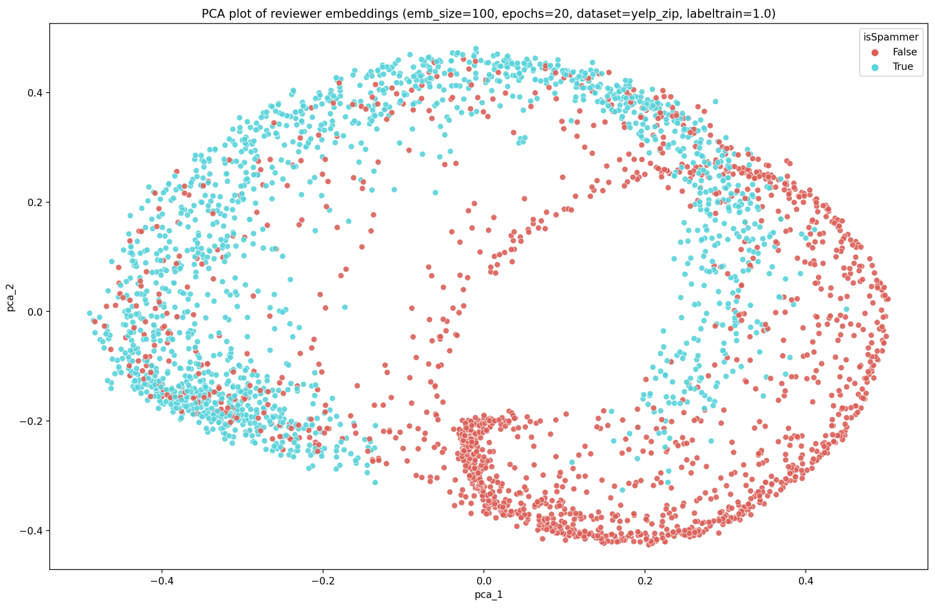


Figure 4.5. PCA plot of reviewer embeddings of YelpZIP

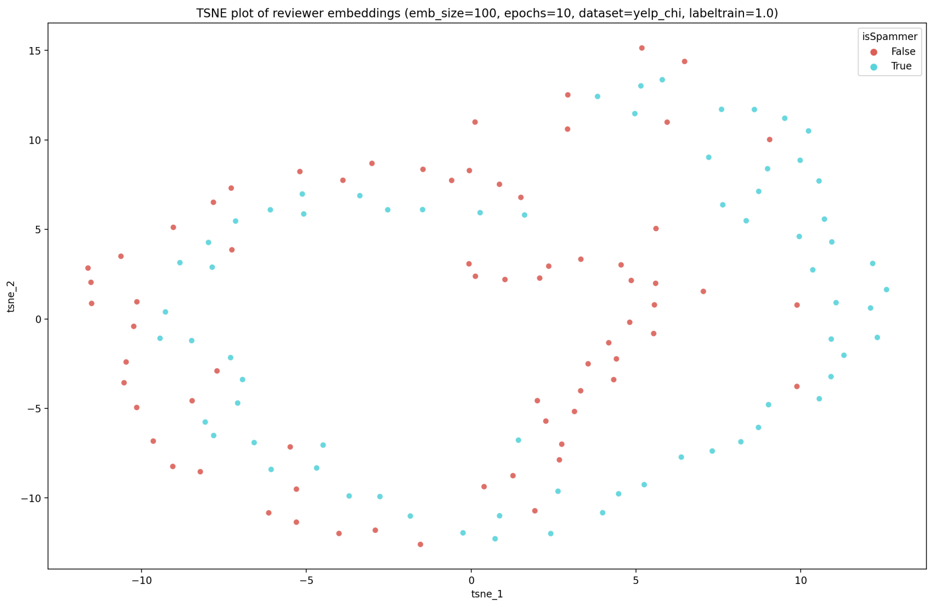


Figure 4.6. t-SNE plot of reviewer embeddings of YelpChi

original reviewer features. The results suggested that the former captures the characteristics of spammers better. Secondly, we observed that multi-view reviewer embeddings can effectively separate spammers and non-spammers through the visualization of PCA and t-SNE plots.

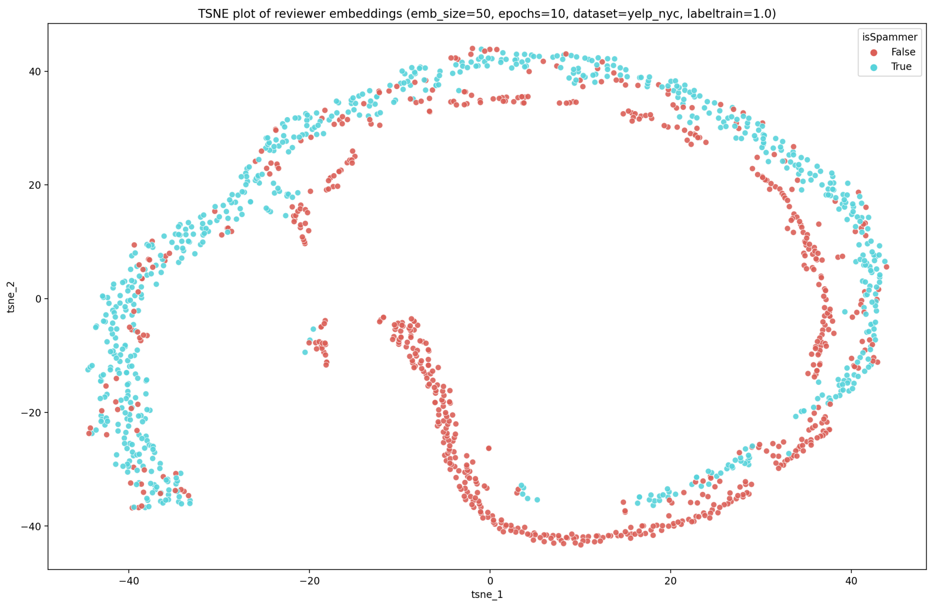


Figure 4.7. t-SNE plot of reviewer embeddings of YelpNYC

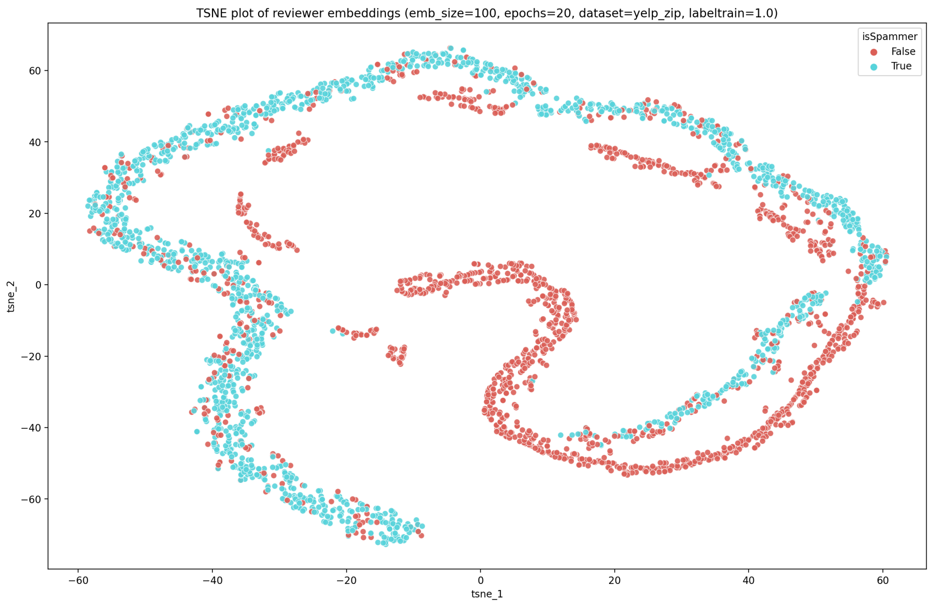


Figure 4.8. t-SNE plot of reviewer embeddings of YelpZIP

The experiment provided another aspect of unsupervised learning that can explain why reviewer embeddings are effective at separating spammers from non-spammers. According to the results, the clusters formed by the vectors of reviewer embeddings aligned with the spammer labels. Combining the results of spammer classification and cluster analysis, we have answered the first research question from the perspectives of both supervised and unsupervised learning.

4.4 Fake Review Classification

In the previous experiments, we showed that multi-view reviewer embeddings are effective at distinguishing spammers and non-spammers. To answer the second research question, our next step is to incorporate the reviewer embeddings with the text embeddings of reviews and investigate to what extent collaboration could improve review identification. As we discussed in previous chapters, it's difficult to recognize untruthful reviews simply based on the review text. Thus, considering text review along with its reviewer's information provides us with a more comprehensive perspective on fake review identification. In fake review classification, we examine whether using a review's hybrid embedding (i.e., text + reviewer) is more sensitive on fake review identification than using only a review's text embedding.

Text embedding approaches, such as Word2Vec (Mikolov, Sutskever, et al., 2013) and GloVe (Pennington et al., 2014), have been widely adopted in various natural language processing (NLP) and understanding tasks. Those approaches have been proved to be able to capture the statistical and semantics patterns in articles. We use a pre-trained Word2Vec model (Google News dataset, 100 billion tokens, 3 million vocabularies and phrases, and an embedding size of 300) and a pre-trained GloVe model (840 billion tokens crawled from the internet, 2.2 million vocabularies, and an embedding size of 300). The Word2Vec model can be accessed from the Github pages of Mihháltz (2021), and the GloVe model is from the web pages of Pennington, Socher, and Manning (2021).

The experimental setup is similar to the reviewer classification experiments. We used the scikit-learn GridSearchCV packages and Random Forest classifiers. For each dataset, we sample a balanced set with the same number of fake reviews and genuine reviews. The balanced set is then sorted by reviewer ids. After that, the indices of reviews of each reviewer are locally shuffled (i.e., the range of indices of a reviewer's reviews does not change). Next, we select the first 80% of reviews as the training set and the other 20% of reviews as the testing set. This sampling ensures there is no more than one reviewer who exists in both training and testing sets. Also, it ensures the size of the training set is exactly 80% and of the testing set is exactly 20% of the balanced set. The best random forest models were selected through 10-fold cross validation based on the parameter settings {'n_estimators': [100,200], 'min_samples_leaf': [1,2], 'max_features': ["auto", "sqrt", "log2"]}. F1 scores were reported as the comparison metrics. For the Word2Vec-only and GloVe-only experiments, we averaged the word embeddings as each review's embedding based on the word occurrences in the review. For the Word2Vec+Reviewer and GloVe+Reviewer experiments, we concatenated the review's text embedding with its reviewer embedding as the hybrid embedding. The hybrid embedding was used as the review's embedding. After that, we took each review's embedding as the features in the fake review classification tasks.

4.4.1 Comparison of different embedding sizes

We examined how different sizes [3, 10, 20, 50, 100] of reviewer embeddings impacted the performance of review classification. The precision, recall, and F1 scores regarding GloVe+Reviewer are reported in Tables 4.8, 4.9, and 4.10. The number regarding Word2Vec+Reviewer are reported in Tables 4.11, 4.12, and 4.13.

For the GloVe experiments, the models with GloVe+Reviewer (size=100) embeddings achieved .952 for YelpChi, .921 for YelpNYC, and .948 for YelpZIP in F1 score while the models with GloVe only embeddings achieved .600 for YelpChi, .629 for YelpNYC, and .629 for YelpZIP. For the Word2Vec experiments, the models with Word2Vec+Reviewer (size=100) embeddings achieved .900 for YelpChi, .924 for YelpNYC, and .939 for YelpZIP in F1 score while the models with Word2Vec only embeddings achieved .650 for YelpChi, .661 for YelpNYC, and .679 for YelpZIP. As shown in Figures 4.9 and 4.10, using hybrid embeddings as features is constantly more effective than using text embeddings only, even if the embedding size is only 3. Moreover, larger reviewer embedding sizes tend to have better performances.

Based on our findings, we have partially answered the second research question by showing that hybrid embedding outperformed a review's text embedding on fake review classification. However, we wonder whether the performance enhancement really comes from the additional characteristics preserved by the hybrid embeddings, or whether it is simply because there are more features in the hybrid embeddings. To answer this question, in the next experiment, we conducted a feature ablation study to analyze each feature used to build reviewer embeddings.

Table 4.8. *Precision (GloVe-only vs. GloVe+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.667	0.611	0.616
glove+reviewer (emb_size=3)	0.636	0.722	0.675
glove+reviewer (emb_size=10)	0.667	0.759	0.749
glove+reviewer (emb_size=20)	0.789	0.827	0.794
glove+reviewer (emb_size=50)	0.952	0.907	0.913
glove+reviewer (emb_size=100)	1.000	0.937	0.962

Table 4.9. *Recall (GloVe-only vs. GloVe+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.545	0.648	0.642
glove+reviewer (emb_size=3)	0.636	0.756	0.744
glove+reviewer (emb_size=10)	0.636	0.848	0.802
glove+reviewer (emb_size=20)	0.682	0.875	0.866
glove+reviewer (emb_size=50)	0.909	0.920	0.927
glove+reviewer (emb_size=100)	0.909	0.906	0.934

Table 4.10. *F1 scores (GloVe-only vs. GloVe+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.600	0.629	0.629
glove+reviewer (emb_size=3)	0.636	0.739	0.708
glove+reviewer (emb_size=10)	0.651	0.801	0.775
glove+reviewer (emb_size=20)	0.732	0.851	0.828
glove+reviewer (emb_size=50)	0.930	0.913	0.920
glove+reviewer (emb_size=100)	**0.952**	**0.921**	**0.948**

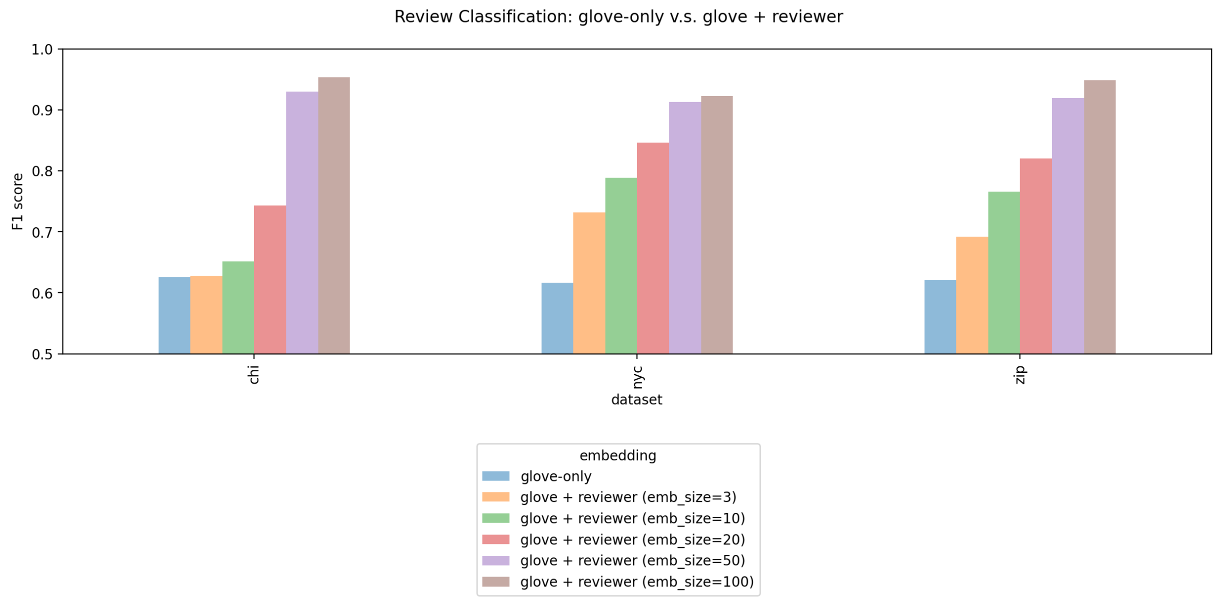


Figure 4.9. F1 scores (GloVe-only vs. GloVe+Reviewer)

Table 4.11. *Precision (Word2Vec-only vs. Word2Vec+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.722	0.655	0.657
word2vec+reviewer (emb_size=3)	0.800	0.736	0.712
word2vec+reviewer (emb_size=10)	0.750	0.761	0.755
word2vec+reviewer (emb_size=20)	0.833	0.793	0.799
word2vec+reviewer (emb_size=50)	0.905	0.888	0.887
word2vec+reviewer (emb_size=100)	1.000	0.942	0.942

Table 4.12. *Recall (Word2Vec-only vs. Word2Vec+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.591	0.668	0.702
word2vec+reviewer (emb_size=3)	0.727	0.751	0.771
word2vec+reviewer (emb_size=10)	0.682	0.820	0.802
word2vec+reviewer (emb_size=20)	0.682	0.848	0.851
word2vec+reviewer (emb_size=50)	0.864	0.925	0.921
word2vec+reviewer (emb_size=100)	0.818	0.906	0.936

4.4.2 Feature ablation

This ablation study helps us understand the contribution of each feature to the overall model. In this experiment, we removed one of the eight features each time, rebuilt the reviewer

Table 4.13. *F1 scores (Word2Vec-only vs. Word2Vec+Reviewer)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.650	0.661	0.679
word2vec+reviewer (emb_size=3)	0.762	0.743	0.740
word2vec+reviewer (emb_size=10)	0.714	0.789	0.778
word2vec+reviewer (emb_size=20)	0.750	0.819	0.824
word2vec+reviewer (emb_size=50)	0.884	0.906	0.904
word2vec+reviewer (emb_size=100)	**0.900**	**0.924**	**0.939**

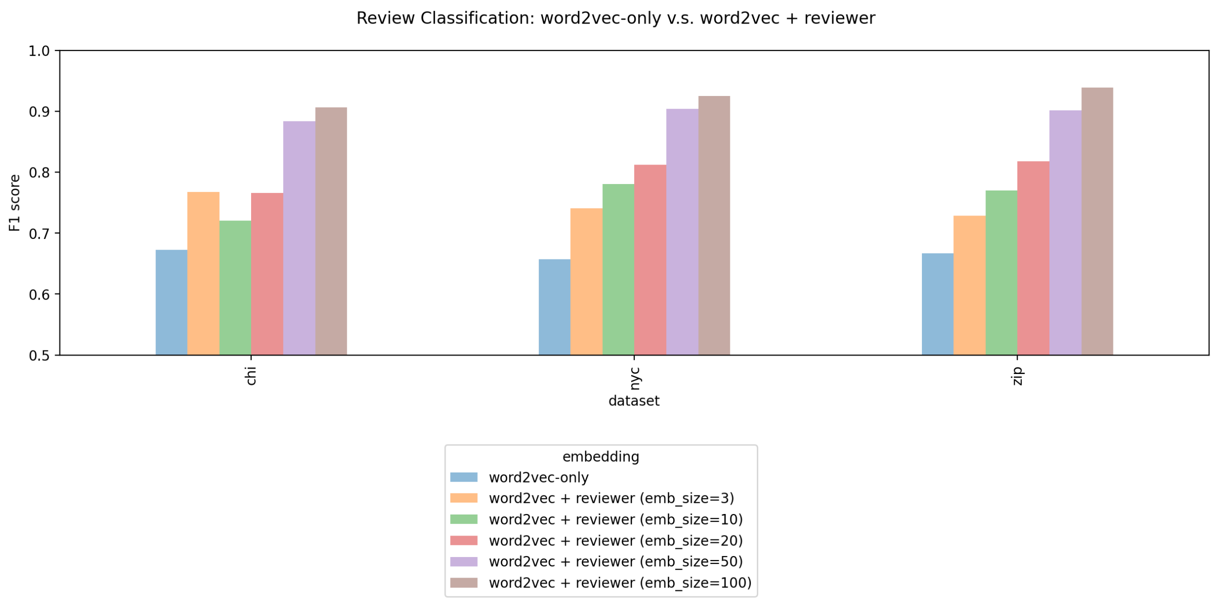


Figure 4.10. F1 scores (Word2Vec-only vs. Word2Vec+Reviewer)

network, generated reviewer embeddings based on the rest of the seven features, and investigated the impact of feature removal on the performance of fake review classification. Tables 4.14 and 4.15 are the results of the feature ablation for YelpChi regarding the GloVe+Reviewer features (reviewer embedding size=100) and the Word2Vec+Reviewer features (reviewer embedding size=100). Tables 4.16 and 4.17 are the results of the feature ablation for YelpNYC, and Tables 4.18 and 4.19 are the results of the feature ablation for YelpZIP.

It is interesting when some features were dropped, the hybrid embeddings built with the remaining features outperformed the hybrid embeddings built with all of the features. The results answer our previous question: does the performance enhancement really come from the

Table 4.14. *YelpChi (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.849	0.925	0.886
PR	0.836	0.910	0.871
NR	0.744	0.866	0.800
avgRD	0.790	0.955	0.865
ERD	0.827	0.925	0.873
RL	0.772	0.910	0.836
ACS	0.817	0.866	0.841
MCS	0.894	0.881	0.887
None	1.000	0.909	0.952

Table 4.15. *YelpChi (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.765	0.925	0.838
PR	0.818	0.940	0.875
NR	0.791	0.791	0.791
avgRD	0.778	0.940	0.851
ERD	0.877	0.955	0.914
RL	0.772	0.910	0.836
ACS	0.819	0.881	0.849
MCS	0.896	0.896	0.896
None	1.000	0.818	0.900

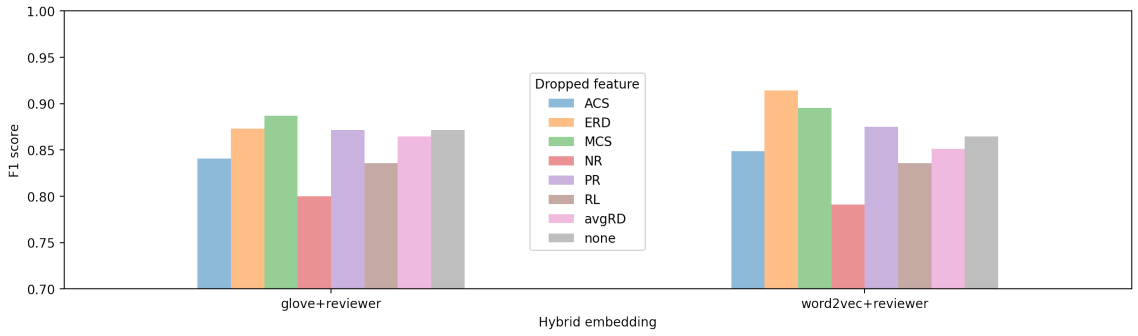


Figure 4.11. F1 scores - individual features (YelpChi)

additional characteristics preserved by the hybrid embeddings, or is it simply because there are more features in the hybrid embeddings? Based on our findings, using more features to build

Table 4.16. *YelpNYC (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.906	0.936	0.921
PR	0.916	0.934	0.925
NR	0.932	0.938	0.935
avgRD	0.925	0.934	0.929
ERD	0.902	0.927	0.914
RL	0.910	0.932	0.921
ACS	0.915	0.928	0.921
MCS	0.907	0.940	0.924
None	0.937	0.906	0.921

Table 4.17. *YelpNYC (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.895	0.946	0.920
PR	0.897	0.928	0.912
NR	0.921	0.939	0.930
avgRD	0.906	0.924	0.915
ERD	0.887	0.929	0.908
RL	0.890	0.939	0.914
ACS	0.907	0.939	0.923
MCS	0.903	0.934	0.918
None	0.942	0.906	0.924

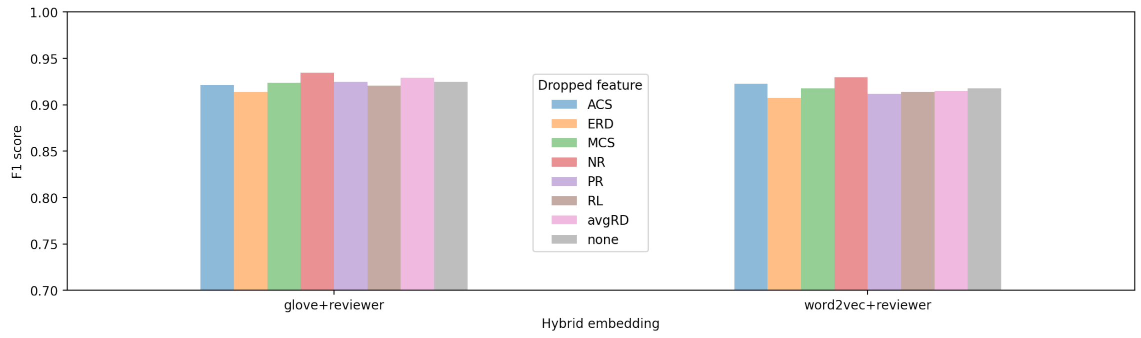


Figure 4.12. F1 scores - individual features (YelpNYC)

hybrid embeddings does not guarantee better performance. On the contrary, including more features actually made the performance worse in some cases. It is because we did not perform

Table 4.18. *YelpZIP (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.897	0.917	0.907
PR	0.932	0.927	0.929
NR	0.920	0.926	0.923
avgRD	0.934	0.908	0.921
ERD	0.915	0.901	0.908
RL	0.911	0.919	0.915
ACS	0.926	0.914	0.920
MCS	0.926	0.919	0.923
None	0.962	0.934	0.948

Table 4.19. *YelpZIP (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
MNR	0.859	0.914	0.885
PR	0.904	0.923	0.913
NR	0.888	0.926	0.907
avgRD	0.913	0.907	0.910
ERD	0.893	0.903	0.898
RL	0.869	0.918	0.893
ACS	0.896	0.913	0.904
MCS	0.895	0.914	0.904
None	0.942	0.936	0.939

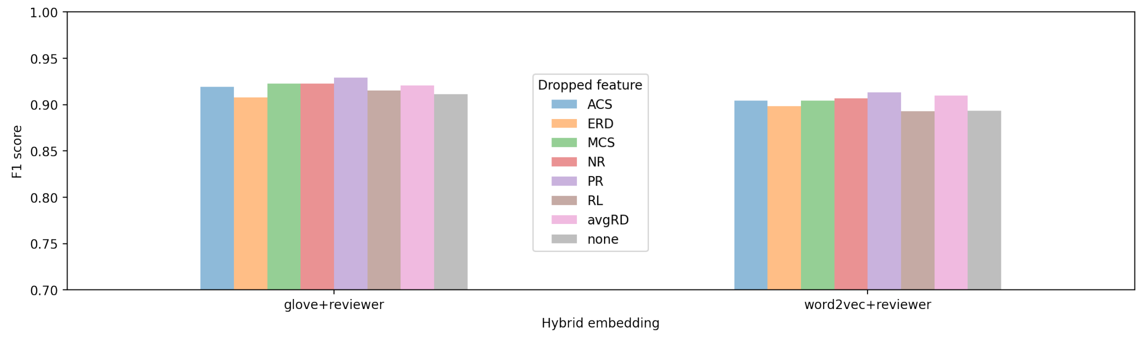


Figure 4.13. F1 scores - individual features (YelpZIP)

feature selection before adopting these eight features. Therefore, dropping certain features might lead to a better performance than using all of the features. The results imply that conducting feature selection for the target dataset and the target type of embedding (e.g. GloVe+Reviewer, Word2Vec+Reviewer) could enhance the performance of hybrid embeddings.

Another finding is: the precision scores for the YelpChi dataset are very high, sometimes even reaching 1.000. It might be because the YelpChi dataset is relatively smaller in size (as described in Table 3.1) compared to YelpNYC and YelpZIP. Ying (2019) discussed that smaller datasets are prone to cause overfitting and achieve a nearly perfect result. Also, the results of YelpChi are inconsistent with the other two datasets in many experiments.

Unfortunately, these results do not suggest any of the eight individual features constantly contribute more than others. The performance of fake review classification varied from different datasets. Thus, we further explored which feature groups constantly contribute more to the overall model in the next sections. In particular, we examined the effect of behavioral and textual features.

4.4.2.1 Behavioral vs. textual

In this section, we compared the dropout experiments of two feature groups: behavioral features (i.e., MNR, PR, NR, avgRD, ERD) and textual features (i.e., RL, ACS, MCS). We conducted fake review classification with the two groups respectively. The experiment helps us understand the performance difference between dropping reviewer behavioral features and dropping reviewer textual features. Tables 4.20, 4.21, and 4.22 are the results for the GloVe+Reviewer features (reviewer embedding size=100). Tables 4.23, 4.24, and 4.25 are the results for the Word2Vec+Reviewer features (reviewer embedding size=100).

Table 4.20. *YelpChi (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.826	0.851	0.838
Textual	0.824	0.910	0.865

As illustrated in Figure 4.14 and 4.15, we found that removing behavioral features constantly caused more of a performance drop than removing textual features for all three datasets and for both GloVe+Reviewer and Word2Vec+Reviewer embeddings. This shows a reviewer's behavioral features contribute more to fake review classification than textual features.

Table 4.21. *YelpNYC (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.857	0.927	0.890
Textual	0.914	0.924	0.919

Table 4.22. *YelpZIP (GloVe+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.862	0.880	0.871
Textual	0.928	0.918	0.923

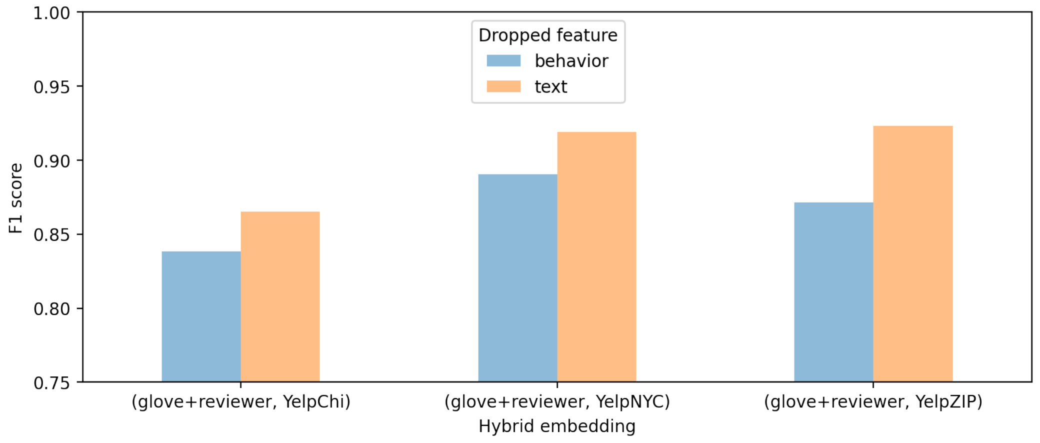


Figure 4.14. F1 scores - behavioral and textual features (GloVe+Reviewer)

Table 4.23. *YelpChi (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.771	0.806	0.788
Textual	0.836	0.910	0.871

Table 4.24. *YelpNYC (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.851	0.936	0.892
Textual	0.917	0.927	0.922

Table 4.25. *YelpZIP (Word2Vec+Reviewer)*

Dropped feature	Precision	Recall	F1 score
Behavioral	0.827	0.893	0.858
Textual	0.893	0.921	0.907

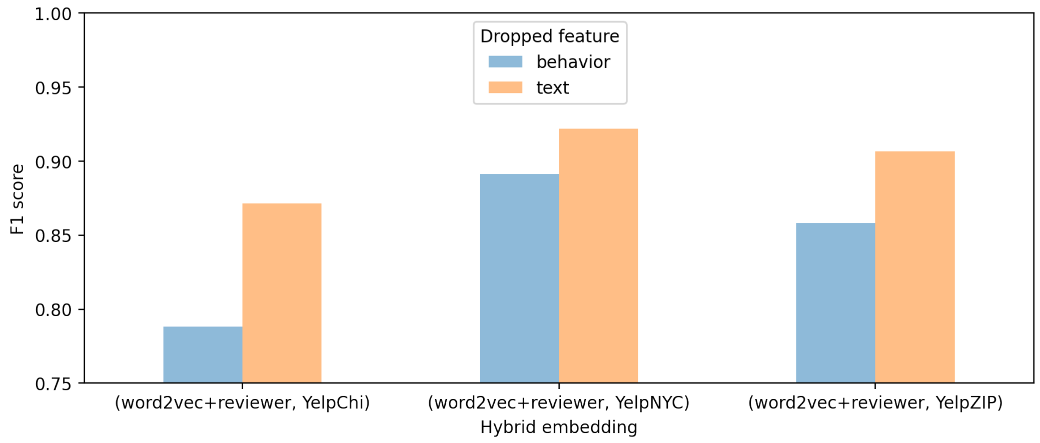


Figure 4.15. F1 scores - behavioral and textual features (Word2Vec+Reviewer)

Our findings are in agreement with those in the studies of Rayana and Akoglu (2015) and Mukherjee, Venkataraman, Liu, and Glance (2013). These two studies suggested reviewer behavioral features played a much more critical role to the overall than reviewer textual features did. The agreement indicates that Yelp's filter might rely more on a behavioral-based approach rather than a textual-based one.

To further understand which features/feature groups contribute more to the overall model, we investigate which feature combinations are more critical in textual features and behavioral features respectively. One method for exploring the eight features is to perform an exhaustive comparison for all combinations. However, it is computationally expensive and does not actually bring us an explainable and conclusive aspect. Thus, we chose feature groups based on the commonality of features. In particular, for textual features, we examine the effect of content similarity (represented by ACS and MCS) and review length (RL). We combined ACS and MCS as one group since they are both related to content similarity. ACS captures unusual reviewers who have different writing patterns from others regarding average content similarity, while MCS addresses unusual reviewers who copy and paste many reviews. On the other hand, for behavioral features, we examine the effect of rating score features (represented by PR and NR) and rating deviation features (represented by avgRD and ERD). Considering PR and NR simultaneously can better capture reviewers who have abnormal rating scores (e.g., reviewers who are recruited to provide only 5 stars to overpraise certain products or those who always provide 1 star to attack business competitors). With regard to rating deviation, avgRD is an external measure that computes the rating deviation between a reviewer and others, while ERD is an internal measure that evaluates if a reviewer's own rating distribution is abnormal. The two aspects help us understand whether a reviewer has an unusual rating distribution. Besides, since MNR has been discussed in the previous experiments and did not show a constant performance pattern, we do not discuss MNR with other behavioral features here.

4.4.2.2 Textual features: content similarity vs. review length

In this section, we investigated the contribution of two different groups of textual features: content similarity features (i.e., ACS, MCS) and the review length (i.e., RL). We conducted fake review classification with the two groups respectively. Tables 4.26, 4.27 and 4.28 are the precision, recall, and F1 scores for GloVe+Reviewer features (reviewer embedding size=100). Tables 4.29, 4.30 and 4.31 are for Word2Vec+Reviewer features (reviewer embedding size=100).

Table 4.26. *Precision: content similarity vs. review length (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.833	0.911	0.921
Review length (RL)	0.772	0.910	0.911

Table 4.27. *Recall: content similarity vs. review length (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.896	0.940	0.923
Review length (RL)	0.910	0.932	0.919

According to the results of GloVe+Reviewer shown in Figure 4.16, removing content similarity features caused a less significant performance drop among all three Yelp datasets than

Table 4.28. *F1 scores: content similarity vs. review length (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.863	0.925	0.922
Review length (RL)	0.836	0.921	0.915

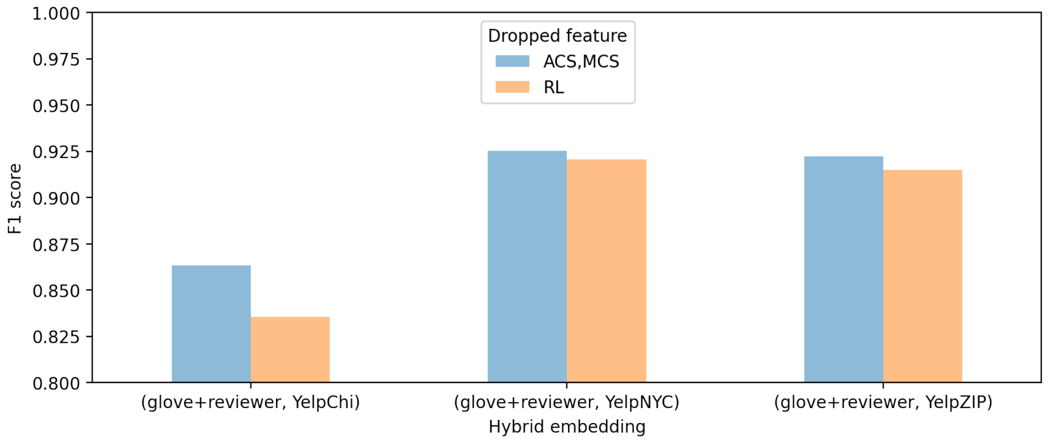


Figure 4.16. F1 scores: content similarity vs. review length (GloVe+Reviewer)

Table 4.29. *Precision: content similarity vs. review length (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.784	0.902	0.881
Review length (RL)	0.772	0.890	0.869

Table 4.30. *Recall: content similarity vs. review length (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.866	0.944	0.925
Review length (RL)	0.910	0.939	0.918

Table 4.31. *F1 scores: content similarity vs. review length (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Content similarity (ACS,MCS)	0.823	0.923	0.902
Review length (RL)	0.836	0.914	0.893

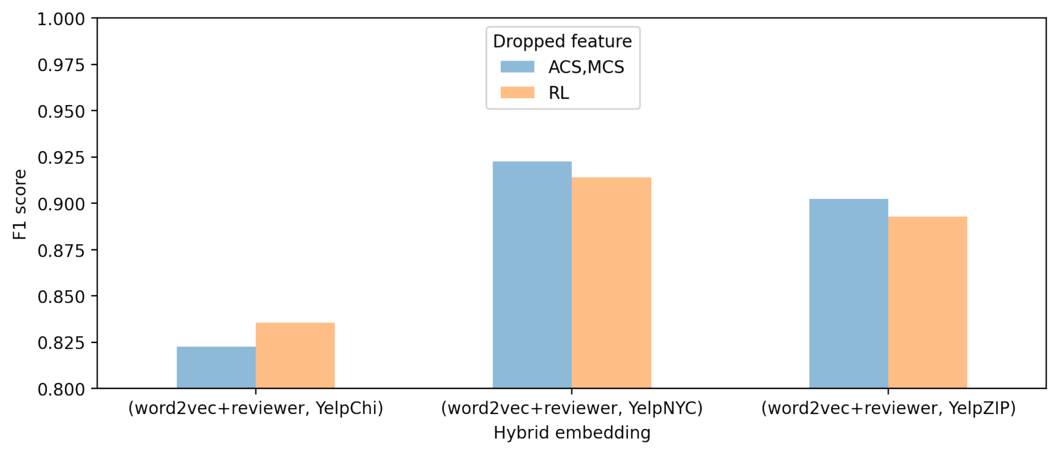


Figure 4.17. F1 scores: content similarity vs. review length (Word2Vec+Reviewer)

removing review length feature. With regard to Word2Vec+Reviewer in Figure 4.17, the trend of results is similar for YelpNYC and YelpZIP. However, it's different for YelpChi. As we discussed in Section 4.4.2, the inconstant performance of YelpChi could be caused by a smaller dataset size. Except for YelpChi, the results implied that review length played a more important role to the overall model than content similarity features did.

It is interesting that review length (RL) seems to be a more important factor than the bigram-based content similarity features (ACS, MCS). These findings are in agreement with those in the study of Mukherjee, Venkataraman, Liu, and Glance (2013), which found removing RL caused a larger performance drop than removing MCS. Our experiment extends the study of Mukherjee, Venkataraman, Liu, and Glance (2013) by adding ACS as a content similarity feature in the ablation study. MCS examines whether some posted reviews are similar to previous reviews, while ACS measures to what degree a reviewer's content similarity should be considered as abnormal. By considering ACS measure in addition to others, our study is more comprehensive than those of Mukherjee, Venkataraman, Liu, and Glance (2013).

Another finding is that removing one (RL) feature caused a more significant performance drop than removing two (ACS, MCS) content similarity features. It supports our previous argument: including more features in hybrid embedding does not necessarily lead to a performance enhancement.

4.4.2.3 Behavioral features: rating score vs. rating deviation

In this section, we investigated the contribution of two different groups of behavioral features: rating score features (i.e., PR, NR) and rating deviation features (i.e., avgRD, ERD). Regarding rating score features, Mukherjee, Venkataraman, Liu, and Glance (2013) found that 15% of the spammers have less than 80% of their reviews as positive (4-5 stars). For rating deviation, Rayana and Akoglu (2015) discussed that reviewers with high avgRD and low ERD are more suspicious. We conducted fake review classification with the two feature groups respectively. Tables 4.32, 4.33 and 4.34 are the precision , recall, and F1 scores for GloVe+Reviewer features (reviewer embedding size=100). Tables 4.35, 4.36 and 4.37 are for Word2Vec+Reviewer features (reviewer embedding size=100).

Table 4.32. *Precision: rating score vs. rating deviation (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.833	0.931	0.924
Rating deviation (avgRD,ERD)	0.836	0.892	0.906

Table 4.33. *Recall: rating score vs. rating deviation (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.896	0.929	0.911
Rating deviation (avgRD,ERD)	0.910	0.924	0.904

Table 4.34. *F1 scores: rating score vs. rating deviation (GloVe+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.863	0.930	0.917
Rating deviation (avgRD,ERD)	0.871	0.908	0.905

According to the results of GloVe+Reviewer and Word2Vec+Reviewer shown in Figure 4.18 and 4.19, removing rating score features caused a less significant performance drop than removing rating deviation features did for YelpNYC and YelpZIP. However, the result for YelpChi on Word2Vec+Reviewer embedding shows a different trend from others, possibly due to the reasons previously outlined. In overall, this implies reviewer's rating deviation contributes

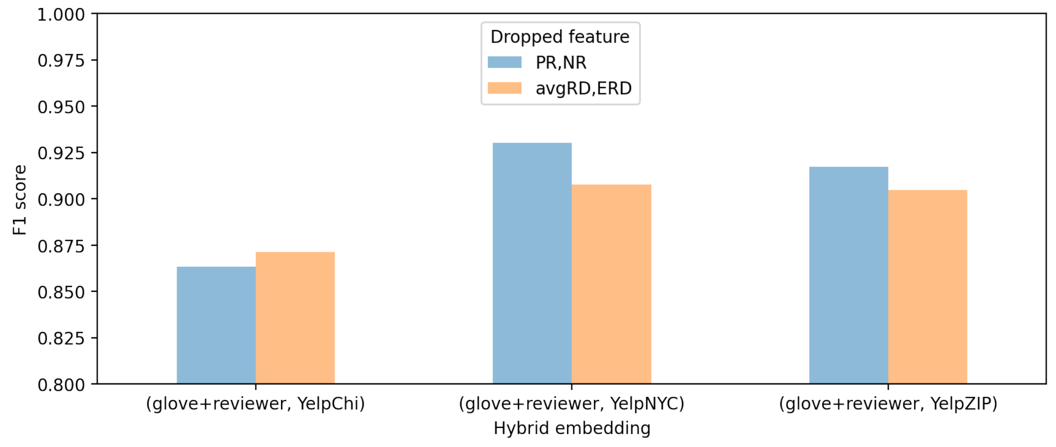


Figure 4.18. F1 scores: rating score vs. rating deviation (GloVe+Reviewer)

Table 4.35. *Precision: rating score vs. rating deviation (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.829	0.928	0.874
Rating deviation (avgRD,ERD)	0.826	0.889	0.879

Table 4.36. *Recall: rating score vs. rating deviation (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.866	0.936	0.908
Rating deviation (avgRD,ERD)	0.851	0.929	0.897

Table 4.37. *F1 scores: rating score vs. rating deviation (Word2Vec+Reviewer)*

Dropped feature	YelpChi	YelpNYC	YelpZIP
Rating score (PR,NR)	0.847	0.932	0.891
Rating deviation (avgRD,ERD)	0.838	0.909	0.888

more than reviewer's rating score to our model. Our findings are different from those in the study of Mukherjee, Venkataraman, Liu, and Glance (2013), which found rating score is more important than rating deviation. There are two possible reasons for the disagreement: 1) their study only considered PR, which captures spammers who are prone to give extreme positive ratings, but did not consider NR, which assumes spammers could abuse negative ratings for

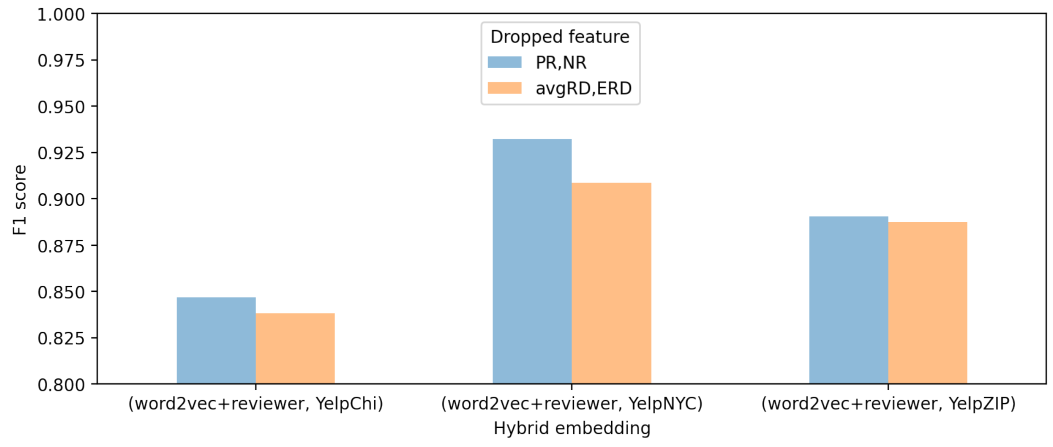


Figure 4.19. F1 scores: rating score vs. rating deviation (Word2Vec+Reviewer)

business competition (Jindal & Liu, 2007); 2) their study only considered rating deviation based on avgRD. Our study, instead, included both avgRD and ERD as the rating deviation features. avgRD is an external measure that computes the rating deviation between a reviewer and others, while ERD is an internal measure that evaluates if a reviewer's own rating distribution is abnormal. Although our conclusion is different from the conclusion of Mukherjee, Venkataraman, Liu, and Glance (2013), we consider ours are more comprehensive because we included PR+NR for rating score features and avgRD+ERD for rating deviation features.

4.4.2.4 Summary

In the last experiment, we partially answered the second research question by showing that hybrid embedding outperformed a review's text embedding in fake review classification. Next, in the feature ablation study, we showed the performance improvement with hybrid embeddings was not simply caused by including more features but was caused by effective reviewer features. In addition, there are three interesting findings from the feature ablation study: 1) reviewer behavioral features contribute more to the overall model of fake review classification than reviewer textual features; 2) regarding reviewer textual features, review length is more important than content similarity; 3) in terms of reviewer behavioral features, rating deviation features played a more important role than rating score features did.

We have answered the second research question via the above experiments. However, to further demonstrate the efficacy of hybrid embeddings in fake review identification, we conduct a comparison between hybrid embeddings and related works in the next section.

4.4.3 Comparison with related works

We compared the results of fake review classification using our hybrid embeddings with related works. The first baseline was proposed by Mukherjee, Venkataraman, Liu, Glance, et al. (2013). They analyzed the reviews of the Yelp datasets and proposed eight statistical behavioral features (e.g., the Activity Window, the Percentage of Positive Reviews). They applied SVM and Naïve Bayes on the behavioral and linguistic features and got the best performance with SVM.

The second baseline was presented by X. Wang et al. (2016). They collected relations between any two entities (reviewers and products) as global information, used the global information as features, and applied SVM to perform review classification experiments.

The third was introduced by X. Wang, Liu, and Zhao (2017b). They proposed a neural network model to detect review spam for the cold-start problem. The model learned embeddings for a review, a review's rating, and a product's average rating respectively. They followed the idea of TransE (Bordes et al., 2013), which encodes the graph structure and represents the nodes and edges (head, relation, tail) in low dimension vector space and treats products as the head part of TransE, reviewers as the relation part, and reviews as the tail part. The authors then used the learned embeddings as features and trained SVM classifiers for fake reviewer classification.

The fourth baseline was presented by X. Wang, Liu, and Zhao (2017a). The researchers proposed an attention-based neural network to detect deceptive review spam by using linguistic and behavioral features. They considered several effective behavioral features proposed by Mukherjee, Venkataraman, Liu, and Glance (2013), Rayana and Akoglu (2015), and X. Wang et al. (2016) as the inputs of the Multi-Layer Perceptron (MLP) hidden layer and took the output vectors as behavioral feature vectors. In addition, they trained a bigram word embedding model as the inputs of a CNN and took the output vectors as linguistic feature vectors.

We used the F1 scores reported in the study of X. Wang, Liu, and Zhao (2017a) as the baselines of our comparison. Since they reported F1 scores for hotel reviews and restaurant reviews separately, we only listed the best numbers of the related works. On the other hand, since our experiments were reported separately for the three Yelp datasets (i.e., YelpChi, YelpNYC, and YelpZIP), we also report the best F1 scores. The comparison of results is shown in Table 4.38. According to the comparison, the hybrid embeddings of Word2Vec+Reviewer and GloVe+Reviewer both outperformed the F1 scores of the baselines (.939 for Word2Vec+Reviewer, .952 for GloVe+Reviewer).

Table 4.38. *Fake review classification (comparison with related works)*

Methods	Precision	Recall	F1
Mukherjee, Venkataraman, Liu, Glance, et al. (2013)	0.845	0.878	0.861
X. Wang et al. (2016)	0.868	0.918	0.892
X. Wang, Liu, and Zhao (2017a)	0.894	0.930	0.912
X. Wang, Liu, and Zhao (2017b)	0.584	0.751	0.657
Hybrid embeddings (Word2Vec+Reviewer)	0.942	0.936	0.939
Hybrid embeddings (GloVe+Reviewer)	1.000	0.909	0.952

4.4.4 Summary

In the experiments, first, we partially answered our second research question by showing that hybrid embedding outperformed a review's text embedding in fake review classification. Second, in the feature ablation study, we further investigated each of the eight features used to build the hybrid embeddings. The result showed including more features does not increase the efficacy of hybrid embeddings in fake review classification, and it cleared the potential doubt of the first experiment: the reason that the hybrid embeddings performed better was simply that they included more features. Last but not least, we compared the hybrid embeddings with other related works that used the same Yelp datasets. The comparison showed the two types of hybrid embeddings (Word2Vec+Reviewer, GloVe+Reviewer) both outperformed related works. Based on the experiments, we have answered the second research question.

4.5 Reviewer network construction

In the methodology, we proposed a heuristic reviewer network construction, which is capable of building a network for nodes based on the feature values of nodes. However, does this approach actually build meaningful reviewer networks that can be used to generate effective reviewer embeddings? Unfortunately, there is no exact answer for this question because the reviewer networks were not provided in the Yelp datasets. Although we could not show whether our heuristic approach built perfect networks for the Yelp datasets, we conducted two experiments to examine whether the reviewer networks built by our approach were more effective at preserving the characteristics of spammers than the reviewer networks built randomly. The first experiment is to compare the clusters of reviewer embeddings built by our reviewer networks with the clusters of reviewer embeddings built by random networks. Similar to the clustering analysis in section 4.3, we compared the purity and NMI scores of the two sets of clusters. The results are shown in Table 4.39.

Table 4.39. *Clusters analysis (our reviewer networks vs. random networks)*

dataset/metrics	Purity (ours)	Purity (random)	NMI (ours)	NMI (random)
YelpChi	**0.674**	0.529	**0.114**	0.002
YelpNYC	**0.702**	0.580	**0.146**	0.021
YelpZIP	**0.741**	0.548	**0.189**	0.008

We compared the clusters of reviewer embeddings built by our reviewer networks with the clusters of reviewer embeddings built by random networks. Based on the results in Table 4.39, using our reviewer networks resulted in better purity scores and NMI scores than using random networks. The results suggest that both Purity and NMI are enhanced when using the reviewer networks built by our approach rather than the random networks.

In the second experiment, we conducted fake review classification and compared reviewer networks built by our approach and reviewer networks built randomly. Tables 4.40, 4.41, and 4.42 show the results of precision, recall, and F1 scores.

According to the results in Table 4.42, there is a consistent performance improvement when using networks built by heuristic network construction to generate reviewer embeddings

Table 4.40. *Precision (heuristic vs. random)*

Embedding	Dataset	Precision (heuristic)	Precision (random)
GloVe+Reviewer	YelpChi	1.000	0.750
GloVe+Reviewer	YelpNYC	0.937	0.813
GloVe+Reviewer	YelpZIP	0.962	0.864
Word2Vec+Reviewer	YelpChi	1.000	0.786
Word2Vec+Reviewer	YelpNYC	0.942	0.791
Word2Vec+Reviewer	YelpZIP	0.942	0.806

Table 4.41. *Recall (heuristic vs. random)*

Embedding	Dataset	Recall (heuristic)	Recall (random)
GloVe+Reviewer	YelpChi	0.909	0.545
GloVe+Reviewer	YelpNYC	0.906	0.936
GloVe+Reviewer	YelpZIP	0.934	0.919
Word2Vec+Reviewer	YelpChi	0.818	0.500
Word2Vec+Reviewer	YelpNYC	0.906	0.931
Word2Vec+Reviewer	YelpZIP	0.936	0.908

Table 4.42. *F1 scores (heuristic vs. random)*

Embedding	Dataset	F1 (heuristic)	F1 (random)
GloVe+Reviewer	YelpChi	**0.952**	0.632
GloVe+Reviewer	YelpNYC	**0.921**	0.870
GloVe+Reviewer	YelpZIP	**0.948**	0.891
Word2Vec+Reviewer	YelpChi	**0.900**	0.611
Word2Vec+Reviewer	YelpNYC	**0.924**	0.855
Word2Vec+Reviewer	YelpZIP	**0.939**	0.854

rather than using networks built randomly. For the Glove+Reviewer features, our approach outperformed randomly built networks by .320 in F1 for YelpChi, .051 in F1 for YelpNYC, .057 in F1 for YelpZIP; for the Word2Vec+Reviewer features, our approach outperformed random by .289 in F1 for YelpChi, .069 in F1 for YelpNYC, and .085 in F1 for YelpZIP.

4.5.1 Summary

By comparing our approach to randomly-created networks, the results show the effectiveness of reviewer networks generated by heuristic network construction. Although our heuristic network construction is not perfect, it helps us build meaningful reviewer networks that are significantly better than random networks. The results also imply that a better approach to reviewer network construction could largely improve the performance of fake review classification. In the future, it is worth exploring more efficient approaches.

4.6 Spammer Threshold

The Yelp datasets do not contain spammer labels. Several studies used the same Yelp datasets as we used in this study, but to the best of our knowledge, we did not find any that discussed the definition of spammer. Jindal and Liu (2008) defined spammers as users with at least one filtered review. However, this could be a careless definition. Firstly, users who wrote dozens of unfiltered reviews were treated as spammers just because they wrote one filtered review. Secondly, the reviewer embeddings learned under this rough spammer definition might not capture the spammer's characteristics well. Thus, we wanted to examine whether setting the spammer threshold at a higher level (i.e., we set a stricter criterion for defining a user as a spammer) impacted the learned multi-view reviewer embeddings when they were used in review classification.

The experimental setup was similar to previous review classification experiments. We tested different spammer thresholds [”$>=1$”, ”$>=2$”, ”$>=3$”, ”$>=4$”], where ”$>=X$” indicates the user is a spammer when they wrote at least X filtered reviews. We compared the hybrid embeddings with different spammer thresholds. The results are shown in tables 4.43 and 4.44.

For YelpChi and YelpZIP, the best models of GloVe+Reviewer embeddings achieved .952 and .948 in F1 score when the number of filtered reviews was greater than or equal to 4. For YelpNYC, the best model achieved .929 when the number of filtered reviews was greater than or equal to 3.

Table 4.43. *F1 scores of GloVe+Reviewer (with different spammer threshold)*

Dataset / Filtered Review	>=1	>=2	>=3	>=4
YelpChi	0.754	0.839	0.865	**0.952**
YelpNYC	0.775	0.876	**0.929**	0.921
YelpZIP	0.796	0.884	0.908	**0.948**

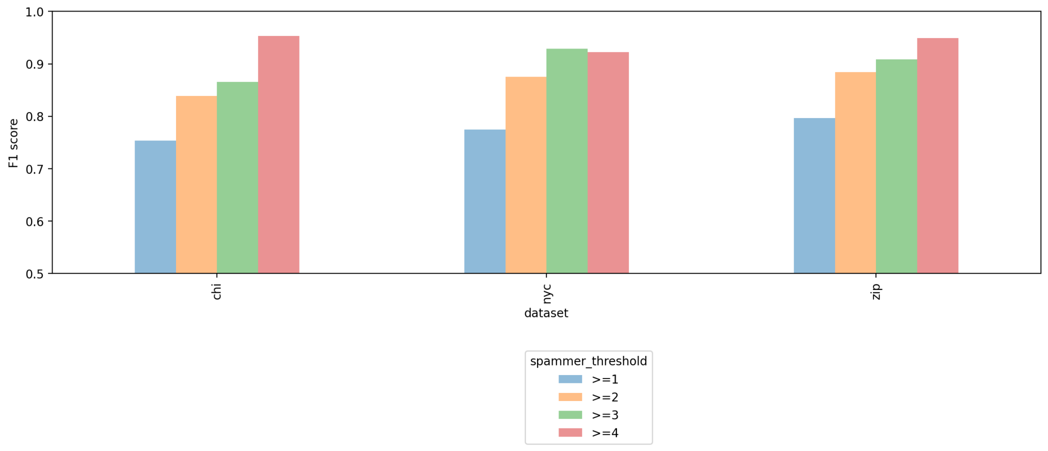


Figure 4.20. Fake review classification GloVe+Reviewer (with different spammer threshold)

Table 4.44. *F1 scores of Word2Vec+Reviewer (with different spammer threshold)*

Dataset / Filtered Review	>=1	>=2	>=3	>=4
YelpChi	0.748	0.784	0.826	**0.900**
YelpNYC	0.775	0.860	**0.927**	0.924
YelpZIP	0.784	0.862	0.891	**0.939**

For YelpChi and YelpZIP, the best models of Word2Vec+Reviewer embeddings achieved .900 and .939 in F1 score when the number of filtered reviews was greater than or equal to 4. For YelpNYC, the best model achieved .927 when the number of filtered reviews was greater than or equal to 3.

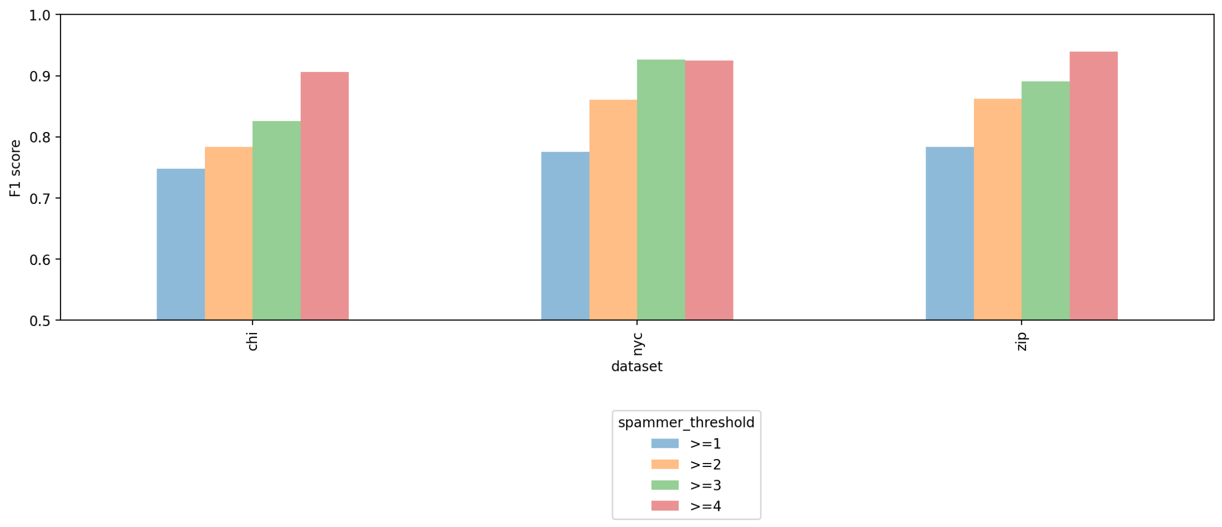


Figure 4.21. Fake review classification Word2Vec+Reviewer
(with different spammer threshold)

4.6.1 Summary

According to the results, using different spammer thresholds for different datasets could significantly improve the performance of review classification. Overall, when the number of filtered reviews was greater than or equal to 4, we had better performance for the three Yelp datasets. Our findings also imply that by carefully choosing the spammer thresholds for the datasets, the learned reviewer embeddings will be more effective at fake review classification.

CHAPTER 5. SUMMARY AND FUTURE WORKS

In this study, we identified and tackled two research gaps: First, previous spammer identification studies that adopted network approaches have not been able to effectively preserve the relationship between reviewers. Second, in previous literature, the text embedding of a review has not been jointly considered with its reviewer's embedding. We proposed and answered two major research questions:

1) *Do reviewer embeddings learn via multi-view network representation approaches preserve the characteristics of spammers?* The research question addresses the first research gap. In order to tackle it, we adopted MVE (Qu et al., 2017), a multi-view node representation learning model, because of its strength in learning reviewer embeddings in consideration of second-order proximity and multiple proximities. To evaluate whether the reviewer embeddings were effective as features of spammer classification, we conducted spammer classification experiments and used the reviewer embeddings as features. In addition, we applied a clustering algorithm to group reviewer embeddings into two clusters and compared the difference between the distribution of clusters and the distribution of spammer labels. Our results showed that generating clusters with the reviewer embeddings capture the difference between spammers and non-spammers better than with the reviewers' features directly. In addition, the two-dimensional PCA and t-SNE visualization plots of reviewer embeddings depicted a clear separation between spammers and non-spammers.

2) *Is the hybrid embedding of a review more sensitive at distinguishing whether a review is fake than the review's text embedding?* This question addresses the second research gap. To fill this gap, we built hybrid embeddings, which consist of each review's text embeddings and its reviewer's embedding. Next, we performed fake review classification experiments and evaluated whether using hybrid embeddings as features was more effective than using text embeddings as features. Our results are promising and suggest that hybrid embeddings are more effective than text-only embeddings on fake review identification. Using hybrid embeddings as features resulted in a performance increase of .3 in F1 score compared to using only text embeddings. Moreover, we conducted a feature ablation study to examine each of the features used to build reviewer embeddings. In the study, we showed the performance improvement of hybrid embeddings was

not simply caused by including more features but rather was caused by effective reviewer features. Furthermore, we compared the results of fake review classification using hybrid embeddings with the results reported by other related studies that also evaluated their methods on the Yelp datasets, namely, Mukherjee, Venkataraman, Liu, Glance, et al. (2013), X. Wang et al. (2016), and X. Wang, Liu, and Zhao (2017a). The results showed the hybrid embeddings of Word2Vec+Reviewer and GloVe+Reviewer both outperformed all other baselines.

Furthermore, we validated the heuristic reviewer network construction proposed in this study. By comparing our approach to randomly-created networks, we showed our heuristic network construction built meaningful reviewer networks that are significantly better than random networks. The results also imply that a better approach to reviewer network construction could largely improve the performance of fake review classification.

Moreover, we challenged the definition of spammer used in previous literature. We conducted fake review classification experiments and compared different spammer thresholds. Our results showed when the spammer threshold is greater than or equal to 4, the learned reviewer embeddings have better performance compared to the baseline. Our findings show that by carefully choosing the spammer thresholds for the datasets, the learned reviewer embeddings may be more effective at fake review classification.

As with any study, there are limitations and improvements that could be made. The following are recommendations for future research on this topic.

1. Transfer learning has gained much research attention and was adopted for sharing the learned models between homogeneous and heterogeneous data. An interesting direction would be to use transfer learning to share the learned models of one dataset with other homogeneous ones. Since constructing text and reviewer embeddings is computationally expensive, it's worth exploring whether the learned models are transferable between different datasets, which could tremendously reduce the computational cost. Furthermore, it is valuable to investigate whether the learned models from the Yelp datasets can be extended to other unlabeled heterogeneous datasets via transfer learning.

2. Although we discussed the reasons why using contextual text embedding rather than static text embedding was not required for this study, it would be interesting to extend this study

and explore the collaboration between contextual text embeddings and multi-view reviewer embeddings.

REFERENCES

Aghakhani, H., Machiry, A., Nilizadeh, S., Kruegel, C., & Vigna, G. (2018). Detecting deceptive reviews using generative adversarial networks. *arXiv preprint arXiv:1805.10364*.

Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., & Smola, A. J. (2013). Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on world wide web* (pp. 37–48).

Belkin, M., & Niyogi, P. (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems* (pp. 585–591).

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, *15*(6), 1373–1396.

Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, *7*(Nov), 2399–2434.

Bhatnagar, V., & Ahuja, S. (2010). Robust clustering using discriminant analysis. In *Industrial conference on data mining* (pp. 143–157).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems* (pp. 2787–2795).

Burgoon, J. K., Blair, J. P., Qin, T., & Nunamaker, J. F. (2003). Detecting deception through linguistic analysis. In *International conference on intelligence and security informatics* (pp. 91–101).

Cagnina, L., & Rosso, P. (2015). Classification of deceptive opinions using a low dimensionality representation. In *Proceedings of the 6th workshop on computational approaches to subjectivity, sentiment and social media analysis* (pp. 58–66).

Cao, S., Lu, W., & Xu, Q. (2015). Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th acm international on conference on information and knowledge management* (pp. 891–900).

Chang, S., Han, W., Tang, J., Qi, G.-J., Aggarwal, C. C., & Huang, T. S. (2015). Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining* (pp. 119–128).

Chua T, T. J., et al. (2009). Nus-wide: A real world web image database from national university of singapore. *Proceedings of the ACM International Conference on Image and Video Retrieval. Santorini Island, Greece*, *8*, 10.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dillon, M. (1983). *Introduction to modern information retrieval: G. salton and m. mcgill. mcgraw-hill, new york (1983). xv+ 448 pp., isbn 0-07-054484-0.* Pergamon.

Dong, M., Yao, L., Wang, X., Benatallah, B., Huang, C., & Ning, X. (2018). Opinion fraud detection via neural autoencoder decision forest. *arXiv preprint arXiv:1805.03379*.

Douglas, B. (2010). *Longman student grammar of spoken and written english.* Pearson Education India.

Dumais, S. T. (2004). Latent semantic analysis. *Annual review of information science and technology*, *38*(1), 188–230.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, *88*(2), 303–338.

Fei, G., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., & Ghosh, R. (2013). Exploiting burstiness in reviews for review spammer detection. *Icwsm*, *13*, 175–184.

Feng, S., Banerjee, R., & Choi, Y. (2012). Syntactic stylometry for deception detection. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2* (pp. 171–175).

Feng, S., Xing, L., Gogar, A., & Choi, Y. (2012). Distributional footprints of deceptive product reviews. In *Sixth international aaai conference on weblogs and social media.*

Feng, V. W., & Hirst, G. (2013). Detecting deceptive opinions with profile compatibility. In *Proceedings of the sixth international joint conference on natural language processing* (pp. 338–346).

FTC. (2009). Guides concerning the use of endorsements and testimonials in advertising. *Commission, FT (Ed. 16 CFR Part 255.*

Fusilier, D. H., Montes-y Gómez, M., Rosso, P., & Cabrera, R. G. (2015a). Detecting positive and negative deceptive opinions using pu-learning. *Information processing & management*, *51*(4), 433–443.

Fusilier, D. H., Montes-y Gómez, M., Rosso, P., & Cabrera, R. G. (2015b). Detection of opinion spam with character n-grams. In *International conference on intelligent text processing and computational linguistics* (pp. 285–294).

Getoor, L. (2005). Link-based classification. In *Advanced methods for knowledge discovery from complex data* (pp. 189–207). Springer.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 855–864).

Gyongyi, Z., & Garcia-Molina, H. (2005). Web spam taxonomy. In *First international workshop on adversarial information retrieval on the web (airweb 2005).*

Hancock, J. T., Curry, L. E., Goorha, S., & Woodworth, M. (2007). On lying and being lied to: A linguistic analysis of deception in computer-mediated communication. *Discourse Processes*, *45*(1), 1–23.

He, R., & McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web* (pp. 507–517).

Hernández, D., Guzmán, R., y Gomez, M. M., & Rosso, P. (2013). Using pu-learning to detect deceptive opinion spam. In *Proceedings of the 4th workshop on computational approaches to subjectivity, sentiment and social media analysis* (pp. 38–45).

Huang, F., Zhang, X., Li, C., Li, Z., He, Y., & Zhao, Z. (2018). Multimodal network embedding via attention based multi-view variational autoencoder. In *Proceedings of the 2018 acm on international conference on multimedia retrieval* (pp. 108–116).

Huang, H.-H., Wen, Y.-W., & Chen, H.-H. (2017). Detection of false online advertisements with dcnn. In *Proceedings of the 26th international conference on world wide web companion* (pp. 795–796).

Huiskes, M. J., & Lew, M. S. (2008). The mir flickr retrieval evaluation. In *Proceedings of the 1st acm international conference on multimedia information retrieval* (pp. 39–43).

Hwang, T., Atluri, G., Xie, M., Dey, S., Hong, C., Kumar, V., & Kuang, R. (2012). Co-clustering phenome–genome for phenotype classification and disease gene discovery. *Nucleic acids research*, *40*(19), e146–e146.

Jindal, N., & Liu, B. (2007). Analyzing and detecting review spam. In *Seventh ieee international conference on data mining (icdm 2007)* (pp. 547–552).

Jindal, N., & Liu, B. (2008). Opinion spam and analysis. In *Proceedings of the 2008 international conference on web search and data mining* (pp. 219–230).

Jindal, N., Liu, B., & Lim, E.-P. (2010). Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th acm international conference on information and knowledge management* (pp. 1549–1552).

Joachims, T. (1998). *Making large-scale svm learning practical* (Tech. Rep.). Technical report, SFB 475: Komplexitätsreduktion in Multivariaten ....

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the eighth acm sigkdd international conference on knowledge discovery and data mining* (pp. 133–142).

Juola, P. (2008). *Authorship attribution* (Vol. 3). Now Publishers Inc.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kong, X., Yu, P. S., Ding, Y., & Wild, D. J. (2012). Meta path-based collective classification in heterogeneous information networks. In *Proceedings of the 21st acm international conference on information and knowledge management* (pp. 1567–1571).

Koppel, M., Schler, J., & Argamon, S. (2009). Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology*, *60*(1), 9–26.

Kumar, A., Rai, P., & Daume, H. (2011). Co-regularized multi-view spectral clustering. In *Advances in neural information processing systems* (pp. 1413–1421).

Lai, C., Xu, K., Lau, R. Y., Li, Y., & Jing, L. (2010). Toward a language modeling approach for consumer review spam detection. In *2010 ieee 7th international conference on e-business engineering* (pp. 1–8).

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, 159–174.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

Li, H., Chen, Z., Liu, B., Wei, X., & Shao, J. (2014). Spotting fake reviews via collective positive-unlabeled learning. In *2014 ieee international conference on data mining* (pp. 899–904).

Li, H., Wang, H., Yang, Z., & Odagaki, M. (2017). Variation autoencoder based network representation learning for classification. In *Proceedings of acl 2017, student research workshop* (pp. 56–61).

Li, J., Ott, M., & Cardie, C. (2013). Identifying manipulated offerings on review portals. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1933–1942).

Lim, E.-P., Nguyen, V.-A., Jindal, N., Liu, B., & Lauw, H. W. (2010). Detecting product review spammers using rating behaviors. In *Proceedings of the 19th acm international conference on information and knowledge management* (pp. 939–948).

Lim, K. W., & Buntine, W. (2015). Bibliographic analysis with the citation network topic model. In *Asian conference on machine learning* (pp. 142–158).

Lin, Y., Zhu, T., Wu, H., Zhang, J., Wang, X., & Zhou, A. (2014). Towards online anti-opinion spam: Spotting fake reviews from the review sequence. In *2014 ieee/acm international conference on advances in social networks analysis and mining (asonam 2014)* (pp. 261–264).

Liu, B. (2007). *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media.

Liu, B., Dai, Y., Li, X., Lee, W. S., & Philip, S. Y. (2003). Building text classifiers using positive and unlabeled examples. In *Icdm* (Vol. 3, pp. 179–188).

Liu, J., Wang, C., Gao, J., & Han, J. (2013). Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 2013 siam international conference on data mining* (pp. 252–260).

Luo, N., Deng, H., Zhao, L., Liu, Y., Wang, X., & Tan, Z. (2017). Multi-aspect feature based neural network model in detecting fake reviews. In *Information science and control engineering (icisce), 2017 4th international conference on* (pp. 475–479).

Ma, B. L. W. H. Y., & Liu, B. (1998). Integrating classification and association rule mining. In *Proceedings of the fourth international conference on knowledge discovery and data mining* (pp. 24–25).

Macdonald, C., Ounis, I., & Soboroff, I. (2007). Overview of the trec 2007 blog track. In *Trec* (Vol. 7, pp. 31–43).

Macskassy, S. A., & Provost, F. (2003). *A simple relational classifier* (Tech. Rep.). NEW YORK UNIV NY STERN SCHOOL OF BUSINESS.

Martinez-Romo, J., & Araujo, L. (2009). Web spam identification through language model analysis. In *Proceedings of the 5th international workshop on adversarial information retrieval on the web* (pp. 21–28).

Mayzlin, D., Dover, Y., & Chevalier, J. (2014). Promotional reviews: An empirical investigation of online review manipulation. *American Economic Review*, *104*(8), 2421–55.

Miháltz, M. (2021, 10). *Github - mmihaltz/word2vec-googlenews-vectors: word2vec google news model.* `https://github.com/mmihaltz/word2vec-GoogleNews-vectors`. ((Accessed on 10/05/2021))

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, *3*(4), 235–244.

Mishne, G., Carmel, D., Lempel, R., et al. (2005). Blocking blog spam with language model disagreement. In *Airweb* (Vol. 5, pp. 1–6).

Mnih, A., & Hinton, G. E. (2009). A scalable hierarchical distributed language model. In *Advances in neural information processing systems* (pp. 1081–1088).

Morin, F., & Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats* (Vol. 5, pp. 246–252).

Mukherjee, A., Liu, B., & Glance, N. (2012). Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st international conference on world wide web* (pp. 191–200).

Mukherjee, A., Liu, B., Wang, J., Glance, N., & Jindal, N. (2011). Detecting group review spam. In *Proceedings of the 20th international conference companion on world wide web* (pp. 93–94).

Mukherjee, A., Venkataraman, V., Liu, B., Glance, N., et al. (2013). Fake review detection: Classification and analysis of real and pseudo reviews. *UIC-CS-03-2013. Technical Report*.

Mukherjee, A., Venkataraman, V., Liu, B., & Glance, N. S. (2013). What yelp fake review filter might be doing? In *Icwsm* (pp. 409–418).

Nasukawa, T., & Yi, J. (2003). Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on knowledge capture* (pp. 70–77).

Newman, M. L., Pennebaker, J. W., Berry, D. S., & Richards, J. M. (2003). Lying words: Predicting deception from linguistic styles. *Personality and social psychology bulletin*, *29*(5), 665–675.

Ni, J., Chang, S., Liu, X., Cheng, W., Chen, H., Xu, D., & Zhang, X. (2018). Co-regularized deep multi-network embedding. In *Proceedings of the 2018 world wide web conference* (pp. 469–478).

Nlp with gensim (word2vec). (2021, 10). `https://samyzaf.com/ML/nlp/nlp.html`. ((Accessed on 10/05/2021))

Ott, M., Cardie, C., & Hancock, J. T. (2013). Negative deceptive opinion spam. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies* (pp. 497–501).

Ott, M., Choi, Y., Cardie, C., & Hancock, J. T. (2011). 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* (pp. 309–319).

Ou, M., Cui, P., Pei, J., Zhang, Z., & Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1105–1114).

Pan, S., Wu, J., Zhu, X., Zhang, C., & Wang, Y. (2016). Tri-party deep network representation. *Network*, *11*(9), 12.

Pennebaker, J., Chung, C., Ireland, M., Gonzales, A., & Booth, R. (2007). The development and psychometric properties of liwc2007. austin, tx. *LIWC. Net*.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).

Pennington, J., Socher, R., & Manning, C. D. (2021, 10). *Glove: Global vectors for word representation.* `https://nlp.stanford.edu/projects/glove/`. ((Accessed on 10/05/2021))

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th acm sigkdd international conference on knowledge discovery and data mining* (pp. 701–710).

Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., & Han, J. (2017). An attention-based collaboration framework for multi-view network representation learning. In *Proceedings of the 2017 acm on conference on information and knowledge management* (pp. 1767–1776).

Rayana, S., & Akoglu, L. (2015). Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining* (pp. 985–994).

Rayana, S., & Akoglu, L. (2016). Collective opinion spam detection using active inference. In *Proceedings of the 2016 siam international conference on data mining* (pp. 630–638).

Rayson, P., Wilson, A., & Leech, G. (2002). Grammatical word class variation within the british national corpus sampler. In *New frontiers of corpus research* (pp. 295–306). Brill Rodopi.

Ren, Y., & Ji, D. (2017). Neural networks for deceptive opinion spam detection: An empirical study. *Information Sciences*, *385*, 213–224.

Ren, Y., Ji, D., & Zhang, H. (2014). Positive unlabeled learning for deceptive reviews detection. In *Emnlp* (pp. 488–498).

Rosso, P., & Cagnina, L. C. (2017). Deception detection and opinion spam. In *A practical guide to sentiment analysis* (pp. 155–171). Springer.

Sadowski, C., & Levin, G. (2007). Simhash: Hash-based similarity detection. *Technical report, Google*.

Shi, Y., Han, F., He, X., He, X., Yang, C., Luo, J., & Han, J. (2018). mvn2vec: Preservation and collaboration in multi-view network embedding. *arXiv preprint arXiv:1801.06597*.

Stamatatos, E. (2009). A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, *60*(3), 538–556.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, *3*(Dec), 583–617.

Sun, Y., Bui, N., Hsieh, T.-Y., & Honavar, V. (2018). Multi-view network embedding via graph factorization clustering and co-regularized multi-view agreement. In *2018 ieee international conference on data mining workshops (icdmw)* (pp. 1006–1013).

Sun, Y., Han, J., Yan, X., Yu, P. S., & Wu, T. (2011). Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, *4*(11), 992–1003.

Sun, Y., Wang, S., Hsieh, T.-Y., Tang, X., & Honavar, V. (2019). Megan: A generative adversarial network for multi-view network embedding. *arXiv preprint arXiv:1909.01084*.

Taboada, M. (2016). Sentiment analysis: An overview from linguistics. *Annual Review of Linguistics*, *2*, 325–347.

Tan, P.-N., & Jin, R. (2004). Ordering patterns by combining opinions from multiple sources. In *Proceedings of the tenth acm sigkdd international conference on knowledge discovery and data mining* (pp. 695–700).

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web* (pp. 1067–1077).

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., & Su, Z. (2008). Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th acm sigkdd international conference on knowledge discovery and data mining* (pp. 990–998).

Tang, L., & Liu, H. (2009a). Relational learning via latent social dimensions. In *Proceedings of the 15th acm sigkdd international conference on knowledge discovery and data mining* (pp. 817–826).

Tang, L., & Liu, H. (2009b). Scalable learning of collective behavior based on sparse social dimensions. In *Proceedings of the 18th acm conference on information and knowledge management* (pp. 1107–1116).

Tang, L., & Liu, H. (2011). Leveraging social media networks for classification. *Data Mining and Knowledge Discovery*, *23*(3), 447–478.

Tu, K., Cui, P., Wang, X., Yu, P. S., & Zhu, W. (2018). Deep recursive network embedding with regular equivalence. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (pp. 2357–2366).

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, *9*(11).

Van Loan, C. F. (1976). Generalizing the singular value decomposition. *SIAM Journal on numerical Analysis*, *13*(1), 76–83.

Wang, C.-C., Day, M.-Y., Chen, C.-C., & Liou, J.-W. (2018). Detecting spamming reviews using long short-term memory recurrent neural network framework. In *Proceedings of the 2nd international conference on e-commerce, e-business and e-government* (pp. 16–20).

Wang, D., Cui, P., & Zhu, W. (2016). Structural deep network embedding. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1225–1234).

Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., … Guo, M. (2017). Graphgan: Graph representation learning with generative adversarial nets. *arXiv preprint arXiv:1711.08267*.

Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., & Yang, S. (2017). Community preserving network embedding. In *Thirty-first aaai conference on artificial intelligence.*

Wang, X., Liu, K., He, S., & Zhao, J. (2016). Learning to represent review with tensor decomposition for spam detection. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 866–875).

Wang, X., Liu, K., & Zhao, J. (2017a). Detecting deceptive review spam via attention-based neural networks. In *National ccf conference on natural language processing and chinese computing* (pp. 866–876).

Wang, X., Liu, K., & Zhao, J. (2017b). Handling cold-start problem in review spam detection by jointly embedding texts and behaviors. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)* (Vol. 1, pp. 366–376).

Weston, J., Ratle, F., Mobahi, H., & Collobert, R. (2012). Deep learning via semi-supervised embedding. In *Neural networks: Tricks of the trade* (pp. 639–655). Springer.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, *2*(1-3), 37–52.

Wu, G., Greene, D., & Cunningham, P. (2010). Merging multiple criteria to identify suspicious reviews. In *Proceedings of the fourth acm conference on recommender systems* (pp. 241–244).

Xie, S., Wang, G., Lin, S., & Yu, P. S. (2012). Review spam detection via temporal pattern discovery. In *Proceedings of the 18th acm sigkdd international conference on knowledge discovery and data mining* (pp. 823–831).

Yang, C., Liu, Z., Zhao, D., Sun, M., & Chang, E. Y. (2015). Network representation learning with rich text information. In *Ijcai* (Vol. 2015, pp. 2111–2117).

Yang, Z., Cohen, W., & Salakhudinov, R. (2016). Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning* (pp. 40–48).

Ying, X. (2019). An overview of overfitting and its solutions. In *Journal of physics: Conference series* (Vol. 1168, p. 022022).

Yuan, L., Li, D., Wei, S., & Wang, M. (2018). Research of deceptive review detection based on target product identification and metapath feature weight calculation. *Complexity*, *2018*.

Zhang, H., Qiu, L., Yi, L., & Song, Y. (2018). Scalable multiplex network embedding. In *Ijcai* (Vol. 18, pp. 3082–3088).

Zhang, W., Du, Y., Yoshida, T., & Wang, Q. (2018). Dri-rcnn: An approach to deceptive review identification using recurrent convolutional neural network. *Information Processing & Management*, *54*(4), 576–592.

Zhao, Y., & Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis.

Zheng, Y., Li, G., Li, Y., Shan, C., & Cheng, R. (2017). Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, *10*(5), 541–552.

Zhu, X., Ghahramani, Z., & Lafferty, J. D. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th international conference on machine learning (icml-03)* (pp. 912–919).

APPENDIX A. SPAMMER CLASSIFICATION

In this appendix, we include the complete experiment results of spammer classification with different configurations. For each configuration, we use the learned multi-view reviewer embeddings as the features of reviewers and spammer labels as the prediction target. The trained models were chosen from 5-fold cross validation and the best F1 scores are reported in the tables. We also include the bar charts for the comparison purpose.

We use the multi-view reviewer embeddings learned with different spammer thresholds 1, 2, 3, 4, 5, 10%, 20%, 30%, 40%, 50% on the spammer classification. For each spammer threshold, firstly, we fix the epochs as 20 and test different embedding size 3, 10, 20, 50, 100. Secondly, we fix the embedding size as 100 and test different epochs 1, 5, 10, 20.

Table A.1. *Spammer classification (different embedding sizes, spammer threshold=1)*

embedding size	3	10	20	50	100
YelpChi	0.566	0.609	0.596	0.603	0.587
YelpNYC	0.556	0.582	0.590	0.603	0.612
YelpZIP	0.572	0.594	0.609	0.592	0.621

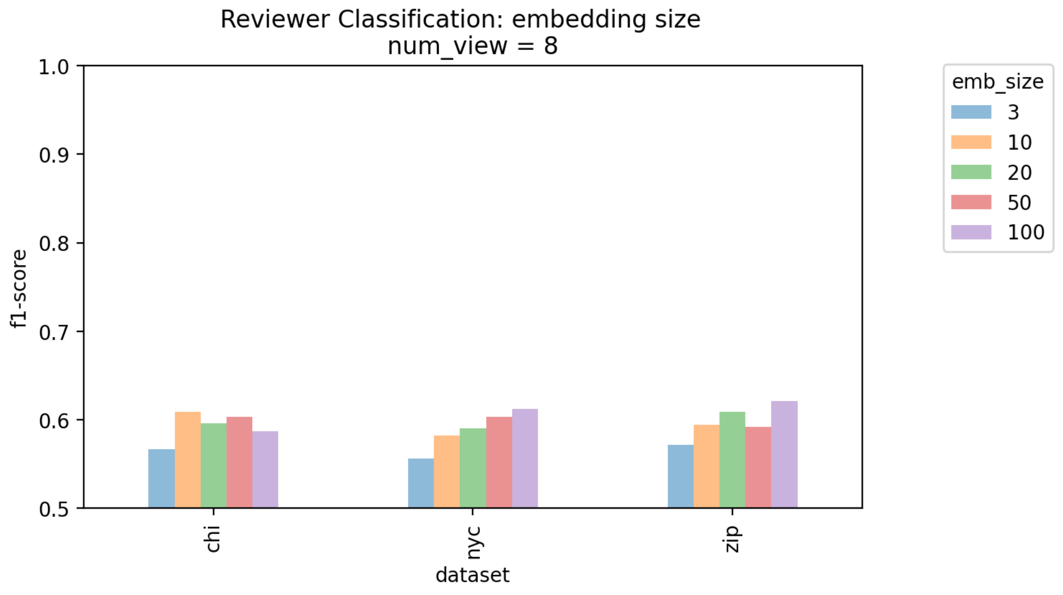


Figure A.1. Spammer classification (different embedding sizes, spammer threshold=1)

Table A.2. *Spammer classification (different epochs, spammer threshold=1)*

training epochs	1	5	10	20
YelpChi	0.575	0.602	0.588	0.587
YelpNYC	0.581	0.586	0.600	0.612
YelpZIP	0.583	0.599	0.614	0.621

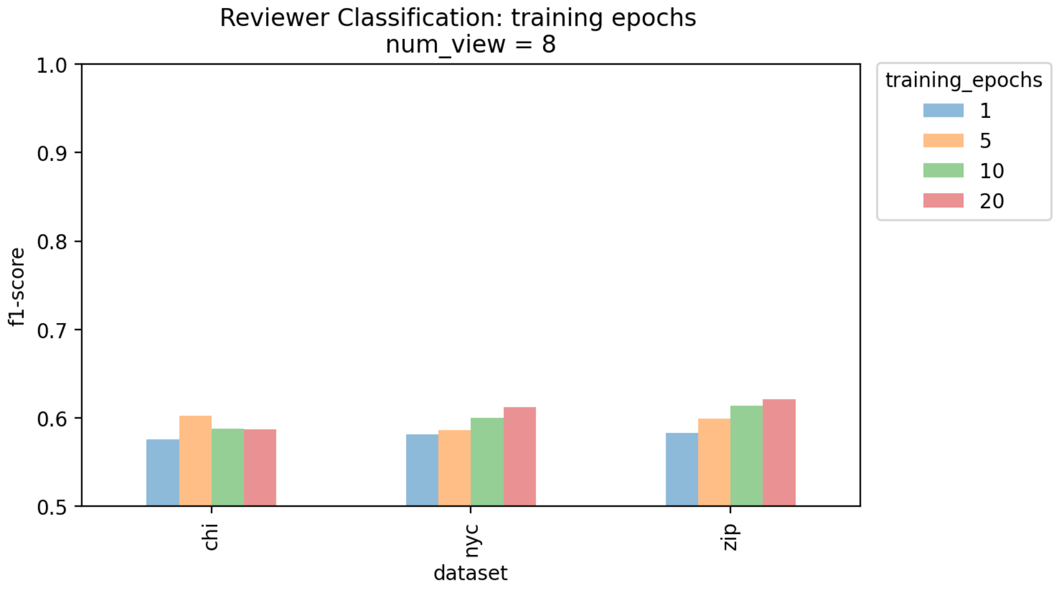


Figure A.2. Spammer classification (different epochs, spammer threshold=1)

Table A.3. *Spammer classification (different embedding sizes, spammer threshold=2)*

embedding size	3	10	20	50	100
YelpChi	0.835	0.820	0.853	0.826	0.851
YelpNYC	0.914	0.976	0.979	0.980	0.982
YelpZIP	0.908	0.955	0.958	0.961	0.963

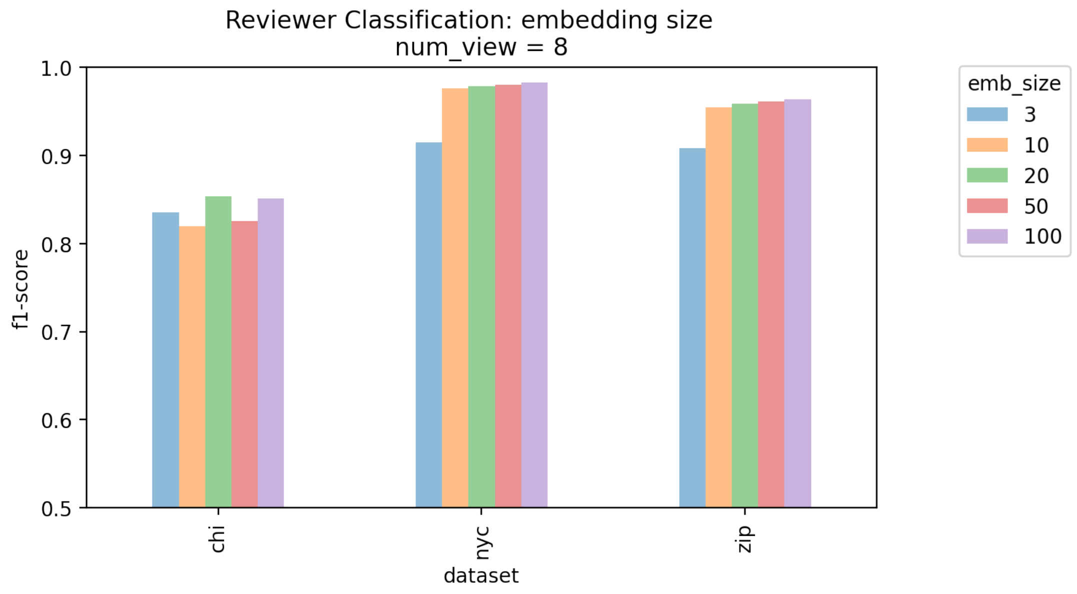


Figure A.3. Spammer classification (different embedding sizes, spammer threshold=2)

Table A.4. *Spammer classification (different epochs, spammer threshold=2)*

training epochs	1	5	10	20
YelpChi	0.878	0.815	0.819	0.851
YelpNYC	0.922	0.981	0.976	0.982
YelpZIP	0.927	0.959	0.966	0.963

Table A.5. *Spammer classification (different embedding sizes, spammer threshold=3)*

embedding size	3	10	20	50	100
YelpChi	0.804	0.833	0.871	0.845	0.859
YelpNYC	0.795	0.818	0.832	0.851	0.846
YelpZIP	0.822	0.833	0.862	0.841	0.864

Table A.6. *Spammer classification (different epochs, spammer threshold=3)*

training epochs	1	5	10	20
YelpChi	0.805	0.742	0.833	0.859
YelpNYC	0.842	0.844	0.858	0.846
YelpZIP	0.837	0.863	0.868	0.864

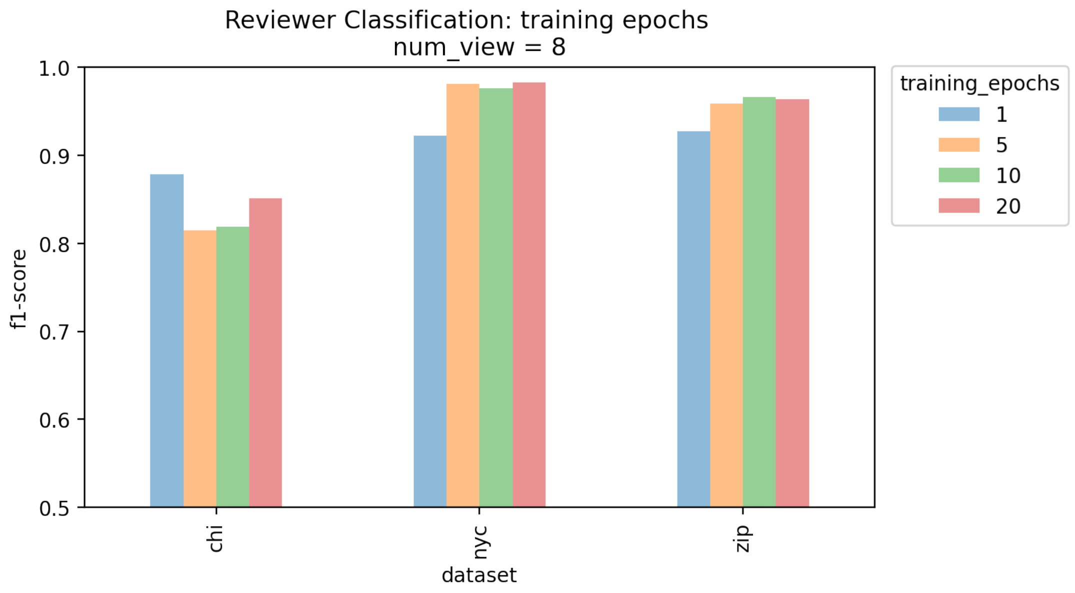


Figure A.4. Spammer classification (different epochs, spammer threshold=2)

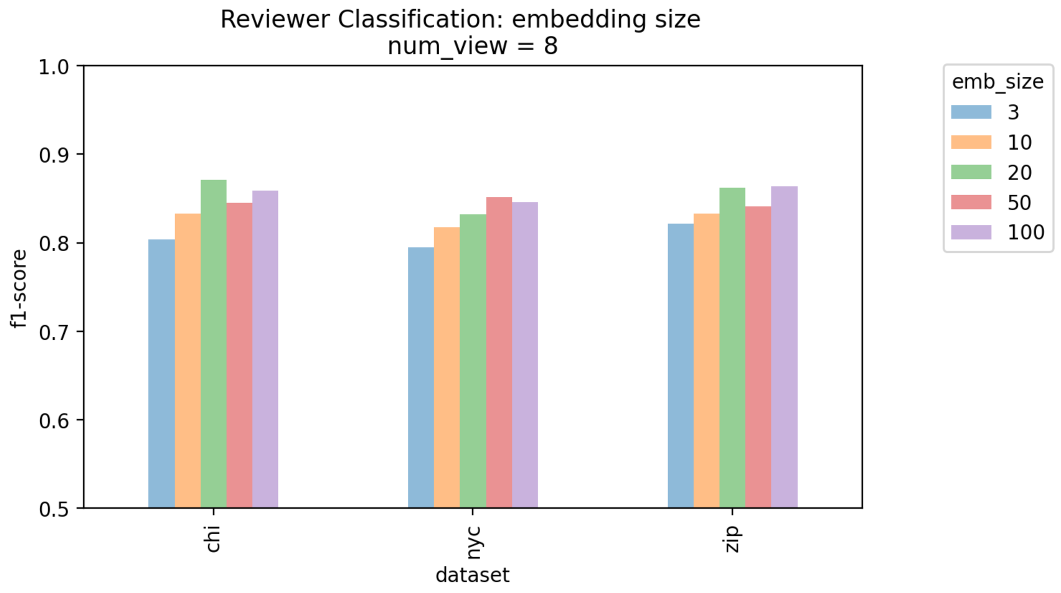


Figure A.5. Spammer classification (different embedding sizes, spammer threshold=3)

Table A.7. *Spammer classification (different embedding sizes, spammer threshold=4)*

embedding size	3	10	20	50	100
YelpChi	0.889	0.813	0.889	0.929	0.929
YelpNYC	0.873	0.905	0.875	0.905	0.885
YelpZIP	0.865	0.886	0.901	0.909	0.907

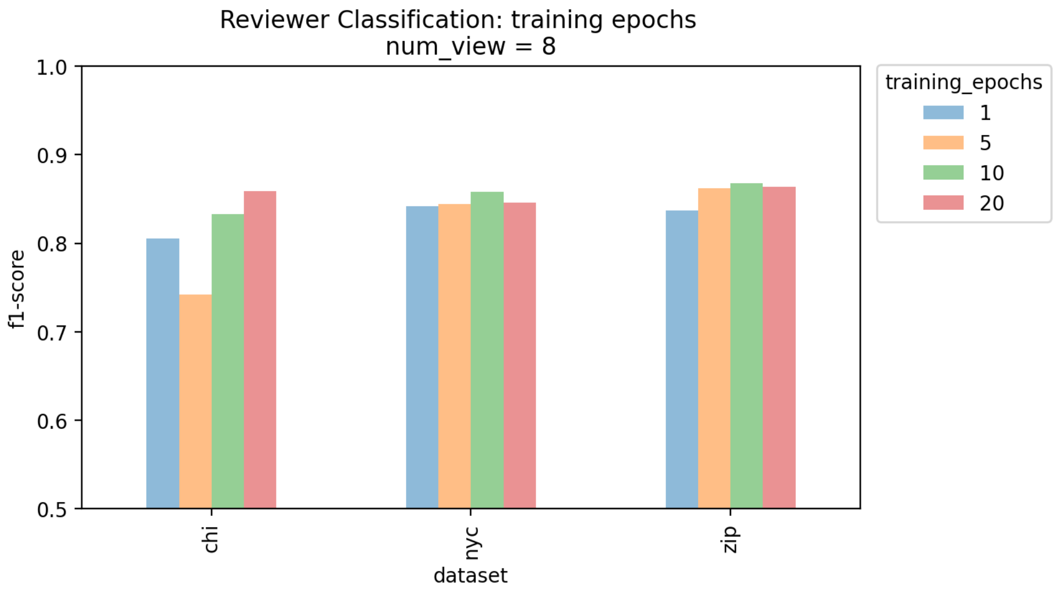


Figure A.6. Spammer classification (different epochs, spammer threshold=3)

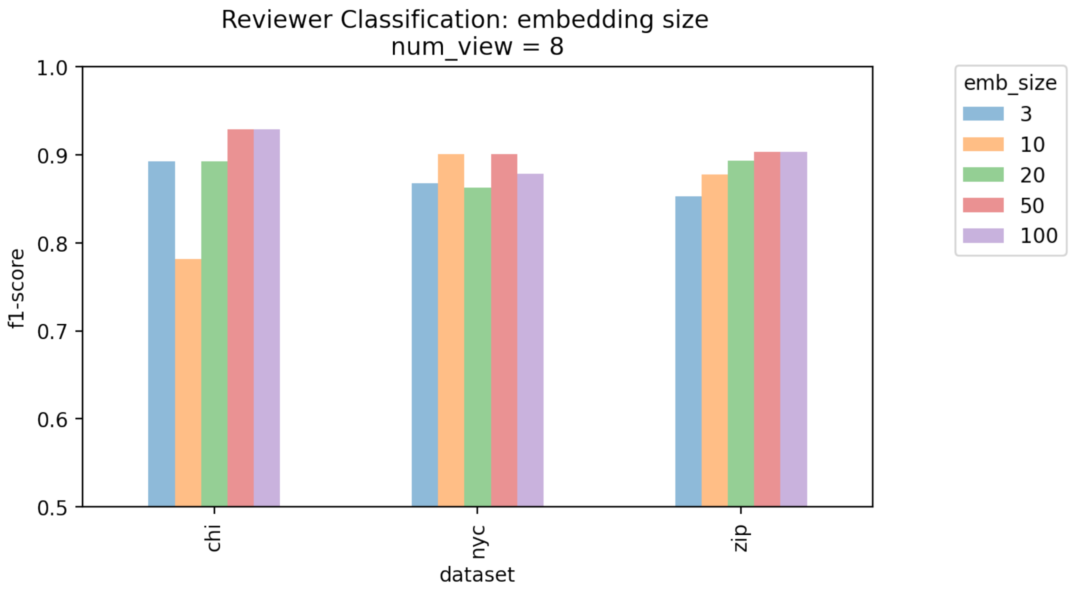


Figure A.7. Spammer classification (different embedding sizes, spammer threshold=4)

Table A.8. *Spammer classification (different epochs, spammer threshold=4)*

training epochs	1	5	10	20
YelpChi	0.933	0.966	1.000	0.929
YelpNYC	0.898	0.866	0.915	0.885
YelpZIP	0.901	0.892	0.906	0.907

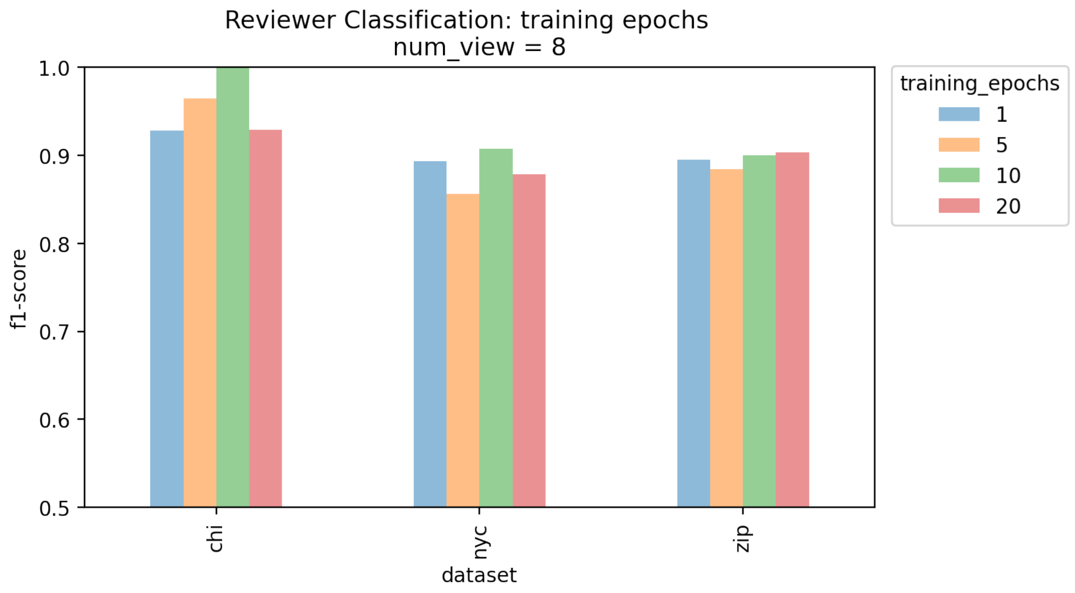


Figure A.8. Spammer classification (different epochs, spammer threshold=4)

Table A.9. *Spammer classification (different embedding sizes, spammer threshold=5)*

embedding size	3	10	20	50	100
YelpChi	0.928	1.000	0.928	1.000	0.857
YelpNYC	0.859	0.858	0.899	0.913	0.899
YelpZIP	0.906	0.931	0.909	0.904	0.901

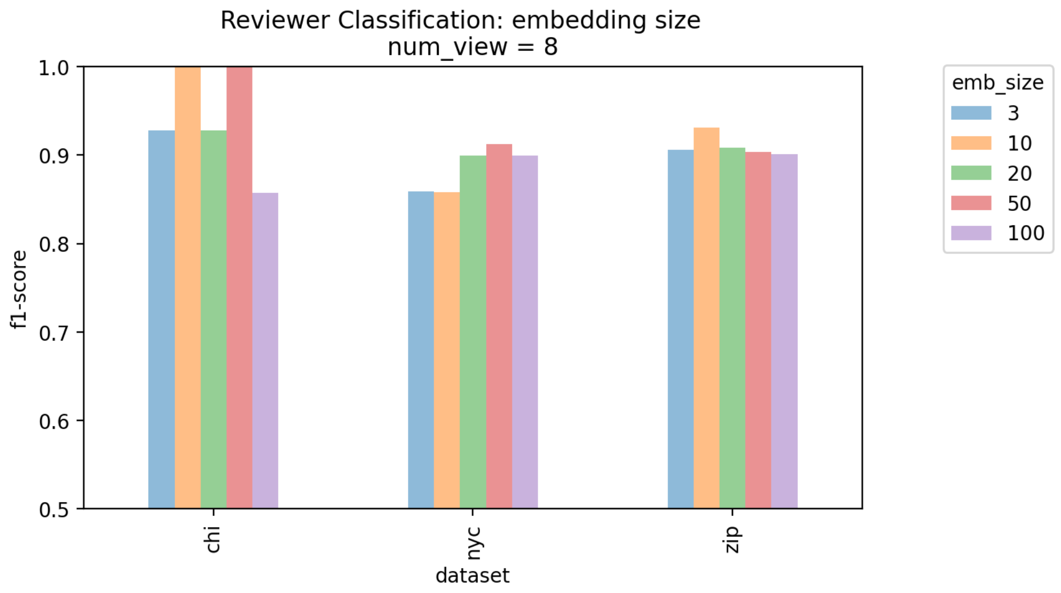


Figure A.9. Spammer classification (different embedding sizes, spammer threshold=5)

Table A.10. *Spammer classification (different epochs, spammer threshold=5)*

training epochs	1	5	10	20
YelpChi	0.928	0.928	0.854	0.857
YelpNYC	0.899	0.919	0.912	0.899
YelpZIP	0.882	0.909	0.890	0.901

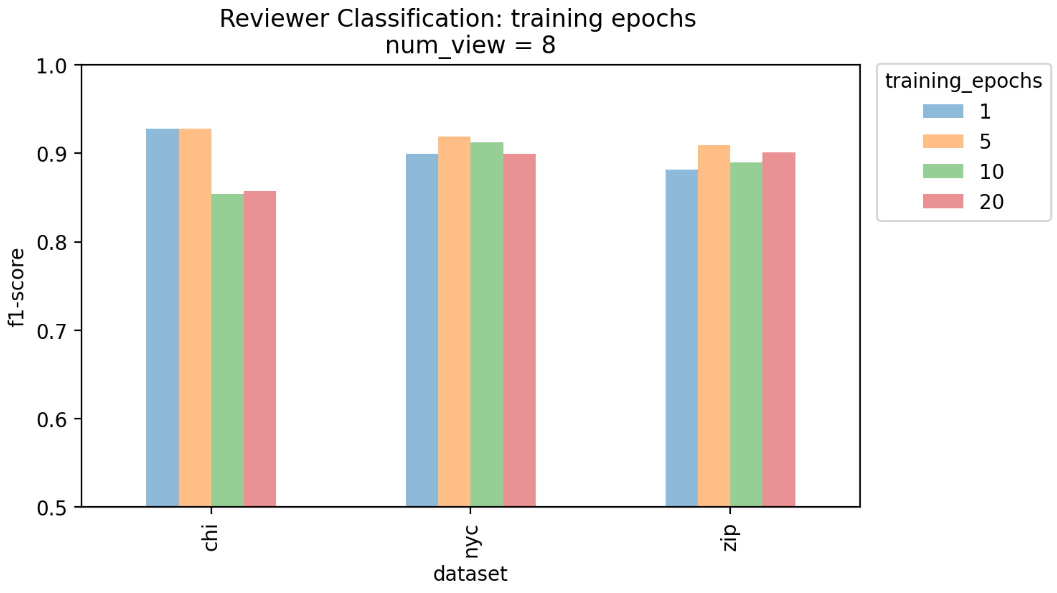


Figure A.10. Spammer classification (different epochs, spammer threshold=5)

Table A.11. *Spammer classification (different embedding sizes, spammer threshold=10%)*

embedding size	3	10	20	50	100
YelpChi	0.593	0.592	0.599	0.605	0.616
YelpNYC	0.552	0.600	0.589	0.606	0.590
YelpZIP	0.538	0.596	0.576	0.591	0.599

Table A.12. *Spammer classification (different epochs, spammer threshold=10%)*

training epochs	1	5	10	20
YelpChi	0.589	0.605	0.595	0.616
YelpNYC	0.587	0.611	0.603	0.590
YelpZIP	0.602	0.598	0.591	0.599

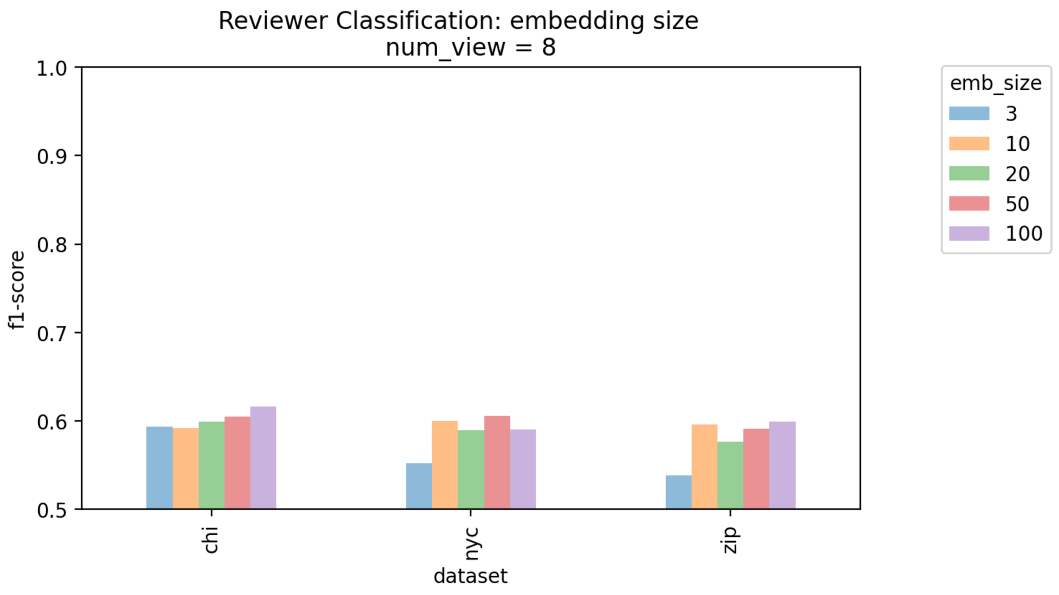


Figure A.11. Spammer classification (different embedding sizes, spammer threshold=10%)

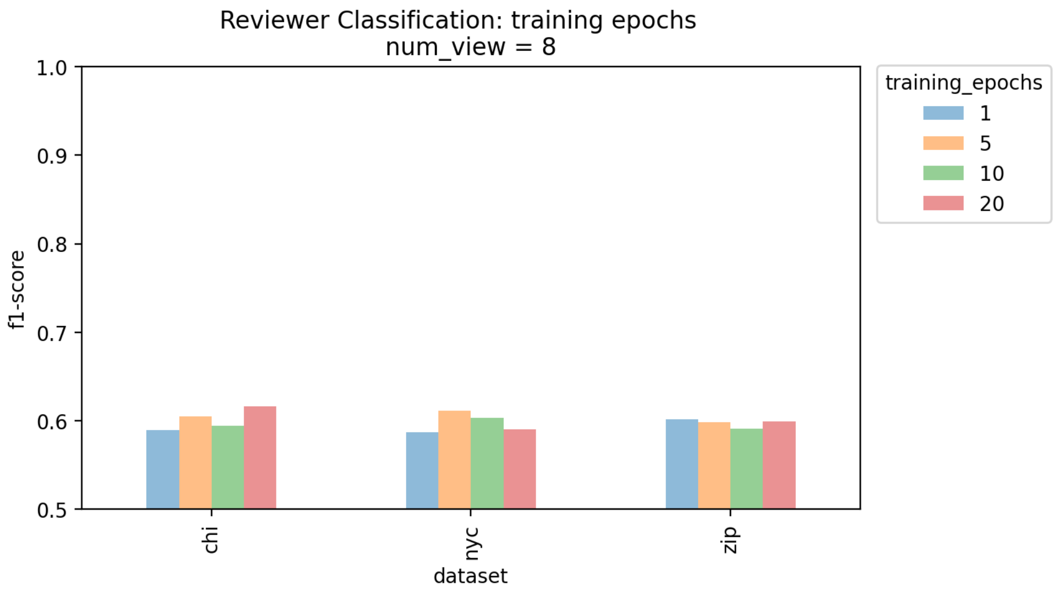


Figure A.12. Spammer classification (different epochs, spammer threshold=10%)

Table A.13. *Spammer classification (different embedding sizes, spammer threshold=20%)*

embedding size	3	10	20	50	100
YelpChi	0.575	0.606	0.597	0.597	0.599
YelpNYC	0.577	0.598	0.612	0.611	0.622
YelpZIP	0.566	0.586	0.604	0.592	0.601

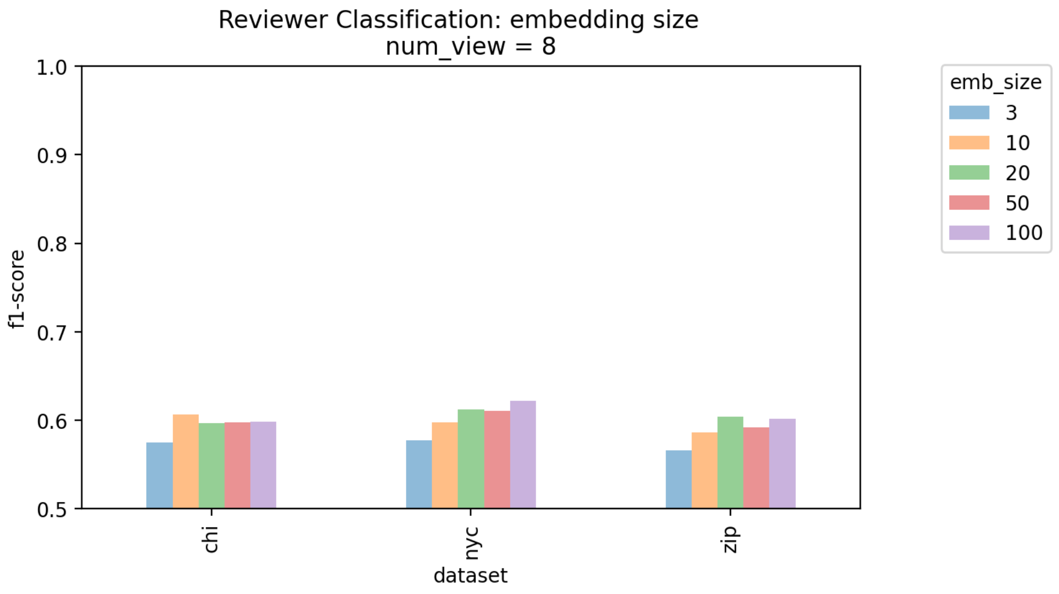


Figure A.13. Spammer classification (different embedding sizes, spammer threshold=20%)

Table A.14. *Spammer classification (different epochs, spammer threshold=20%)*

training epochs	1	5	10	20
YelpChi	0.591	0.592	0.589	0.599
YelpNYC	0.547	0.588	0.587	0.622
YelpZIP	0.579	0.582	0.592	0.601

Table A.15. *Spammer classification (different embedding sizes, spammer threshold=30%)*

embedding size	3	10	20	50	100
YelpChi	0.559	0.620	0.593	0.595	0.605
YelpNYC	0.548	0.582	0.591	0.595	0.598
YelpZIP	0.549	0.594	0.612	0.598	0.611

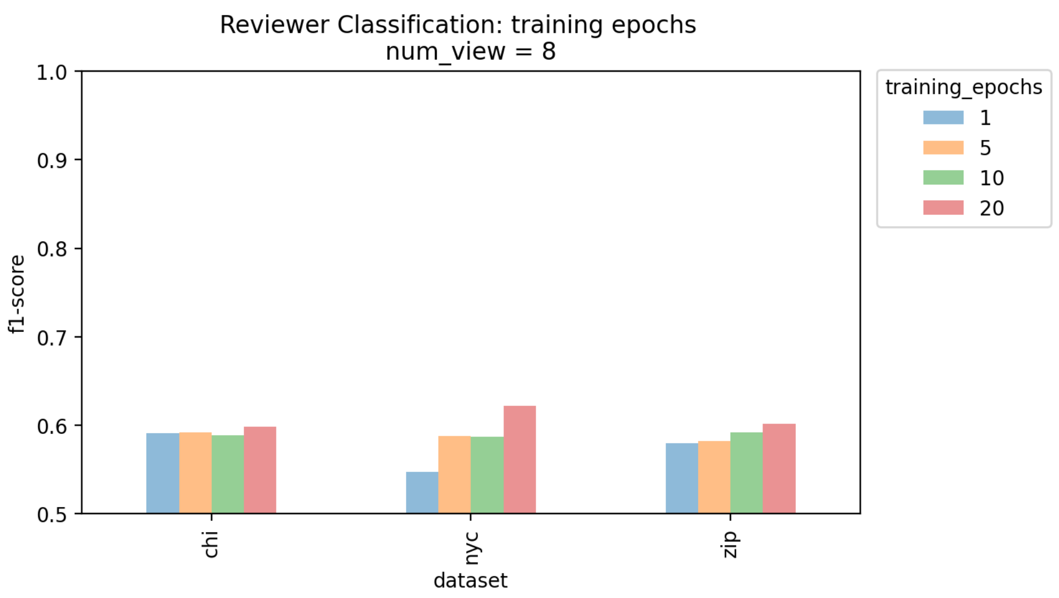


Figure A.14. Spammer classification (different epochs, spammer threshold=20%)

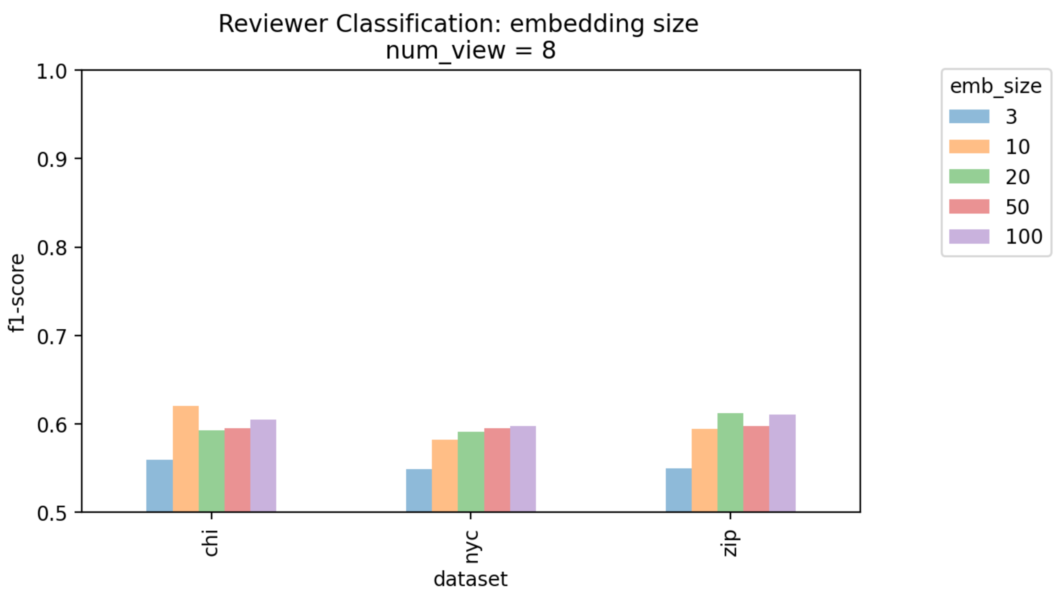


Figure A.15. Spammer classification (different embedding sizes, spammer threshold=30%)

Table A.16. *Spammer classification (different epochs, spammer threshold=30%)*

training epochs	1	5	10	20
YelpChi	0.587	0.576	0.595	0.605
YelpNYC	0.580	0.595	0.608	0.598
YelpZIP	0.575	0.603	0.597	0.611

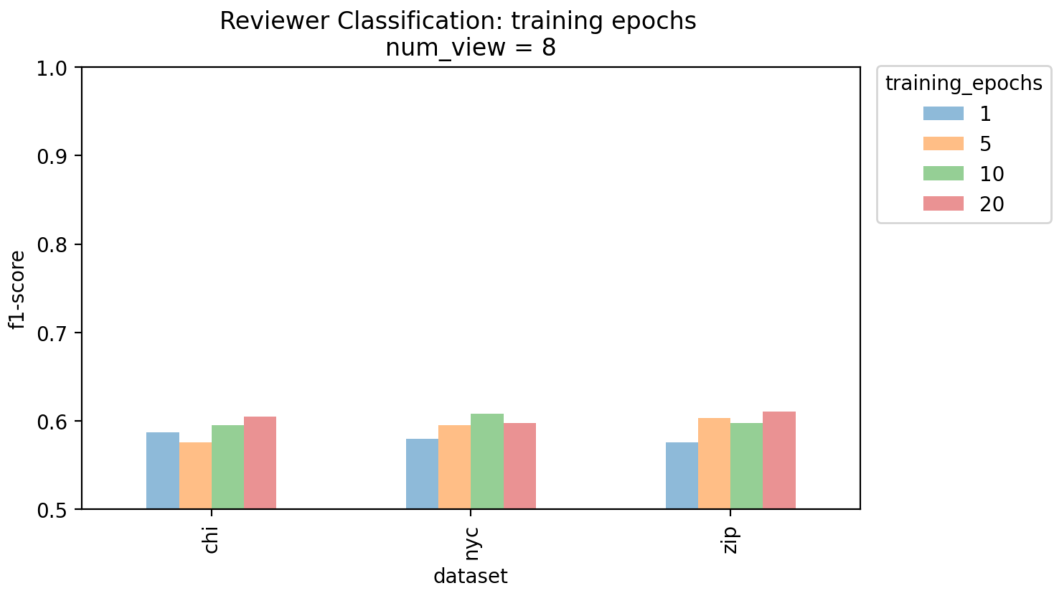


Figure A.16. Spammer classification (different epochs, spammer threshold=30%)

Table A.17. *Spammer classification (different embedding sizes, spammer threshold=40%)*

embedding size	3	10	20	50	100
YelpChi	0.568	0.587	0.583	0.615	0.600
YelpNYC	0.571	0.599	0.612	0.621	0.600
YelpZIP	0.563	0.599	0.611	0.602	0.604

Table A.18. *Spammer classification (different epochs, spammer threshold=40%)*

training epochs	1	5	10	20
YelpChi	0.603	0.587	0.603	0.600
YelpNYC	0.594	0.586	0.616	0.600
YelpZIP	0.604	0.603	0.610	0.604

Table A.19. *Spammer classification (different embedding sizes, spammer threshold=50%)*

embedding size	3	10	20	50	100
YelpChi	0.576	0.588	0.608	0.602	0.589
YelpNYC	0.551	0.622	0.620	0.607	0.607
YelpZIP	0.577	0.594	0.601	0.581	0.597

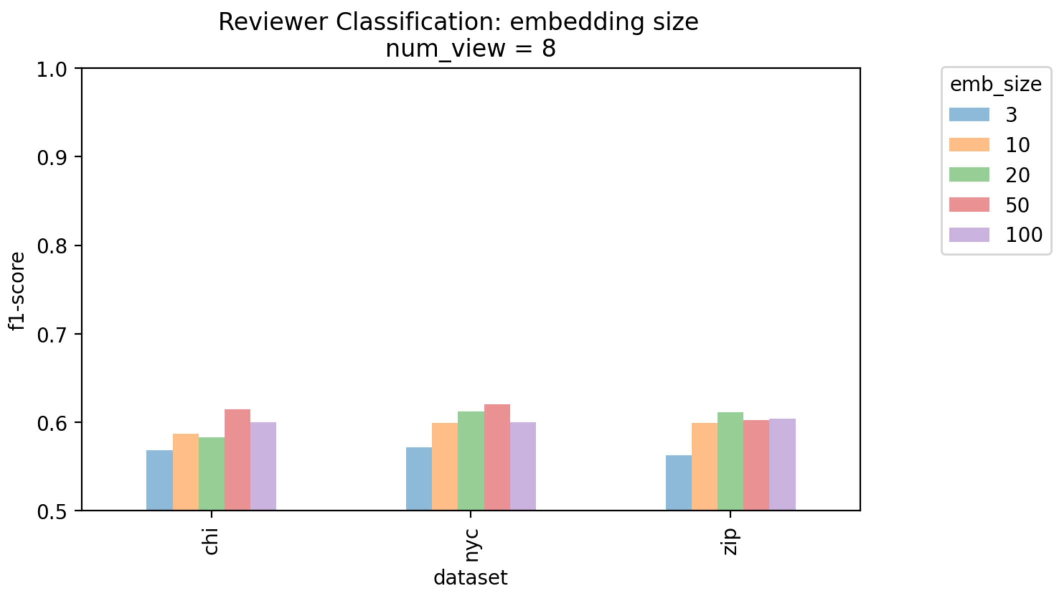


Figure A.17. Spammer classification (different embedding sizes, spammer threshold=40%)

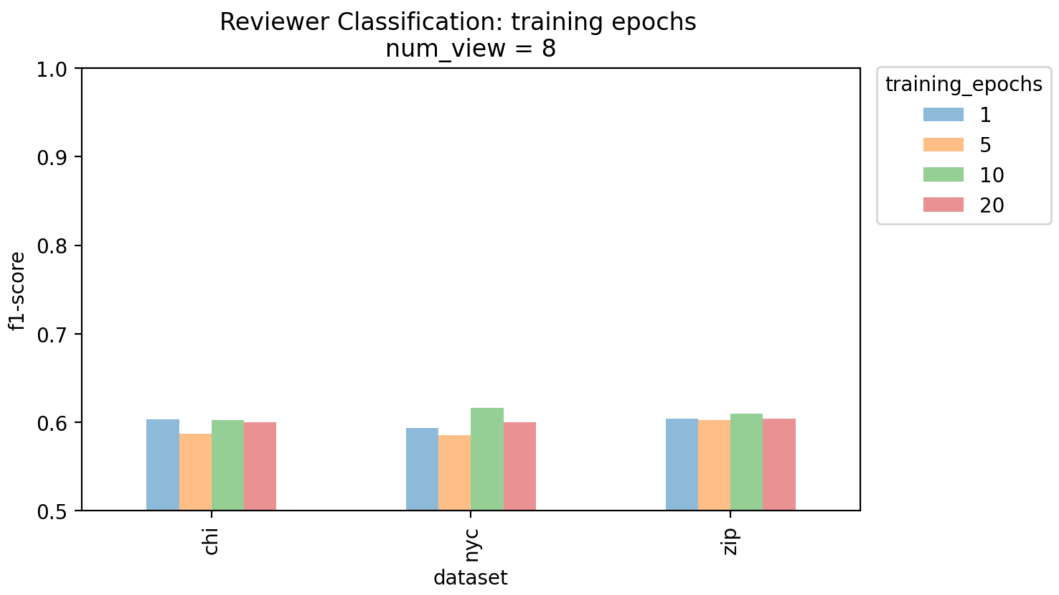


Figure A.18. Spammer classification (different epochs, spammer threshold=40%)

Table A.20. *Spammer classification (different epochs, spammer threshold=50%)*

training epochs	1	5	10	20
YelpChi	0.593	0.583	0.619	0.589
YelpNYC	0.564	0.616	0.600	0.607
YelpZIP	0.590	0.594	0.597	0.597

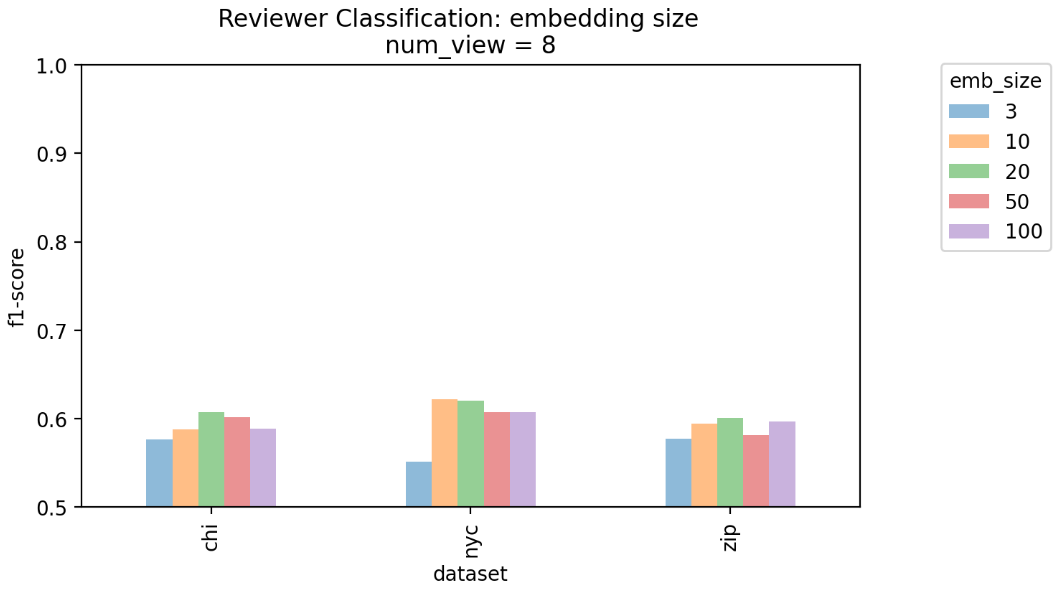


Figure A.19. Spammer classification (different embedding sizes, spammer threshold=50%)

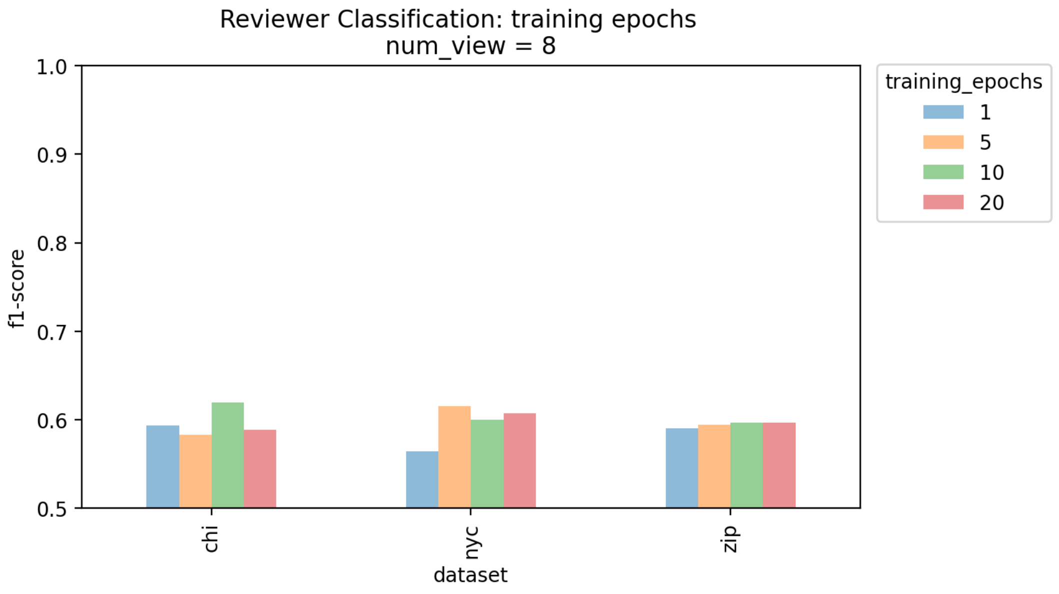


Figure A.20. Spammer classification (different epochs, spammer threshold=50%)

APPENDIX B. FAKE REVIEW CLASSIFICATION

In this appendix, we include the complete experiment results of fake review classification with different configurations. For each configuration, we use the hybrid embeddings (i.e. text + reviewer) as the features of reviews and take the filtered labels in the Yelp datasets as the prediction targets. The trained models were chosen from 5-fold cross validation and the best F1 scores are reported in the tables. We also include the bar charts for the comparison purpose.

We use the hybrid embeddings (Word2Vec/Glove text embeddings + multi-view reviewer embeddings learned with different spammer thresholds 1, 2, 3, 4, 5) on the spammer classification. For each spammer threshold, we test different embedding size 3, 10, 20, 50, 100 of the reviewer embeddings.

Table B.1. *F1 scores (Word2Vec+Reviewer, spammer threshold=1)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.633	0.653	0.655
word2vec + reviewer (emb_size=3)	0.711	0.714	0.717
word2vec + reviewer (emb_size=10)	0.710	0.732	0.727
word2vec + reviewer (emb_size=20)	0.723	0.748	0.739
word2vec + reviewer (emb_size=50)	0.729	0.771	0.779
word2vec + reviewer (emb_size=100)	0.748	0.775	0.784

Table B.2. *F1 scores (GloVe+Reviewer, spammer threshold=1)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.607	0.603	0.597
glove + reviewer (emb_size=3)	0.688	0.706	0.718
glove + reviewer (emb_size=10)	0.706	0.722	0.724
glove + reviewer (emb_size=20)	0.712	0.738	0.740
glove + reviewer (emb_size=50)	0.731	0.777	0.792
glove + reviewer (emb_size=100)	0.754	0.775	0.796

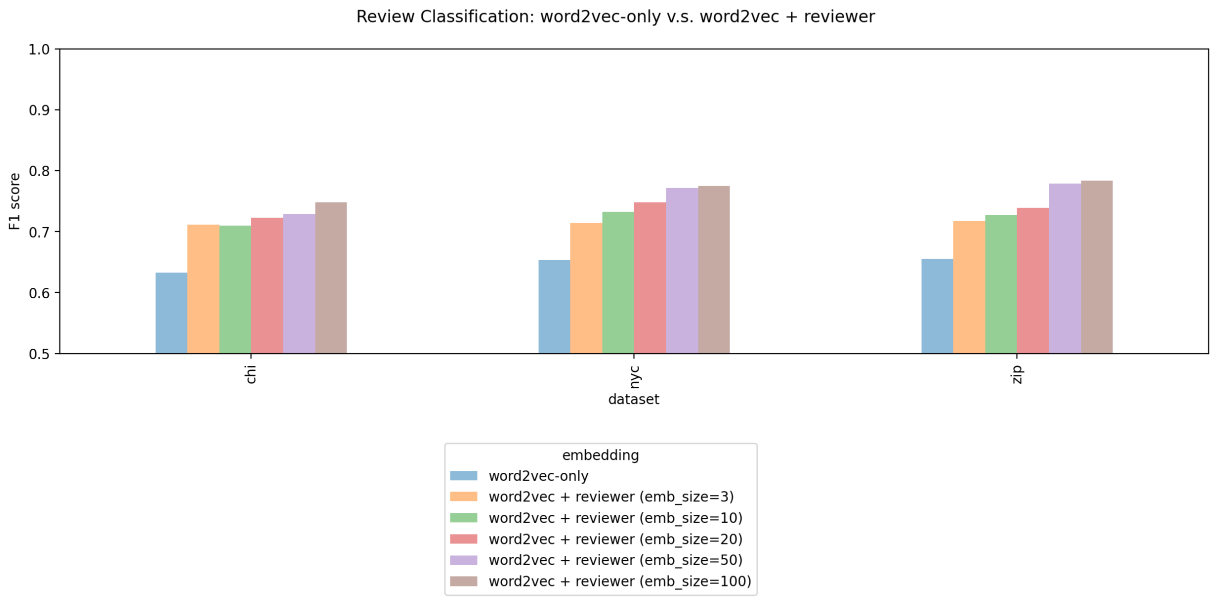


Figure B.1. Fake review classification (Word2Vec+Reviewer, spammer threshold=1)

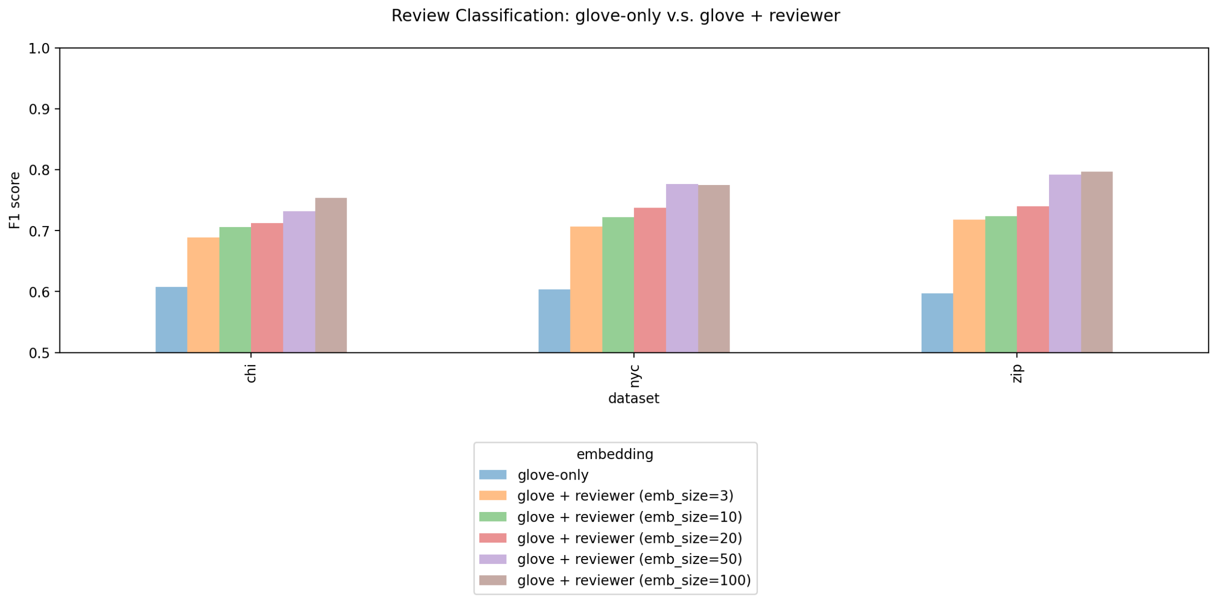


Figure B.2. Fake review classification (GloVe+Reviewer, spammer threshold=1)

Table B.3. *F1 scores (Word2Vec+Reviewer, spammer threshold=2)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.667	0.645	0.607
word2vec + reviewer (emb_size=3)	0.694	0.747	0.755
word2vec + reviewer (emb_size=10)	0.713	0.790	0.780
word2vec + reviewer (emb_size=20)	0.693	0.790	0.803
word2vec + reviewer (emb_size=50)	0.763	0.810	0.826
word2vec + reviewer (emb_size=100)	0.784	0.860	0.862

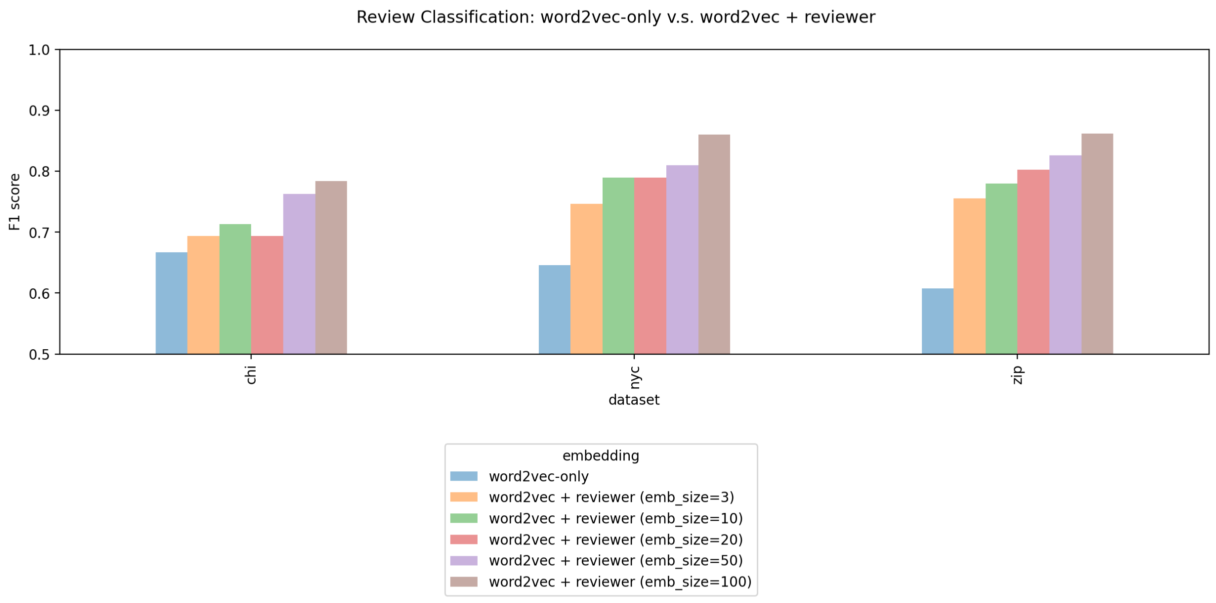


Figure B.3. Fake review classification (Word2Vec+Reviewer, spammer threshold=2)

Table B.4. *F1 scores (GloVe+Reviewer, spammer threshold=2)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.635	0.593	0.589
glove + reviewer (emb_size=3)	0.669	0.753	0.737
glove + reviewer (emb_size=10)	0.708	0.763	0.769
glove + reviewer (emb_size=20)	0.724	0.785	0.786
glove + reviewer (emb_size=50)	0.775	0.815	0.840
glove + reviewer (emb_size=100)	0.839	0.876	0.884

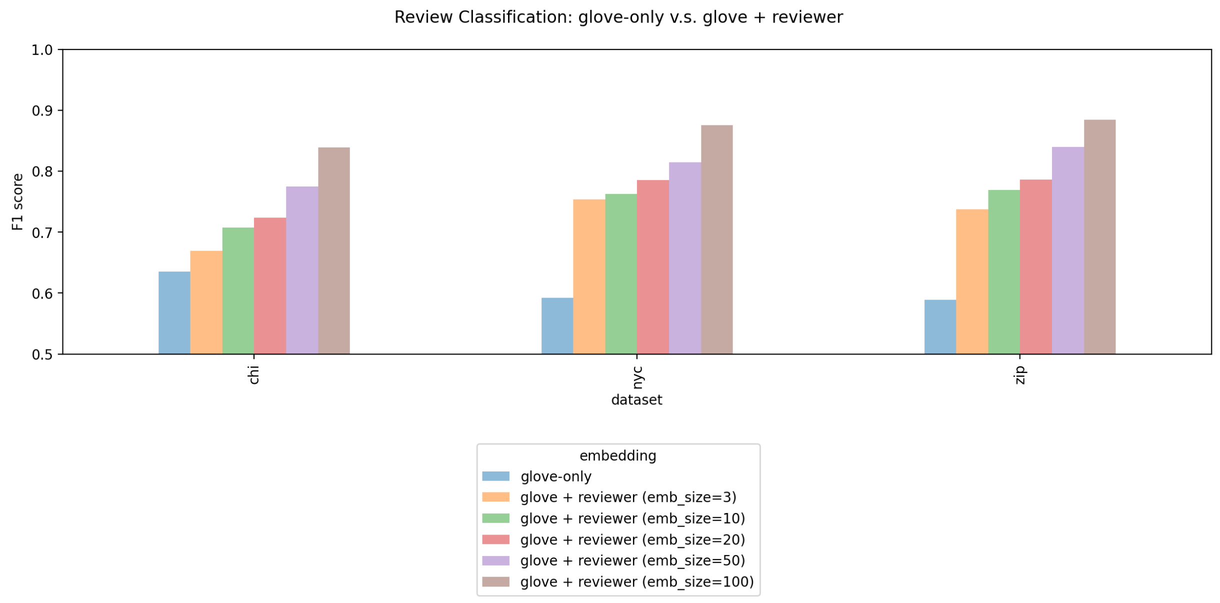


Figure B.4. Fake review classification (GloVe+Reviewer, spammer threshold=2)

Table B.5. *F1 scores (Word2Vec+Reviewer, spammer threshold=3)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.627	0.632	0.610
word2vec + reviewer (emb_size=3)	0.648	0.729	0.670
word2vec + reviewer (emb_size=10)	0.610	0.750	0.711
word2vec + reviewer (emb_size=20)	0.698	0.798	0.741
word2vec + reviewer (emb_size=50)	0.772	0.851	0.833
word2vec + reviewer (emb_size=100)	0.826	0.927	0.891

Table B.6. *F1 scores (GloVe+Reviewer, spammer threshold=3)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.610	0.619	0.569
glove + reviewer (emb_size=3)	0.664	0.705	0.659
glove + reviewer (emb_size=10)	0.670	0.735	0.714
glove + reviewer (emb_size=20)	0.698	0.796	0.752
glove + reviewer (emb_size=50)	0.804	0.861	0.864
glove + reviewer (emb_size=100)	0.865	0.929	0.908

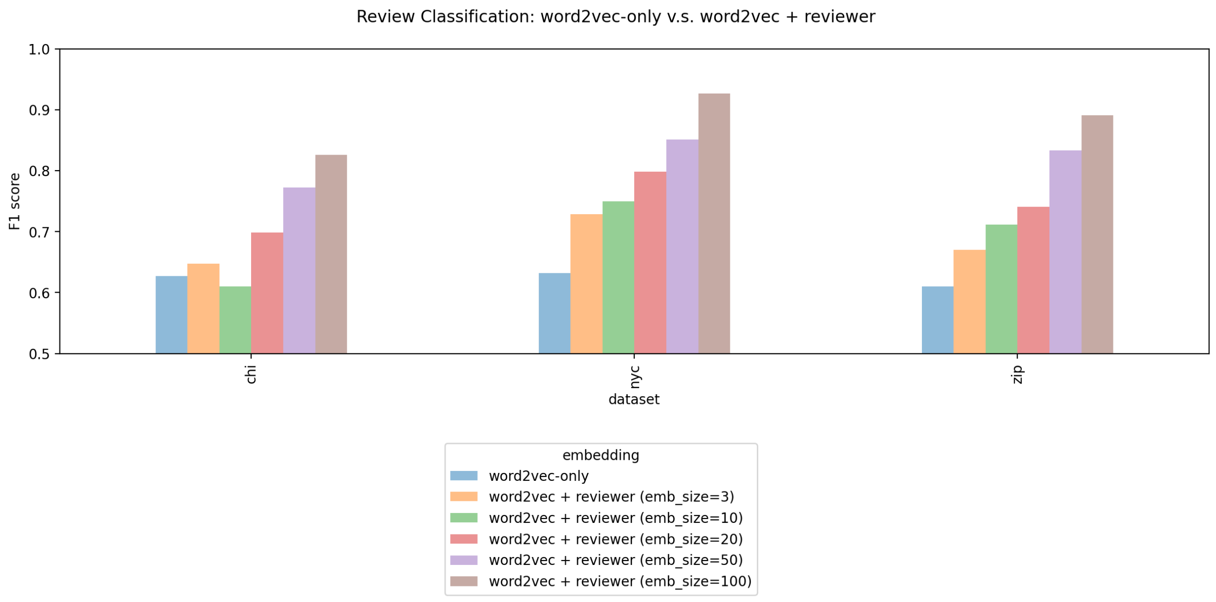


Figure B.5. Fake review classification (Word2Vec+Reviewer, spammer threshold=3)

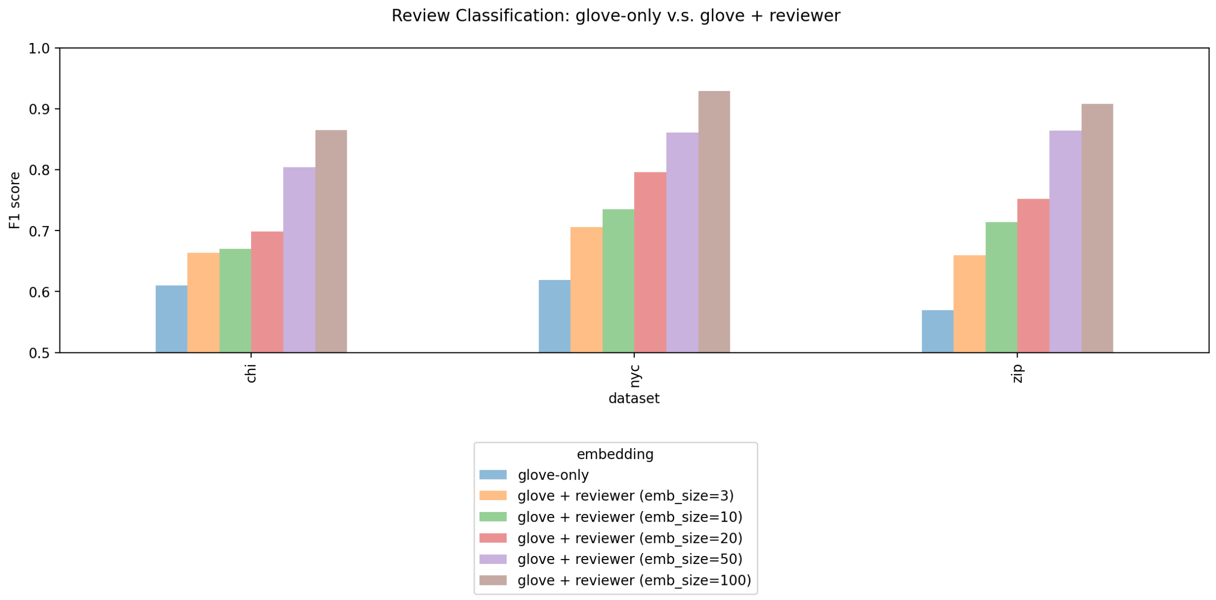


Figure B.6. Fake review classification (GloVe+Reviewer, spammer threshold=3)

Table B.7. *F1 scores (Word2Vec+Reviewer, spammer threshold=4)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.650	0.661	0.679
word2vec+reviewer (emb_size=3)	0.762	0.743	0.740
word2vec+reviewer (emb_size=10)	0.714	0.789	0.778
word2vec+reviewer (emb_size=20)	0.750	0.819	0.824
word2vec+reviewer (emb_size=50)	0.884	0.906	0.904
word2vec+reviewer (emb_size=100)	0.900	0.924	0.939

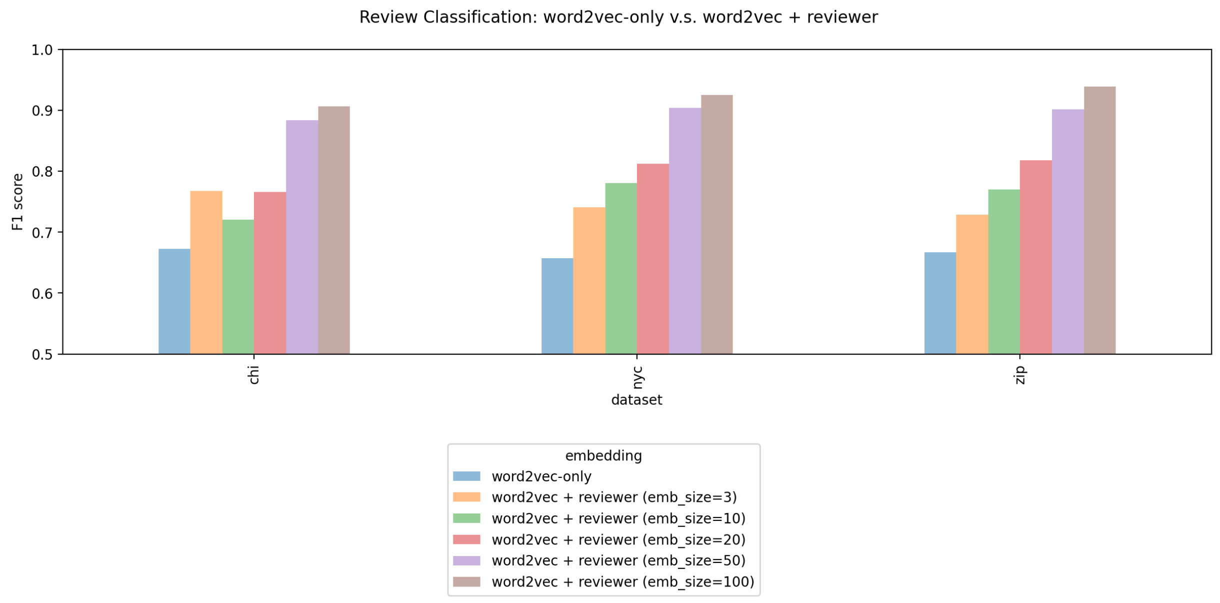


Figure B.7. Fake review classification (Word2Vec+Reviewer, spammer threshold=4)

Table B.8. *F1 scores (GloVe+Reviewer, spammer threshold=4)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.600	0.629	0.629
glove+reviewer (emb_size=3)	0.636	0.739	0.708
glove+reviewer (emb_size=10)	0.651	0.801	0.775
glove+reviewer (emb_size=20)	0.732	0.851	0.828
glove+reviewer (emb_size=50)	0.930	0.913	0.920
glove+reviewer (emb_size=100)	0.952	0.921	0.948

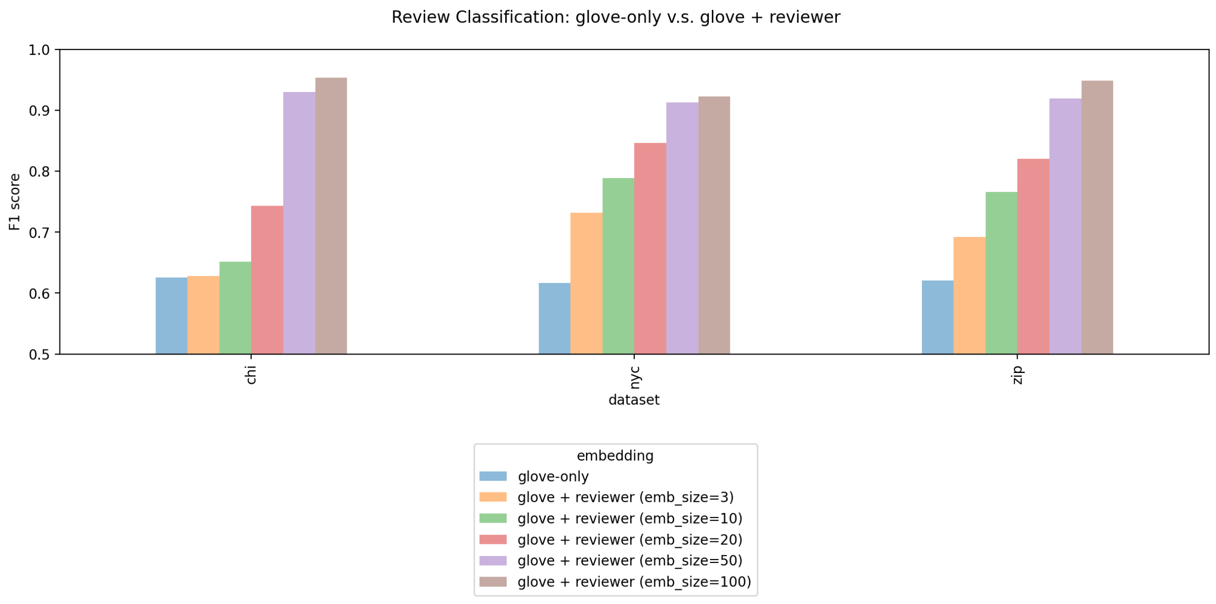


Figure B.8. Fake review classification (GloVe+Reviewer, spammer threshold=4)

Table B.9. *F1 scores (Word2Vec+Reviewer, spammer threshold=5)*

	YelpChi	YelpNYC	YelpZIP
word2vec-only	0.642	0.630	0.642
word2vec + reviewer (emb_size=3)	0.899	0.701	0.702
word2vec + reviewer (emb_size=10)	0.697	0.752	0.733
word2vec + reviewer (emb_size=20)	0.899	0.863	0.794
word2vec + reviewer (emb_size=50)	0.847	0.925	0.873
word2vec + reviewer (emb_size=100)	0.850	0.948	0.928

Table B.10. *F1 scores (GloVe+Reviewer, spammer threshold=5)*

	YelpChi	YelpNYC	YelpZIP
glove-only	0.560	0.624	0.586
glove + reviewer (emb_size=3)	0.688	0.670	0.651
glove + reviewer (emb_size=10)	0.744	0.758	0.742
glove + reviewer (emb_size=20)	0.670	0.875	0.807
glove + reviewer (emb_size=50)	0.697	0.918	0.909
glove + reviewer (emb_size=100)	0.850	0.945	0.936

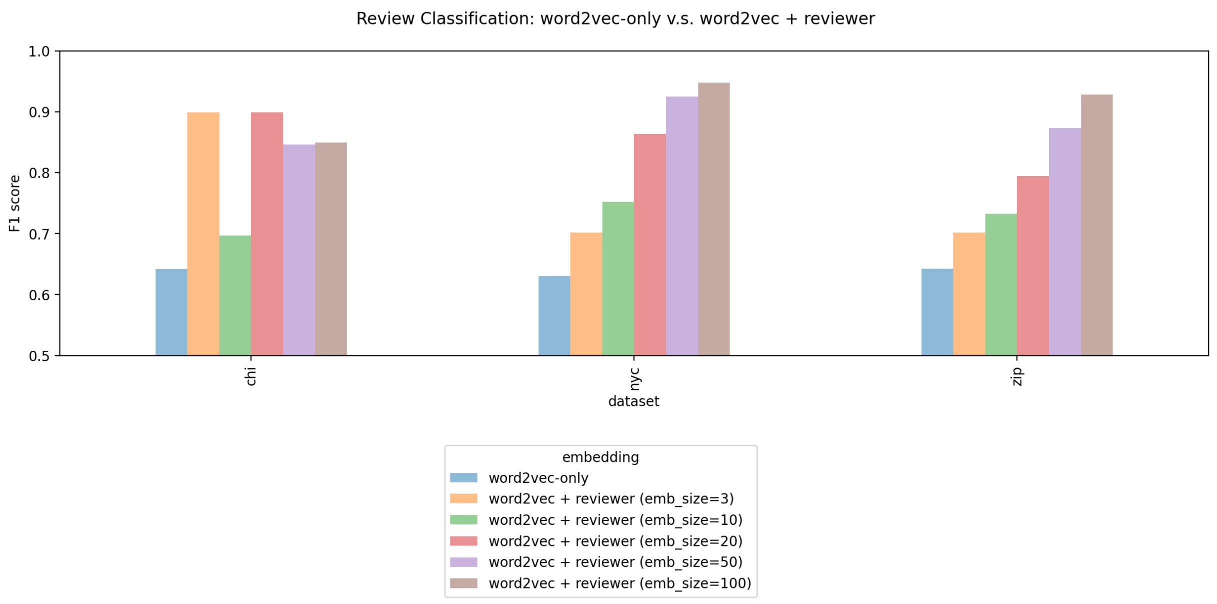


Figure B.9. Fake review classification (Word2Vec+Reviewer, spammer threshold=5)

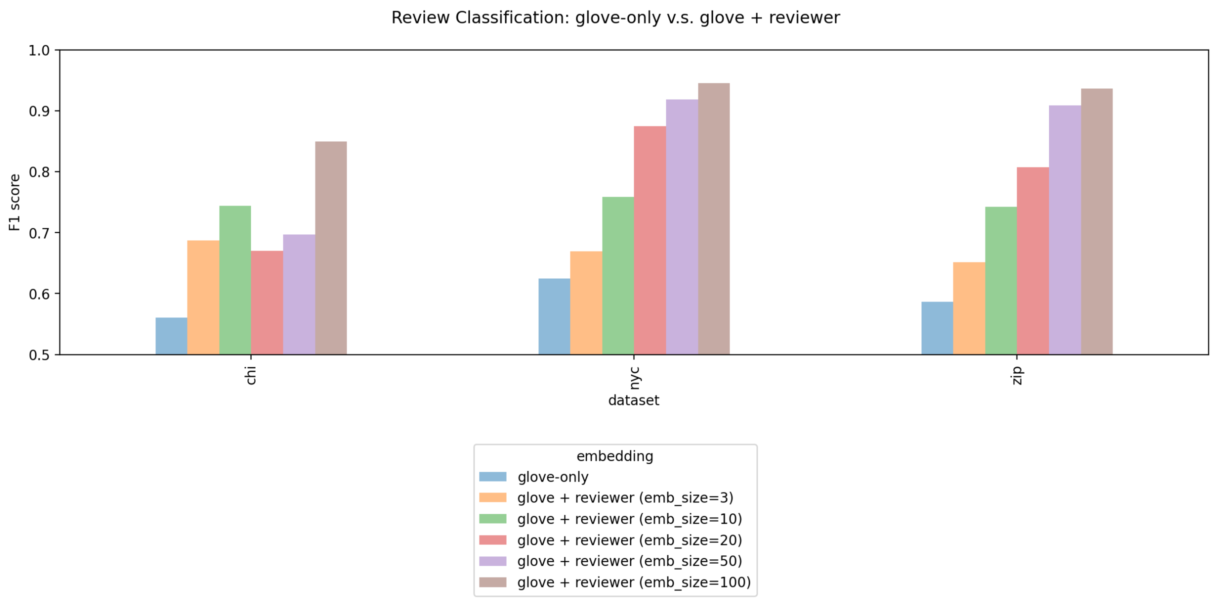


Figure B.10. Fake review classification (GloVe+Reviewer, spammer threshold=5)

APPENDIX C. DIFFERENT PERCENTAGES OF NODE LABELS

In this appendix, we include the complete experiment results of multi-view reviewer learning through MVE (Qu et al., 2017). The tool implemented by the authors of MVE has the functionality to take a list of node labels during the embedding learning. According to the experiments performed in Qu et al. (2017), the labeled data moderately increased the performance of node classification on their DBLP dataset (from .540 to .590 in micro-F1) and Youtube dataset (from .825 to .850 in AUC).

In this study, we tested whether different percentages of node labels used for MVE embedding learning impact the results of reviewer classification. First, we randomly sampled 10%, 20%, ..., 100% of node labels for MVE embedding learning. Second, we compared the testing F1 scores of the classification models trained on different percentage of node labels. According to the results listed below, there is not apparent trend that which percentage of node labels performs better for the Yelp datasets. Therefore, we chose 100% of node labels for our experiments.

Table C.1. *F1 scores of fake review classification (GloVe-only vs. GloVe+Reviewer)*

Percentage of node labels	YelpChi	YelpNYC	YelpZIP
10%	0.854	0.874	0.860
20%	0.857	0.871	0.903
30%	0.893	0.886	0.876
40%	0.856	0.867	0.891
50%	0.964	0.874	0.906
60%	0.856	0.896	0.887
70%	0.892	0.885	0.906
80%	0.892	0.859	0.881
90%	0.893	0.875	0.878
100%	0.929	0.878	0.903

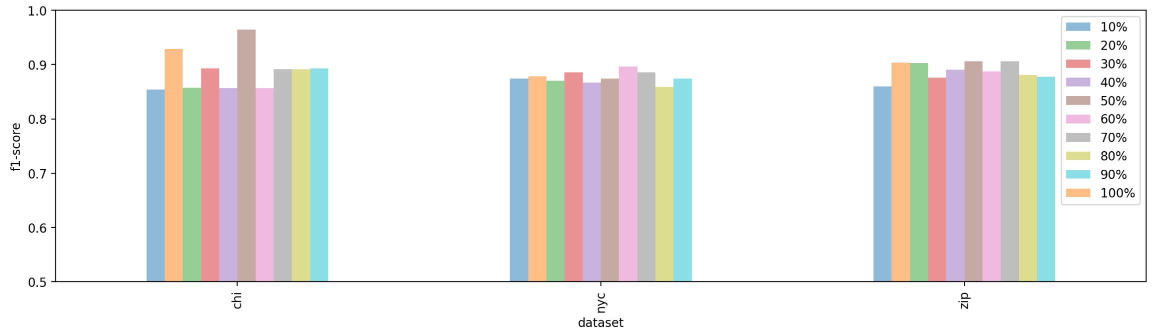


Figure C.1. Spammer classification with different percentages of node labels