FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Vandana Yadav

HYBRID RECOMMENDATION SYSTEM USING PRODUCT REVIEWS

Master's Thesis
Degree Programme in Computer Science and Engineering
June 2022

Yadav V. (2022) Hybrid Recommendation System Using Product Reviews. University of Oulu, Degree Programme in Computer Science and Engineering, 47 p.

ABSTRACT

Several businesses/smart applications rely on personalizing their services to adapt to the user's preferences. Personalized services are developed using recommendation systems based on user's feedback on products/services, needs, habits and social or demographic characteristics. Several businesses from e-commerce (suggesting users what to buy) to hospitality services (suggesting which hotel to book) focus on using recommendation systems to achieve a personalized experience for their users. Majority of recommendation systems make use of only product ratings shared by the users, this may pose challenges like sparsity of ratings. The wide availability of other attributes of products or users like textual product reviews provided by users or product descriptions in e-commerce and hospitality domains present a gold mine of additional personalising information with which to supplement their ratings based recommendation system.

Recommendation systems majorly involves two tasks: rating (predict ratings that user might assign to a product) and ranking (recommend products based on predicted rank scores) prediction tasks. In this thesis, we propose a novel hybrid recommendation system using the state-of-the-art DeepFM model which makes use of multiple textual features derived from product reviews particularly contextual sentence embedding vectors, average sentiment scores and linguistic cues such as presence/absence of negation in the product reviews in combination with ratings shared by users to enhance the prediction of the desired ratings or rank scores. We evaluated our system with commercial datasets from Amazon and Datafiniti for both tasks: predicting rating and recommendations based on predicted rank scores. We utilised different metrics for both types of tasks. From our evaluation we infer that using contextual sentence embedding vectors extracted using BERT, average sentiment scores and presence/absence of negation in the product reviews obtained from VADER, does impact the prediction of ratings and recommendations based on predicted scores of the recommendation system which only utilises product ratings as user preferences. Furthermore, we can conclude from our evaluation that (A) contextual embedding vectors and average sentiment scores together along with ratings in the proposed hybrid system improves prediction of desired ratings, (B) contextual embedding vectors, average sentiment scores and presence/absence of negation in the product reviews together along with ratings in the proposed hybrid system improves prediction of desired ratings as well, (C) contextual embedding vectors and average sentiment scores together along with ratings in the proposed hybrid system improves recommendations based on rank scores.

Keywords: personalizing, contextual embedding vectors, sentiment scores, recommendation system, negation

TABLE OF CONTENTS

FOREWORD

This thesis was written for my master's degree in Computer Science and Engineering with specialization in Artificial Intelligence at University of Oulu, Finland. The subject of my thesis was related to hybrid recommendation system using multiple information derived from textual product reviews like sentiments of the customer, contextual embedding and linguistic cue incorporating the meaning of the reviews for data from e-commerce and hospitality domains. Recommendation system is a very exciting and interesting topic of research area as it includes methods from several disciplines such as Machine Learning, Natural Language processing, Computational linguistics and many more for improving the recommendations thereby enhancing personalization of services for customers.

I would take this opportunity to thank my supervisor Dr. Mourad Oussalah. Prior to the thesis, I worked with him on "Negation Detection and Sentiment analysis utilizing restaurant and hotel-based reviews datasets" as a University Trainee in the university and was delighted when he agreed to supervise me in the thesis. I am really grateful for his guidance, suggestions and support in overall thesis process. Also, I am thankful to my family members for their constant support.

Oulu, June 23rd, 2022

Vandana Yadav

LIST OF ABBREVIATIONS AND SYMBOLS

ML	Machine Learning
DL	Deep Learning
AI	Artificial Intelligence
NLP	Natural Language Processing
CBF	Content-based filtering
CF	Collaborative Filtering
BERT	Bidirectional Encoder Representations from Transformers
PCA	Principal component analysis
VADER	Valence Aware Dictionary and sentiment Reasoner
RMSE	Root Mean Square Error
MAE	Mean Absolute Error
MAP	Mean Averaged Precision
DCG	Discounted Cumulative Gain
IDCG	Ideal Discounted Cumulative Gain
NDCG	Normalised Discounted Cumulative Gain
NER	Named-entity recognizer

1. INTRODUCTION

What is recommendation systems? A recommendation system is defined as a subclass of an information filtering system that seeks to predict the preference a user would give to a product [1]. Recommendation systems include two major tasks that are ranking and rating prediction tasks [2]. The rating task aims to predict what rating a user would give to a product. Once product ratings are predicted, then the system can recommend products probably based on the products with top predicted ratings. On the other hand, the ranking task aims to recommend the most relevant products to the users based on the rank scores predicted. Both prediction tasks can consider different information, and hence, both can be useful [2]. In the thesis, we are evaluating the proposed recommendation system for predicting desired ratings and providing recommendations based on predicted rank scores using one of the state-of-the-art algorithms on datasets from the e-commerce and hospitality domain.

Recommendation systems are of two types: personalized or generic. Personalized recommendation systems' main goal in applications is to predict what ratings users would give to items or recommend meaningful and relevant products or services to the collection of users by tailoring them to users' needs and interests. Generic recommendation systems are based on information about product popularity, user rankings, and any general information other than the user's profile.

Why is personalization important for recommender systems? Nowadays, smart applications have become ubiquitous in daily life as people depend on a wide range of services provided by them. It is important to make these services usable for a better user experience. One of the important aspects of user experience is the adaptation of services to the user's preferences. Personalization of services increases engagement, intuitiveness, and the number of customers thereby increasing the profit of the applications. Over the years, personalized recommendation systems have gained popularity in a variety of smart applications from e-commerce applications like Amazon and eBay to content delivery services like Netflix to travel booking applications like TripAdvisor. Products or items are general words in recommendation systems. The type of product or item varies based on the application's domain. For instance, on the Amazon application, different types of products can be recommended to users like books, music, video games, etc. On TripAdvisor, the product would be hotels that can be recommended to the users, and news platforms would have content as the product. These smart applications offer a huge number of products and choosing from this pool of products can be an overwhelming and confusing task for users [3] which may cause users to not use the application at all or just use it once and not come back to use it again. This would lead to a reduction in the number of customers in long run and thus, it can affect the application's (retailer's) profitability [4]. In such a case, to encourage customers/users to come back to access the application, recommending a certain number of products to customers that match their needs and interests can help them in making a choice and thus, could help in maintaining customer loyalty. Furthermore, the recommendation of products can help in making applications more intuitive and engaging. This can be addressed by highlighting new or upcoming products to users based on their behavior while they use the application such as whether users purchased the recommended products or not and how many times users browsed products other than recommended products. Thus,

personalized recommendation systems have become a vital part of smart applications and an unceasing research topic in both industry and academia.

What kind of information can be used in personalized recommendation systems?

Customers use information such as their interests, needs, and details about products to decide whether to buy products or not. Most of the applications are trying to integrate such information by utilising product details or product feedback shared by users to develop personalized recommendation systems. Nowadays, e-commerce, hospitality (like hotel booking) and other platforms offers customers/users to provide their feedback about products. Product feedback can be of different forms such as ratings, votes, emoticons, and textual reviews.

Ratings form of product feedback are mostly utilised in various recommendation system methods [5]. Further, textual reviews form of product feedback can be utilised in recommendation system methods. As different type of information from textual product reviews (like sentiments of user towards a product) can be extracted. Textual product reviews can provide either a positive, neutral, or negative opinion about the product that can help users in making decisions. This is where one of the important tasks of Natural Language Processing comes into the picture i.e., Sentiment analysis. Sentiment analysis deals with identifying/predicting sentiments of the individuals from textual data. The sentiment of the users can be utilized as additional information in developing personalized recommendation systems [6] [7]. It poses several challenges as text can include negations, slang, emoticons, and sometimes subtle hidden meanings. For instance, if a review on a game website contains the text, "I am not unhappy with this game, I have played it all day", the word "unhappy" has a negative connotation, but the review is very positive as the gamer continued playing it all day. Negations play a very important role in sentiment analysis as their presence can alter the meaning of the text. Several studies deal with identifying the negation cues (negators) and their scope. Negation cues such as not, never, and nothing or prefixed/suffixed (dis-, im-, and un-) words that have negative connotations e.g. disappointed affects parts of the sentence thereby affects overall sentiment value of the sentence. The scope of the negation cues defines what parts of the sentence have the effect of negation cues. By identifying the phrases/expressions which are affected by negation cues, polarity of those phrases can be updated which contributes to the overall sentiment value of the sentence can be improved. Using the improved sentiment values in recommendation systems can improve the results of the recommendation systems. Further, negations can be a independent piece of knowledge which can used in recommendation systems. Sentence embedding is the collective name for a set of techniques to map sentences to vectors of real numbers [8]. As textual data is ubiquitous, representing the meaning of a sentence has become important for many downstream tasks. Sentence embedding can be used in recommendation systems by either incorporating embedding into sentiment classification tasks to get sentiments of users about products or by using embedding representations of textual data (like product reviews) as an independent piece of knowledge about product or user. As we can see different type of information can be drawn from product reviews (like sentiment scores, embedding vectors). Therefore, we are interested in using product reviews from e-commerce and hospitality domain in the presented recommendation algorithm.

In the thesis, we propose a hybrid recommendation algorithm based on two types of tasks of recommendation system (rating and ranking prediction tasks) using information extracted from textual product reviews such as sentence embedding vectors, sentiment scores and presence of negations in the reviews.

1.1. Research Questions

In this thesis, we address the following research questions:

- **RQ1**: Does the sentence embedding of textual product reviews shared by the users as single piece of information impact the ranking (recommendations of relevant items) and rating prediction (to what extent the user likes an item) tasks of recommendation systems?
- **RQ2**: Do sentiments extracted from product reviews effect the ranking (recommendations of relevant items) and rating tasks of recommendation systems?
- **RQ3**: Does occurrence of negations in product reviews as an additional feature with other features affect the ranking (recommendation of relevant items) and rating (prediction of ratings that users would assign to items) tasks of recommendation systems?
- **RQ4**: Which combination of features efficiently outputs the recommendation list to users (ranking task) and predicted ratings that users would assign to products (rating task)?

1.2. Contributions

In our thesis, to the develop hybrid recommendation system, we adopted one of the recommendation algorithms called DeepFM for both type of recommendation tasks: prediction of desired ratings (rating prediction task) and providing recommendations based on predicted ranking scores (ranking prediction task) using several features from product reviews to address the above research questions. Our contributions were based on the e-commerce and hospitality domain datasets: Amazon Digital Music, Amazon Video Games and Datafiniti Hotel Reviews. Our contributions are as follows:

- We extract average sentiment scores of the product reviews (including sentiments or opinion of users about products) and presence/absence of negation in the product reviews to use them as an independent features in the recommendation algorithm. This is done using Valence Aware Dictionary and sEntiment Reasoner [9] which is a lexicon-based method
- We extract the contextual sentence embedding of textual reviews (incorporating the meaning of the product reviews) to utilise these embedding vectors as an independent feature in the recommendation algorithm. We do so using an attention-based ML method called bidirectional Encoder Representations

from Transformers (BERT) [10] along with Principal Component Analysis [11] method.

- We compare the use of different features from product reviews by considering them in different combinations aiming to obtain the best combination of features which would enhance the prediction of desired ratings and providing recommendations based on predicted ranking scores using the DeepFM [12] recommendation system.
- We show the comparison between the use of only contextual embedding or only sentiment scores of textual reviews as an additional feature along with ratings in the recommendation algorithm for both type of recommendation tasks: rating and ranking prediction tasks.
- Furthermore, we show the comparison between the use of only embedding of textual reviews and embedding of textual reviews along with negation labels (whether negations occur in the reviews or not) in the textual reviews in the recommendation algorithm for both type of recommendation tasks: rating and ranking prediction tasks.
- Also, we present an exploratory analysis of the datasets to inspect the behavior of negations in the product reviews classified as positive, neutral, and negative based on sentiment scores extracted from VADER.

1.3. Structure of Thesis

The rest of the thesis is organized as follows. Section 2 describes the related works in recommendation systems, sentiment analysis and linguistic cues. In Section 3 we focus on methods, concepts, and datasets. Section 4 provides results and discussions. Section 5 presents a summary of the thesis. The code developed as part of this thesis has been made available in a public GitHub repository. A URL link to this repository has been provided in the Appendix, placed in this thesis as Section 7.

2. RELATED WORK

2.1. Recommendation Systems

The methods involved in the recommendation systems can be classified as Collaborative Filtering (CF), Content-based filtering (CBF), and Hybrid-based filtering methods based on the input features and the domain where it is supposed to be utilized.

2.1.1. Content-Based Filtering Techniques

Content-based filtering methods make use of attributes of the product (like product descriptions) and user profile to identify products that users might be interested in and recommend them. User profiles can be created and updated using past purchase choices, product feedback from users, user interactions, or some preferences inputted by users [13] [14].

Several studies have emerged over the years to improve the quality, and performance of recommendation systems as well as develop recommendation systems for variety of domains such as e-commerce, social media, news, tourism (like hotel reservation) and entertainment (like recommending music, movies). Conventionally, CBF uses product titles or descriptions in text form. For example, in movie recommendation genre of the movie which is in text form are used in some of the recommendation methods by determining genre correlation [15]. As there is increase in user generated data (like reviews, tags, likes/dislikes), in recent years several methods/studies has been developed using product details along with user generated data (creates user profile).

Many methods are developed using machine learning or data mining. For example, to recommend articles to a reader, all articles already read by the reader are analyzed. Keywords from these articles can be extracted using one of the well-known text mining methods called TF-IDF. Weights of all keywords are integrated to get the representation of the article in form of a multi-dimensional vector. Further, clustering algorithms can be used to get the center of the vector that would represent the interests of the reader [16].

In CBF, users get the freedom to build their own profiles by using ratings or some form of system to input their preferences (e.g., like/dislike an item). This not only helps in developing a user profile but rather, helps in providing transparency to the users that how the recommendation system works. Most importantly, CBF is sufficient to predict/recommend items for new users who have not rated any item or given their preference in any manner as only item attributes can be used to produce the initial recommendations.

2.1.2. Collaborative Filtering Techniques

The collaborative filtering approach identifies products to recommend based on users with similar tastes/likings or identifies similar items that users rated in the past. Unlike CBF, CF uses product ratings shared by users. In CF, firstly it is assumed that users who have rated the same ratings to the products are likely to have similar preferences.

Secondly, it is assumed that products that are rated by users are supposed to be of users' liking, thus new or different products similar to the products rated by the user can be recommended to the user. Thus, the similar users or products are found by finding similarities of the users using ratings [17].

CF is categorized into two main groups: memory-based and model-based methods [18]. The memory-based algorithms use item ratings to determine the similarity between users or items. Whereas model-based algorithms develop models using different methods from data mining or machine learning [19] [20] [21].

Each of these methods has its own advantages and disadvantages. Memory-based methods are easy to implement and use, results are highly explainable and new data can be added easily. However memory-based methods do not handle data sparsity well, do not scale well with a high volume of data, and do not show high accuracy. Model-based methods consider matrix sparsity handling and scalability and show high accuracy. However, it takes a lot of time to create and update the model and the results are not well explainable [18] [22] [23]. Also, there are several hybrid methods combining features of both memory and model-based methods [23].

2.1.3. Hybrid-Based Filtering Techniques

Hybrid-based filtering methods of recommendation systems are a combination of two or more recommender systems to have better performance over CF and CBF methods. One of the ways to develop a hybrid-based filtering method is to combine both CF and CBF methods to include the advantages of both methods. As hybrid methods involve two or more recommendation methods, the process to combine the different methods is classified into several hybridization methods: weighted, switching, mixed, feature combination, cascade, feature augmentation, and meta-level [24]. The weighted process combines scores or votes from different recommendation methods to get an integrated recommendation list. The switching process chooses between recommendation methods based on the situation. Recommendations from all recommendation methods are listed at the same time in the mixed process. The feature combination process uses features from different recommendation data sources together in a single recommendation system. In the cascade process, the output of one recommender system is used in another to refine the final recommendation system. The feature augmentation technique, uses the output of the one recommendation system an input feature of another system. In the meta-level process, the model generated by one recommender system can be used in another system.

2.1.4. Challenges in Recommendation Systems

Different techniques/methods in the recommendation systems face several challenges which are described as follows:

A. **Sparsity**: Sparsity is one of the major problems and it arises due to the fact that users do not rate all the products. This problem is faced particularly in

collaborative filtering techniques as this technique in most situations depends on user-item rating matrix and thus ratings in the matrix are sparse [25].

B. **Scalability**: Scalabilty is one the attributes of the system to determine how smoothly system handles continuous growing data. Data is generated at a high rate on different smart applications and thus raises a question how well recommendation systems utilised in the smart applications can deal with enormously growing data on the application. In most cases, approximation methods are used to handle the scalability issue which improves the performance however, reduces the accuracy [25].

C. **Cold Start**: Cold start issue arises when a new user or product appears in the system, meaning system doesn't have information about user or product like user profile, user's past preferences or feedback of products. This makes it challenging to recommend list of products to the new user or new products to the existing user. Cold start problem is classified into three problems: new user problem, new item problem and new system problem [25]. Cold start problem is a major problem in CF techniques as there will be no rating available for the new user for any product or no product rating for new product. However, content-based filtering techniques can be useful in case of new products as these type of techniques doesn't depend on ratings shared by user rather depend on product attributes.

D. **Over Specialization**: Recommendation systems are based on the user's profile, their preferences, needs and interactions with the items which can raise over specialization issue thereby making the system very restrictive and not allowing to recommend upcoming items. [25]

E. **Shilling Attacks**: This issue arises when a malicious user or competitor enters false ratings of the items or preferences to affect the popularity. In this case it affects the quality of recommendations and thus affecting the performance and accuracy of the system. [26]

2.2. Sentiment Analysis

Sentiment analysis is a task of extracting the opinions or sentiments of individuals about specific entities using different types of data like images, videos, text or sound or gestures. In our thesis, we will be focusing on textual form of data.

2.2.1. Sentiment Analysis Methods

Sentiment Analysis methodologies can be broadly classified into three categories, namely lexicon based, machine learning based and hybrid approaches [27].

Lexicons methods assign polarity values to tokens, where tokens can be individual words of text. These values can be discrete , i.e. -1 , 0 and 1 for negative, neutral and positive respectively, or continuous from a range [-1, 1] based on the intensity of the

polarity. Further, the scores for each token are aggregated by summing the positive, neutral and negative scores separately. Finally, we arrive at the overall polarity of the text based on the maximum scores. A key advantage of lexicon based methods is that it does not need large amount of training data as is often the case with machine learning methods. However, the polarity assignment may not be domain agnostic, as the same word may have different connotations and thus polarity. For instance, in a product review of a network connection, a "huge lag" is a negative review but one that says of a movie, "huge queues for tickets" is a positive one. Lexicon based methods can be dictionary based, or corpus based. The dictionary approach creates an initial dictionary of a few words and uses an online dictionary to expand that dictionary by incorporating synonyms and antonyms of those words in a recursive fashion until no new words can be added.

Corpus based approaches include statistical and semantic methods. Statistical approaches look for co occurrence patterns so that if a word appears in positive text more that in negative ones, it is more likely to be positive and vice versa. Thus, for each new token, the orientation is calculated by counting the frequency of occurrence alongside other, previously calculated tokens.

The semantic approach calculates a similarity score between words, usually using Wordnet [28] to find synonyms and antonyms in order to build a lexicon model for use in sentiment analysis.

Over the years, incorporation of sentiment analysis into recommendation systems has gained popularity specifically where product or service feedback is shared by customers. It is majorly utilised to offer solution to specific challenges faced by in recommendation systems like sparsity of common ratings or cold-start problem (it occurs when new user or product are introduced in the system) [29] [30].

Sentiment analysis results can be used in different ways in recommendation models. Its output can be used as a way to draw out user preferences or develop user profile or to enhance CBF and CF methods of recommendation system. Also, its output can be combined with output of recommendation system to further improve the predictions. Further, its output can also serve as validation tool for a recommendation system.

The major advantage of using a lexicon-based approach is that sentiment classification is explainable as it is based on a set of rules and a dictionary of sentiment-carrying words, expressions, or entities (like LOL and meh or smileys) defined by humans, making them more reliable. Whereas machine learning approaches are like a black box and require additional techniques (Explainable AI methods) to interpret what linguistics features are considered while modeling. The ML methods for classification problems learn all features automatically through optimization. However, ML methods are trained on the labeled data with respect to a specific domain, at times this method would not work with another domain [31].

VADER is one of the popular lexicon-based approaches as its effectiveness was compared with several state-of-practice benchmarks like sentiWordNet [32], Linguistic Inquiry and Word Count [33] and General Inquirer [34] with data from social media (Twitter), entertainment (Movie), and e-commerce (Amazon Product reviews) [9]. The dictionary includes sentiment-carrying words or expressions that are more sensitive to social media and microblog contexts. VADER has been utilized in several studies for sentiment analysis task with respect to variety of domains. It was employed in various studies to obtain the sentiments expressed in Twitter data for variety of purposes [35]

[36] [37]. As well as some studies use outputs of VADER to label textual data with sentiment information (positive/negative) where no ratings or sentiment information is available for evaluation. For instance, a study focused on using VADER's output for labeling email conservation prior to modeling the LinearSVM model with email content with sentiment labels to predict sentiment classification of emails [38].

As we are interested in product reviews from e-commerce and tourism domains in our recommendation model and considering the advantages of the lexicon-based approach and VADER's sensitivity towards social media and microblog context, makes VADER a highly suitable and promising method to employ for extracting sentiment scores from textual product reviews and using it in the proposed recommendation algorithm (DeepFM model).

2.3. Linguistic Cues

Linguistics cues are structured information which can be drawn from unstructured text. It usually involves grammatical structures or other linguistic attributes (like contractions, negations, degree modifiers). Text Mining is a process that involves extraction of interesting or non-trivial patterns such as linguistics cues from unstructured textual resources [39]. It comprises of variety of methods such as Information Retrieval, Information Extraction, Natural Language Processing, Machine Learning, and Data Mining to elicit linguistic cues from free textual data [40].

One of the most popular sub-tasks of the Information Retrieval is Named-entity recognition (NER) which aims to extract entities and categorize them as specific entities such as person, organization, language, or location from the text. For instance, NER identifies "Finland" as a location category and "Finnish" as a language category. These entities can be used as linguistic cues for domain-specific tasks to draw out further inferences in the dataset considered for analysis or use for other tasks like aspect-based sentiment analysis where entities are extracted and assigned sentiments instead of assigning sentiments to the whole text. This can help in identifying what aspects of the text are mostly classified as positive or negative.

NLP is set of techniques of text mining which comprises various methods from several disciplines to extract linguistics features. It can be utilized to extract some grammatical features from the text. For example, Part-of-Speech tagging, are used to tag parts of speech like noun, adjective, and verbs to each token of the text. These parts of speech information can be utilized as a linguistic feature in different tasks where semantic analysis is required. In addition to parts of speech, other linguistic features like negation also contribute to semantic analysis as it alters the meaning of the sentence. These linguistic cues contribute to the meaning of the sentence. Several NLP and ML methods extract negations to use in tasks like sentiment analysis. Sentiment analysis is most popular which involves negation detection methods in its process. A study presents a set of rules to compute sentiment when negation is present [41]. Some studies show that information about the scope of negation is advantageous in prediction of sentiments [42] [43]. Negation detection approaches ranges from rule-based approaches to machine learning and neural networks. Negex is one of the rule-based method that detects negation cues based on simple regular expression algorithm specifically to determine whether a disease or finding are found in clinical texts [44].

The authors of DEEPEN proposed an improved version of NegEx using dependency parsing [45]. As well as another author utilised dependency parsing to improve Mayo clinic's clinical Text Analysis and Knowledge Extraction System (cTAKES) negation annotator [46]. NegBERT presents a negation detection and scope resolution method based on transfer-learning approach of Machine Learning field [47].

As our thesis focus is to use textual data from product feedback in recommendation system. We try to incorporate as much semantic information of the textual reviews by using linguistic cues such as presence or absence of negation in combination with contextual embedding of the textual reviews to verify whether linguistic cue: presence/absence of negation impact the prediction of the recommendation model.

3. IMPLEMENTATION

3.1. Dataset

For evaluation of DeepFM algorithm, we have used three datasets: Datafiniti Hotel reviews, Amazon Digital Music and Amazon Video Games datasets. **Datafiniti Hotel reviews** dataset is provided by Datafiniti's business database [48]. Datafiniti collect data from different websites to create standardized database of business, products and property information. Here, we have utilised Hotel reviews dataset which has around 10000 records of about 1400 hotels and includes information like hotel location, name, username, rating, textual review and more.

Digital Music and **Video Games** datasets are part of Amazon Product dataset [49] which is a collection of sub datasets for different types of products like books, digital music, video games, toys, etc. It contains product reviews such as ratings (from 1 to 5), textual reviews and votes as well as metadata like product descriptions, category, price and brand from Amazon that includes 142.8 million reviews collected from May 1996 - July 2014. For our evaluation, we selected 5-core version of Digital Music and Video Games sub datasets. Digital Music dataset contains 64,706 reviews and Video Games dataset 231,780 reviews.

3.2. Methodology

In this section we describe the overall method applied to develop the proposed hybrid recommendation system. We cleaned, pre-processed, extracted and prepared features/predictors (namely contextual embedding, sentiment score and presence/absence of negation in the product reviews) from product reviews using different methods. Then, utilised the features/predictors in different combinations in the recommendation algorithms for predicting desired ratings and providing recommendations based on predicted rank scores. Here, we adopted one of the popular state-of-the-art deep learning models called DeepFM model as recommendation algorithm [12]. Figure 1 shows the overall method adopted to build the proposed hybrid recommendation system in the thesis.

As a first step, we cleaned and pre-processed the datasets by removing samples where ratings or text reviews were missing and those sample where text reviews consist of only non English entries. Additionally, in this stage, we removed numeric entries from the text reviews and converted the user and product Ids from alphanumeric to numeric values. Table 1 shows unique users and items and total records in each dataset after cleaning and pre-processing steps. For instance, the cleaned and pre-processed Amazon Digital Music dataset includes information like ratings and textual reviews of 5541 users and 3568 items (Music) and have 64705 total number of records/samples of different user-item pairs.

After this, we proceeded to extract and prepare features for the recommendation algorithm. First part of preparing feature involves extracting contextual sentence embedding of product reviews using Bidirectional Encoding (BERT) model [10] and Python Transformer library [50]. For each user-item pair in the dataset, embedding of review was extracted. Dimension of each embedding was of [1, 768] which is very

Table 1. Details about the datasets after data pre-processing

Datasets	Unique users	Unique items	Total records	Average rating per item	Standard deviation
Datafiniti Hotel reviews	6941	1670	9998	3.98	1.17
Amazon Digital Music	5541	3568	64705	4.22	1.08
Amazon Video Games	24303	10672	231736	4.08	1.28

high and would be challenging while training the final recommendation model. So, we proceeded to reduce the dimension of the each embedding vector using Principal Component Analysis (PCA) model. Performed PCA using scikit-learn library [11]. Each high dimensional vector representing product review was reduced to number of PCA components where explained variance of model was equal to 90%. We choose explained variance as 90% because explained variance is the part of model's overall variance that is explained by factors that are actually present in the data and not by error variance [51]. So, the number of components for 90% explained variance would result in strongest factors explaining the variance of the model and thus, it could help in improving the predictions [52]. The extracted number of PCA components according to explained variance for Datafiniti Hotel reviews, Amazon Digital Music, and Amazon Video Games datasets were equal to 16, 13, and 12 respectively and thus, these were reduced and final dimensions of product reviews' embedding vector for respective datasets. In this way lower dimension of embedding vector of product reviews was obtained.

In second part of preparing feature we extracted average sentiment scores of product reviews from Valence Aware Dictionary and sentiment Reasoner (VADER) [9]. It provides sentiment scores in four parts: positive, neutral, negative and compound sentiment scores. First three sentiment scores describes how much positive, neutral or negative a text is (these all adds up to 1 or near to it with float operation) and compound score describe overall sentiment value. Here, we used compound sentiment score provided by VADER. Compound sentiment score ranges from -1 to 1. VADER is said to provide better sentiment scores at sentence level than to consider whole text at once. Product reviews consisted of several sentences. So, we determined sentiment score of each sentence of the review and then took average of sentiment scores of the review. In this way we calculated sentiment score for all reviews.

While determining sentiment score, VADER consider negations like "not great" found in the sentence. We extracted number of negations found in the product reviews and then used it to create/additional a feature "whether negation occurs in the product review or not". If number of negations found was more than 0, the feature was labelled as 1 otherwise 0. In the thesis we would refer this feature also as negation label. At this stage, we have successfully extracted following features from product reviews for all datasets using BERT and VADER methods: contextual embedding vectors, sentiment scores, and presence/absence of negation given by negation label. As ratings are shared by users these are considered to indicate user preference and thus can be used as feature. In the recommendation model (DeepFM), we used ratings along with additional information by aggregating the extracted features from product reviews

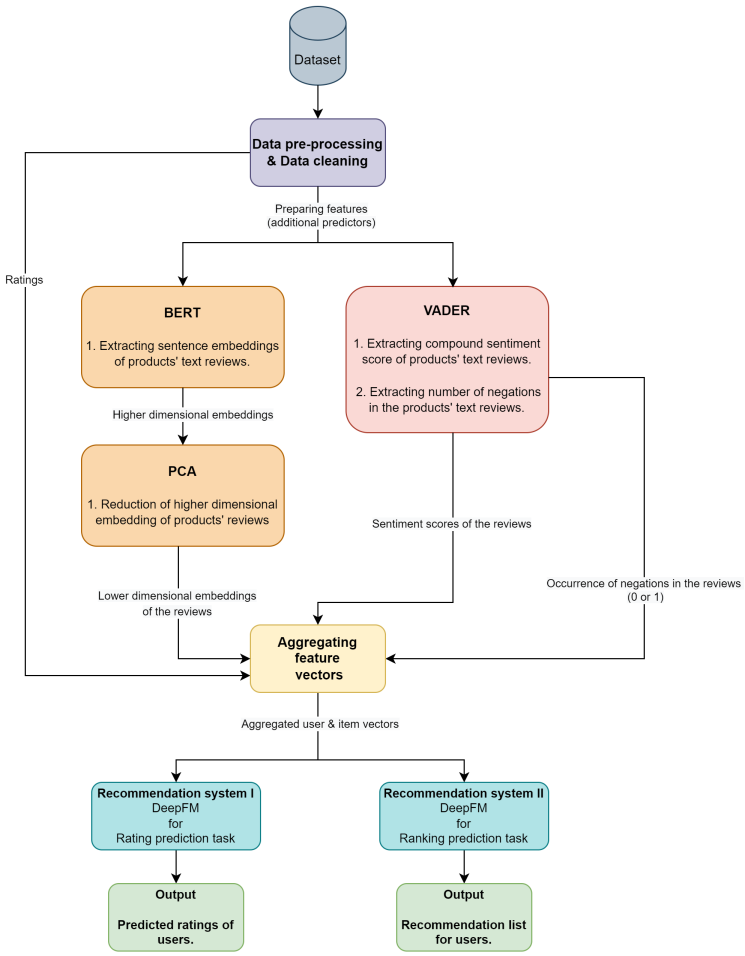

Figure 1. Overall method.

in different combinations to evaluate which combination of features predicts desired ratings or provides recommendations based on predicted rank scores better. Table 2 lists additional features utilised in the recommendation algorithm for predicting desired ratings and providing recommendations based on predicted rank scores.

Table 2. Shows which additional features in combination with ratings are used in different predictor combinations

P1	no additional feature used.
P2	Contextual embedding vectors
P3	Sentiment scores of the reviews.
P4	Contextual embedding vectors & Sentiment scores of the reviews.
P5	Contextual embedding vectors, Sentiment scores & presence/absence of negation in product reviews
P6	Contextual embedding vectors & presence/absence of negation in product reviews

Table 3. Shows which features are included in which predictor combinations

Combination name	Ratings	Review embedding	Sentiment scores	Negation/ label
P1	Yes	No	No	No
P2	Yes	Yes	No	No
P3	Yes	No	Yes	No
P4	Yes	Yes	Yes	No
P5	Yes	Yes	Yes	Yes
P6	Yes	Yes	No	Yes

We have used two types of recommendation algorithm based on type of task: rating and ranking prediction tasks. For both tasks we used DeepFM algorithm and utilised its one of the public/open-source implementations called LibRec [53].

3.3. Concepts

In this section, we describe the fundamentals of different methods presented in this thesis. Our thesis involves usage of NLP with Machine Learning and Rule-based approaches.

NLP is sub-field of computational linguistics and Artificial Intelligence (AI) which focuses on giving computers ability to analyze, understand and draw meaning from text or speech in similar way humans do [54]. NLP involves variety of tasks to analyze and interpret the language either in form of text or speech such as Word Sense Disambiguation, Sentiment Analysis, Named-entity Recognition, Topic Modeling, Speech Recognition, etc. These tasks can be solved using either Machine Learning or Rule-based approaches. Machine Learning is also a sub-field of AI which focus on system's ability to learn and adapt to new data without human intervention by training on large corpus to make decisions or predictions in variety of downstream tasks [55]. Further ML is classified into several categories such as Supervised learning, Unsupervised learning, Reinforcement learning and Deep learning according to the nature of downstream task considered [55].

- **Supervised learning:** The algorithms are trained and evaluated using large datasets. Supervised learning algorithms require datasets where desired results or outcomes (labels) are available. Supervised learning is popular in classification and regression tasks provided where label information is available.
- **Unsupervised learning:** Such algorithms are trained and evaluated on datasets which doesn't consists of label information.
- **Reinforcement learning:** Such algorithms are based on choosing best approach called policies. These policies are evaluated by rewarding expected results and penalizing undesired results. The approach which receives most awards are chosen by the model.

- **Deep learning:** Deep learning techniques are used where conceptual features are needed to be extracted from complex and unstructured data. DL techniques involves usage of neural networks. Neural networks are designed to imitate human brain so that computers can be trained accordingly.

Rule-based approaches focuses on set of of rules and facts defined and programmed by the humans to generate predefined outputs [56]. In NLP domain, rule-based systems focuses on developing rules based on linguistic structures and features that in similar manner matches the building grammar rules and structures by humans [57].

3.3.1. DeepFM

DeepFM [12] is a deep learning model that is a combination two different types of algorithms called Deep Neural Networks (DNN) and Factorization Machines (FM) by incorporating benefits of both algorithms.

Figure 2 shows the DeepFM model architecture. It shows that DeepFm has two embed layers first and second order embedded layers, FM layer, deep component (hidden layers) and output layer combines outputs of both FM and deep parts.

FM component: As part of FM component [58] and [59], first order embedded layer collects overall attributes of individual features (called as order-1 features) and then, in the FM layer, the dot product of order-1 features and weights vector are taken which is represented by the Addition unit. On the other hand, second order is a shared embedded layer for both deep and FM components. In the FM layer, collection of Inner Product units takes pairs of latent embedding vectors which addresses second order feature interactions. The output of FM component is summation of Addition and all Inner Product units.

Deep component: The deep component is a feed-forward neural network and helps in learning high-order feature interactions. Before feeding the input vectors to the hidden layers of the neural network, input vectors are compressed to low-dimensional dense vector by the dense embedding layer (shared embed layer) [59].

The major advantage of the DeepFM model are that it can be trained efficiently as well as it doesn't require expertise in feature engineering of raw features as both algorithms have shared inputs at different parts of the model [12]. The final output of DeepFM model integrates outputs from both DNN and FM algorithms and the formula of DeepFM model is:

$$\hat{y} = sigmoid(y_{FM} + y_{DNN}), \tag{1}$$

where $\hat{y}$ is the final output (predictions), and y_{FM}, y_{DNN} are the outputs for FM and Deep Neural Network component.

In our method, we have used DeepFM for two tasks of recommendation systems: rating and ranking prediction tasks. We utilise public implementation of DeepFM algorithm called Librec [53] for both tasks. For rating prediction task, for our implementation, we set number of epochs equal to 15 and learning rate as 0.001. We chose neural network hidden states as as [128, 64, 32], and the embedding size as 16. For ranking prediction task, we set number of epochs equal to 5 and set low learning

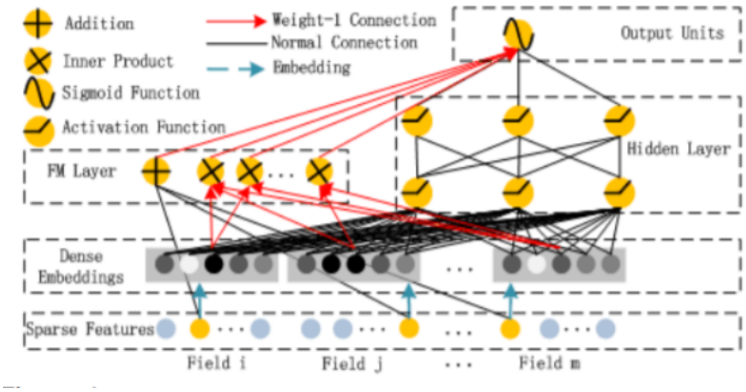

Figure 2. DeepFM architecture [12].

rate as 0.00001 to avoid over-fitting of the model. We chose neural network hidden states as as [128, 64, 32, 16], and the embedding size as 16.

3.3.2. BERT

Bidirectional Encoder Representations from Transformers (BERT) [10] is a language deep learning model developed by Google AI team in 2018 has gained lot of popularity in NLP domain. As it helps computers to understand ambiguous words/phrases by checking text in both directions to establish context of the words/phrases. Whereas, other language models like Word2vec or GloVe are lacking when interpreting context and ambiguous words/phrases. Moreover, it has achieved high performance in several NLP tasks such as semantic role labeling, sentiment analysis, sentence classification and disambiguation of words or phrases with multiple meanings.

Its architecture is developed based on the Transformer. BERT is a pre-trained model on a huge corpus of unlabelled text including the entire Wikipedia and book corpus (800 million words). It is trained based on two methods: Masked Language Model (MLM) and Next Prediction Sentence. In MLM, 15% of words are masked and then the model is expected to predict the masked words and in Next Prediction Sentence, the model is expected to predict whether the first sentence comes after the second sentence or before the second sentence. Bert model has two types of pre-trained models Base and Large, both take input of 512 dimensions shown in Figure 3. Further, both of these models have two variants i.e., case and uncased.

As BERT is pre-trained model on huge corpus and used in several NLP tasks, we selected it in our method, to extract contextual embedding of product reviews shared by users for all datasets. From different variants of BERT model, we chose the BERT Base Uncased pre-trained model and Figure 4 shows BERT Base Uncased model. It comprises of 12 transformer blocks, 12 attention heads, 110 million parameters, and has an output size of 768-dimensions.

Figure 5 shows the architecture for extraction of contextual sentence embedding of a sentence. It takes input sentences, adds CLS and SEP tokens at beginning and end of sentence and then tokenized input sentence is passed into the pre-trained model. We can use outputs of different layers as sentence embedding. Here, we chose to consider

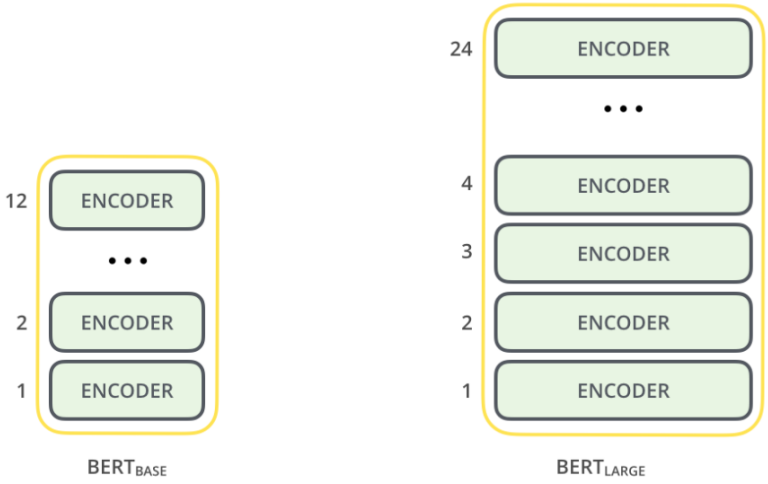

Figure 3. BERT Base and Large [60].

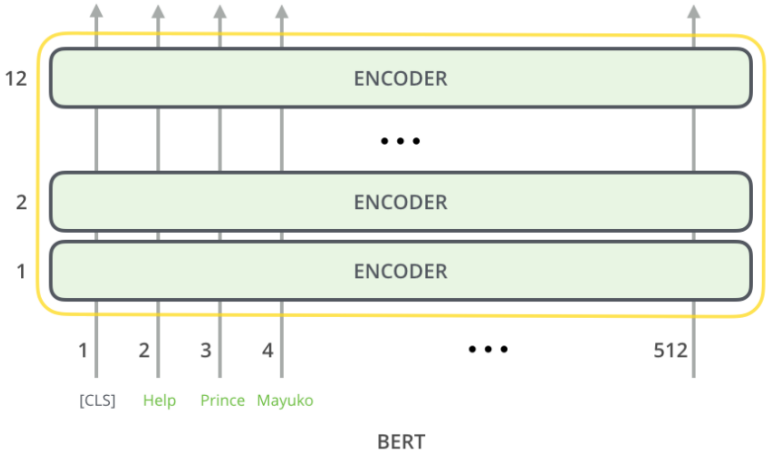

Figure 4. BERT Base Uncased model variant of BERT model [60].

outputs from final layer. As BERT authors evaluated by feeding outputs (embedding) from different layers as input feature to a BiLSTM for named entity recognition task and observed that for final layer output Dev F1 score was 94.4% which is quite high [10]. Final layer of the model outputs a vector of size of [batch, maximum length of token, hidden states] (hidden states equal to 768 in BERT Base). Each token of the sentence has its output size of 768. First token in the sequence is CLS that includes the entire context of the sentence and thus can be used as final sentence embedding of the sentence.

3.3.3. Valence Aware Dictionary and SEntiment Reasoner

VADER [9] is a open-source rule-based engine that is used for sentiment analysis of unstructured or structured text. It relies on use of gold standard dictionary of lexical features along with their sentiment intensities (called as valence scores). In addition

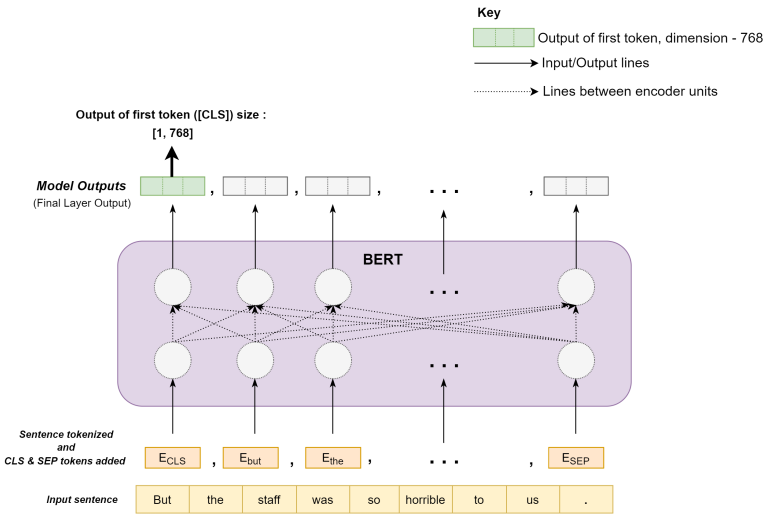

Figure 5. BERT Base Uncased model architecture for extraction of sentence embedding [60].

to lexicon, it uses five generalized heuristics related to grammatical and syntactical structures and attributes that humans consider while expressing the sentiment of a text. This has shown improvement in accuracy when compared with individual human raters for correctly classifying sentiments as positive, neutral, or negative for different types of data of social media domain such as tweets, Amazon product reviews, NY Times editorial and movie reviews. It has several advantages over different sentiment analysis engines based on ML models. It doesn't require extensive training of large datasets and secondly, the results can be easily explained as the rules and lexicon are developed by humans whereas ML models are like a black box and their results are challenging to explain.

The lexicon/dictionary maps words (like terrible and awesome) and several popular and common lexical features of microblogs expressing specific sentiment such as emotions (like :) and :D), initialism and acronyms (like LOL and OMG) and slangs (like Nah and meh). As these are found commonly in unstructured text, it becomes easier to assign polarity to such entities when they are available in lexicon. Valence score of each type of entity are in range of -4 to +4.

As mentioned earlier it considers five generalized heuristics to calculate sentiment scores [61]. The five rules are as follows:

Catching Polarity Negation: After identifying the word or lexical feature that holds specific sentiment, it checks in sequence three entities prior to the sentiment-laden lexical feature for negations (from list of predefined negations). If negations are found in those three words, valence score is updated. The list contains typical negation like "not good" as well as contractions like "isn't good". For example, a sentence "The movie wasn't good.", in this negation is "wasn't" before a sentiment-laden word "good", so valence score is updated.

Punctuation: Punctuation is particularly used to increase the sentiment intensity of the phrase. The most popular punctuation is exclamation mark which raises the intensity without altering the meaning of the phrase. For example, a sentence without

exclamation mark: "The movie was good." is definitely positive however, sentence with exclamation mark: "The movie was good!!" has more intensity.

Capitalization: Capitalization is similar to punctuation rule, words or phrases in all caps increases the relevance of the sentiment-laden word and thus, when all-caps words are found their sentiment intensity is increased.

Degree modifiers: Degree modifiers also, alters the sentiment of the sentence. Degree modifiers either intensify the sentiments called boosters (like "very") or weakens the sentiment values called dampeners (like "kind of"). Also, there degree modifiers in form slangs (like "friggin", "kinda") which either boosts or dampens the sentiment of the sentence.

Polarity shift due to Conjunctions: Conjunctions also, play an important role in affecting the sentiment of the sentence. Conjunctions like "but" shifts to the dominance of polarity after conjunction "but". So, latter part of sentence would contribute more in calculation of sentiment of the sentence.

As it particularly adapts well with social media content, we have used VADER for extracting sentiment scores of product reviews of the datasets which we have adopted for evaluation. Product reviews consist of several sentences, so we considered to find average sentiment score of review by averaging the sentiment score of all sentences in the reviews. After extracting valence scores from dictionary and altering them based on the rules mentioned above, finally scores are normalised in range of -1 to +1 (from -4 to +4) range which are called compound sentiment scores. It also, provides three sentiment scores defining the intensity of each type of sentiment (positive, neutral and negative). These scores sums up to 1 or near to 1 with float operation. The authors of VADER considered following threshold to classify the sentence as positive, neutral and negative based on compound sentiment scores: compound score greater than equal to 0.05 as positive sentiment, compound score greater than -0.05 and less than 0.05 as neutral sentiment and compound score less than equal to -0.05 as negative sentiment.

In our method, we were interested in finding whether negations are present in the product reviews given by the users. VADER implementation, considers/handles negations found in the text by altering the valence scores accordingly. As we saw in negations rule that valence score is updated when negation is found in the sentence by checking the trigram prior to sentiment-laden lexicon feature. So, at this check we extracted the number of negations found in a sentence and following it by consolidating the number of negations of sentences of the review. After that we used number of negations to build a feature "whether negations occur in the product reviews or not" and labeled the feature with 0 and 1. 0 meaning negations do not occur in the reviews and 1 meaning negations so occur in the sentence.

3.4. Evaluation Metrics

In this stage, we evaluated the proposed algorithm for all the three datasets using ratings and other additional features for rating and ranking prediction tasks. All datasets were split into two sets 80% and 20% of the whole dataset as training and testing sets respectively. Training sets were used to train and develop the model whereas testing set was used for evaluation using the metrics presented in this section.

For both tasks, we evaluated the algorithm with different combination of predictors as mentioned in Table 3 using evaluation metrics utilised in LibRec implementation [53]. In LibRec implementation, for rating prediction task, predicted ratings of the products are used for evaluation and compared with the ratings in the dataset. On the other hand in LibRec implementation, for ranking prediction task, scores for all products are predicted and these scores are further used to build recommendation list for all users which are used for evaluation. These recommendation lists are compared with desired (ground truth) recommendation lists of the users. This ground truth was created by determining items consumed by each user [53].

For rating prediction task, we chose following metrics:

Root Mean Square Error (RMSE): RMSE is a quadratic score which also measures the mean magnitude of the errors in predictions. Instead of taking absolute of difference between predicted and actual observation, in this metric, difference value is squared and then averaged over test set. Further, square root of averaged squared value is taken to calculate error value [62]. This is defined in the following formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y_j})^2} \tag{2}$$

Mean absolute error (MAE): MAE is a linear score which measures the magnitude of errors in the set of predicted observations. It is calculated by taking average over absolute difference between predicted and actual observations of test set. By taking absolute of difference, each difference have equal weights [62].

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y_j}| \tag{3}$$

RMSE scores are always greater than or equal to MAE scores representing the variance in the individual errors in the sample. If the difference between RMSE and MAE is increases, variance in the errors increases whereas if difference is zero then all error are of same magnitude. Both RMSE and MAE has their own advantages. As the difference is squared before averaging, RMSE shares comparatively high weights to large errors and thus it could be used in situations where large errors are specially unsuited. However, in RMSE, taking square root of averaged squared difference raises some implications and thus, in such situations MAE is more suitable [62]. Both RMSE and MAE can range from 0 to infinity and lower values of both represent better a model adapts to the data.

For ranking prediction task, we chose following metrics:

The most important aspect of the recommendation systems of ranking type task is recommending top-N products to the users. So, it is more reasonable to calculate evaluation metrics for N products rather than for all products. Therefore, concept of precision, recall, MAP and NDCG at N arises, where N is integer value that can be selected by the user to get top N recommendations. These metrics are explained below:

Precision@N: Precision is measure of how many relevant documents are in all retrieved documents. So, Precision@N would measure how many relevant items are

in the top N recommendations [63]. In following manner, precision@N is computed here [63]:

$$precision@N = \frac{N\ recommendations \cap Desired\ recommendation}{N} \tag{4}$$

Recall@N: Recall is measure of how many relevant documents are captured from all relevant documents. So, Recall@N would measure how many relevant items are captured in the recommendations among all relevant items. Basically, it focuses on coverage of relevancy of the top N recommendations [63]. In following manner, recall@N is computed here [63]:

$$Recall@N = \frac{N\ recommendations \cap Desired\ recommendations}{size\ of\ desired\ recommendation} \tag{5}$$

MAP@N: The Mean Average Precision@N measures the average precision@N (AP@N) averaged over all users. AP@N is given by following formula [64]:

$$AP = \frac{1}{n}\sum_{k=1}^{n} p(k)rel(k) \tag{6}$$

where $p(k)$ denotes precision@k, n is number of items in the recommendation list, $rel(k)$ is an indicator function, which is 1 if rank k item is relevant, 0 otherwise.

MAP is average of AP over all users and is represented in formula as [63]:

$$MAP = \frac{1}{|U_{all}|}\sum_{u=1}^{|U_{all}|} AP(u) \tag{7}$$

NDCG@N: NDCG stands for Normalized Discounted Cumulative Gain and is defined as the Discounted Cumulative Gain value that is normalized by Ideal Discounted Cumulative Gain such that its value is always ranges between 0 and 1. [63].

$$NDCG = \frac{DCG(N)}{IDCG(N)}, \tag{8}$$

DCG and IDCG formula is given by [63]:

$$DCG(N) = \sum_{i=1}^{N} \frac{G_i}{\log_2(i+1)} \tag{9}$$

where G_i denotes gains up to a position N.

IDCG is the DCG value for the most ideal ranking, which is ranking the products top down based on their relevance up to position N.

$$IDCG(N) = \sum_{i=1}^{|I(N)|} \frac{G_i}{\log_2(i+1)} \tag{10}$$

where I(N) denotes ideal list of items upto position N, $|I(N)| = N$. [63]

4. RESULTS & DISCUSSIONS

In this chapter, we present results and their analysis of three experimental phases in our thesis using the concepts, methodology and evaluation metrics mentioned in previous chapter.

4.1. Experiment 1: Exploratory Data Analysis

Initially, we started with exploratory analysis of the cleaned and pre-processed datasets considered in the thesis.

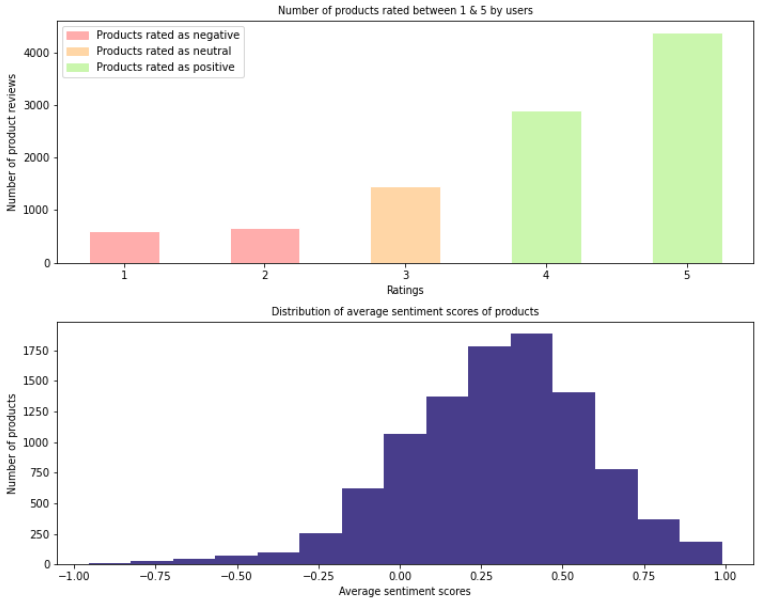

Figure 6. Distribution of product ratings and average sentiment scores for Datafiniti Hotel Reviews.

In our thesis, we are interested in product feedback namely ratings and textual product reviews. In all datasets, each user has provided ratings and textual reviews for different products. The ratings are integer values which measure likeability of the product by user. Ratings ranges from 1 to 5 where 1 being the most disliked product to rating equal to 5 being the most liked product. Whereas, the product reviews are in form of text and length of text varies for each product. Also, for each user-product pair, we provided average sentiment score and number of negations extracted from VADER. The average sentiment scores ranges from -1 (most disliked) to 1 (most liked).

For the analysis, we classified product rating equal to 1 and 2 as negative rated products (product least liked by the user), rating equal to 3 as neutral rated reviews (product not extremely liked or disliked by the user), and rating equal to 4 and 5 (product liked by the user) as positive rated products. On the other hand, for average sentiment scores, we considered sentiment scores greater than 0.05 as positive, sentiment scores less than 0.05 and greater than -0.05 as neutral, and sentiment scores less than and equal to -0.05 as negative.

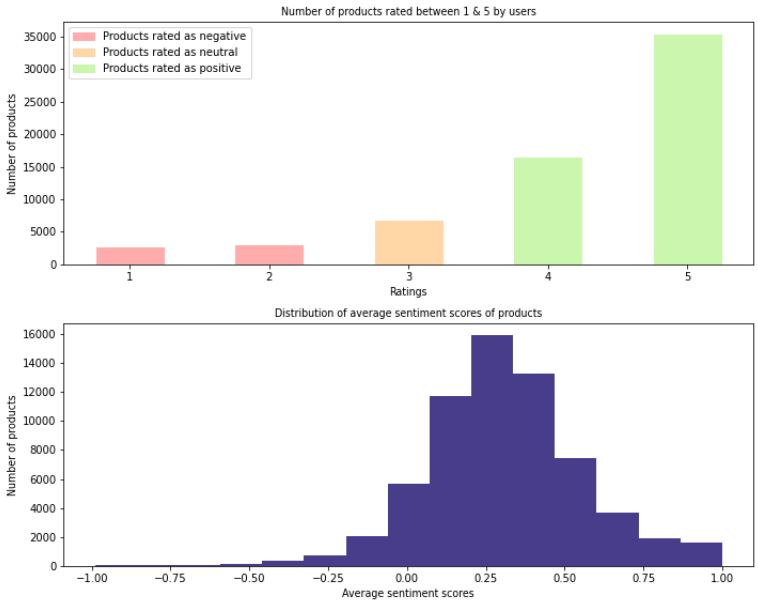

Figure 7. Distribution of product ratings and average sentiment scores for Amazon Digital Music.

We began our analysis by looking into how ratings and average sentiment scores of textual product reviews are distributed in the datasets. The results are shown in Figure 6, Figure 7, and Figure 8. Each figure comprises of two plots: first shows the number of product ratings assigned by users and second shows the distribution of average sentiment scores of products extracted from textual product reviews in the Datafiniti Hotel reviews, Amazon Digital Music and Amazon Video Games datasets respectively. In all datasets, in product ratings plot we observe that the large number of products corresponds to rating equal to 5 that are positive rated products, then are number of products rated as neutral and least number of products are found to be rated as negative. Similarly, in the distribution of average sentiment scores in all datasets, peak is around 0.25 which indicates that there is higher probability to observe products with positive sentiments.

We know that negations play a crucial role in determining the sentiments of the reviews as negations can alter the sentiment values. So, next we proceeded to look into distribution of negations in textual product reviews in the datasets and compare with sentiments (negative, neutral or positive) classified based on product ratings given by users and product average sentiment scores from textual product reviews. The results for Datafiniti Hotel reviews, Amazon Digital Music and Amazon Video Games datasets are shown in Figure 9, Figure 10 and Figure 11 respectively. In the figures, bars in cyan color shows the number of products which are classified as negative, neutral or positive based on product ratings. Second bar (dark blue) shows the number of products which are classified as negative, neutral or positive based on average sentiment scores. Third bar (purple) which is within second bar as represents number of negations in the textual reviews retrieved from same sentiment analysis engine as average sentiment scores.

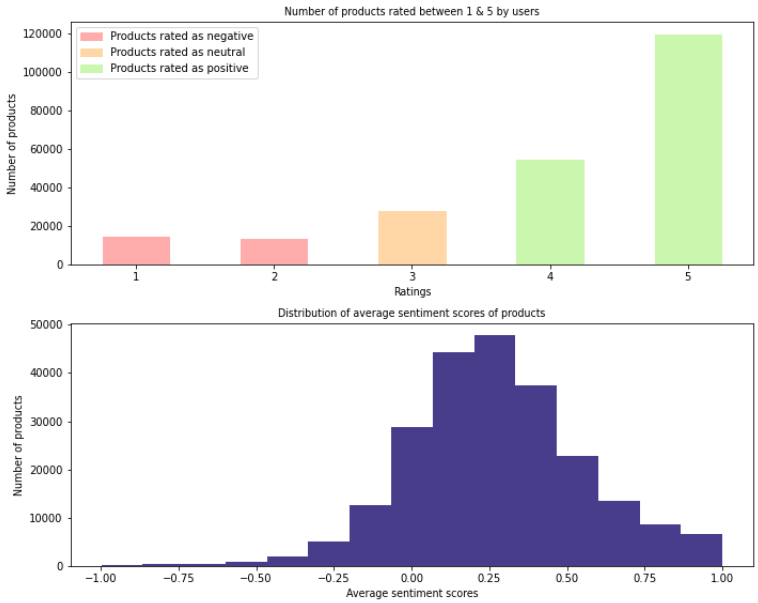

Figure 8. Distribution of product ratings and average sentiment scores for Amazon Video Games.

Firstly, we observe that number of products whose sentiment is classified based on average sentiment scores using VADER almost aligns with the number of products whose sentiment is classified based on product ratings (ground truth) indicating that VADER average sentiment scores of text/reviews can be used in downstream tasks like sentiment analysis.

Secondly, we observe that all products with sentiments (negative, neutral and positive) classified based on average sentiment scores of textual reviews indicate presence of negations. In Datafiniti Hotel Reviews, 1187 products' reviews show presence of negation out of 8056 products with positive sentiment from textual reviews, 438 products' reviews show presence of negation out of 807 products with neutral sentiment from textual reviews, and 475 products' reviews show presence of negation out of 1135 products with negative sentiment from textual reviews. Similar is the case in Amazon Digital Music and Video Games datasets. However, when we compare among reviews of products corresponding to negative, neutral and positive sentiment, we observe that percentage of negations in the product reviews present with respect to neutral and negative sentiment is higher than percentage of negations found in the product reviews classified as positive sentiment. For Datafiniti Hotel reviews dataset, percentage of negations in the product reviews classified as negative sentiment is 42% (475 out of 1135), neutral sentiment is 54% (438 out of 807), and positive sentiment is 15% (1187 out of 8056). For Amazon Digital Music dataset, percentage of negations in the product reviews classified as negative sentiment is 82% (3162 out of 3852), neutral sentiment is 99% (4149/4153), and positive sentiment is 45% (25754 out of 56700). For Amazon Video Games dataset, percentage of negations in the product reviews classified as negative sentiment is 75% (18299 out of 24260), neutral sentiment is 86% (18724 out of 21712), and positive sentiment is 46% (85611 out of 185764).

These results indicate that presence of negations is significant in reviews with negative and neutral sentiments.

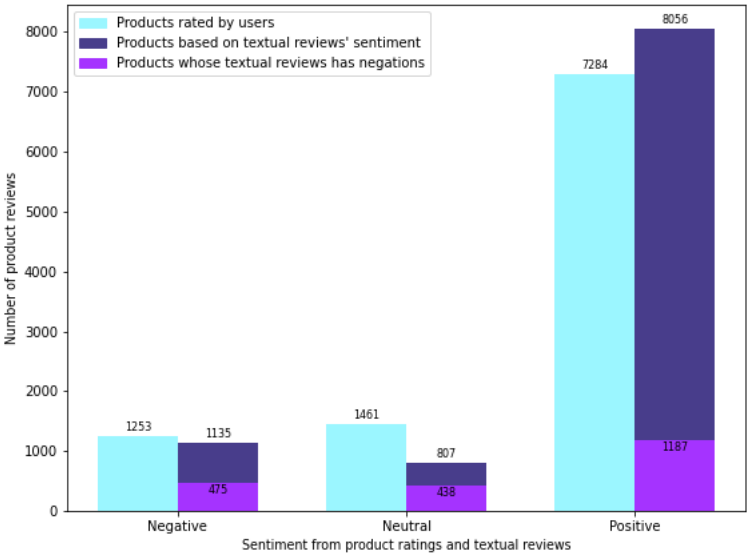

Figure 9. Shows products classified based on ratings and average sentiment scores as well as number of negations in the reviews (reviews classified as negative, positive & neutral based on average sentiment scores) for Datafiniti Hotel Reviews.

4.2. Experiment 2: Prediction of Rating Task

In this experiment, we evaluated the proposed recommendation system for rating prediction task. We evaluated the algorithm/model by using the predictors/features with different combinations. We have named the different combinations as P1, P2, P3, P4, P5, and P6. Table 3 shows all the combinations in detail. For instance, P4 is combination of ratings, embedding of the product reviews and average sentiment scores of product reviews. These all are aggregated into vector are used for training/modeling.

For each combination of predictors, whole dataset was split into 80% of whole dataset for training the model and 20% of whole dataset for testing the trained model. For evaluation, RMSE and MAE evaluation metrics were used. The results for each combination of features for Datafiniti Hotel reviews, Amazon Digital Music and Amazon Video Games datasets are showed in Figure 12, Figure 13 and Figure 14 respectively.

Firstly, we observe that RMSE values are more than MAE values for all combinations of all datasets. This is expected behaviour as mentioned in previous chapter. Also, difference between RMSE and MAE is not very large so magnitude of individual errors are not large. [62].

Comparison of combinations of different predictors: When we train the proposed model with only ratings (P1), it is considered pure model as no additional information about user or product is used. For the pure model we get large RMSE and MAE error values. For Datafiniti Hotel Reviews, Amazon Digital Music and Amazon Video Games datasets, RMSE values are 1.29, 1.14 and 1.27 respectively. MAE values for Datafiniti Hotel Reviews, Amazon Digital Music and Amazon Video Games datasets are 1.04, 0.81, and 0.91 respectively. On the other hand, when use

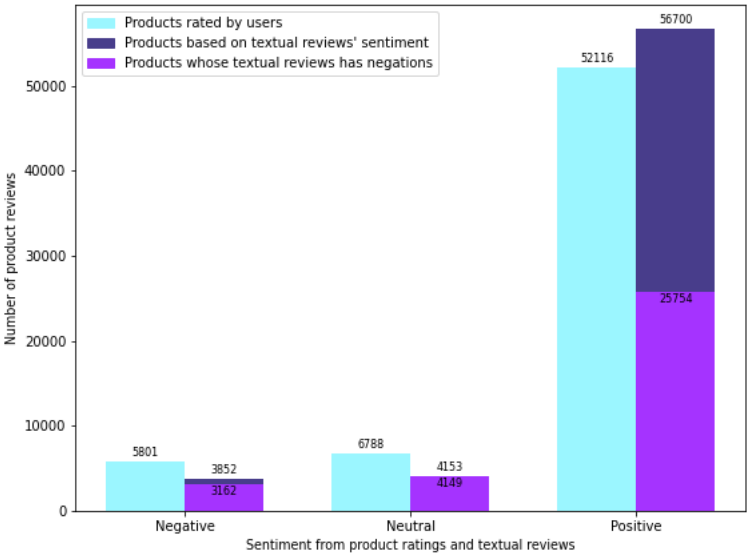

Figure 10. Shows products classified based on ratings and average sentiment scores as well as number of negations in the reviews (reviews classified as negative, positive & neutral based on average sentiment scores) for Amazon Digital Music.

extra predictors/features like contextual embedding vectors, average sentiment scores and presence of negations in textual reviews to train the recommendation model, it is called hybrid model as along with pure features additional features of products and user are used.

We observed that using only contextual embedding of the reviews as an additional feature (P2 combination), both error metrics are reduced in comparison with pure model for all datasets. For instance, in case of Amazon Video Games dataset, RMSE and MAE value of the model with P2 combination is reduced to 1.16 and 0.82 respectively. This indicates that pure model for rating prediction task improves on using contextual embedding of the product reviews. This addresses second part of research question, **RQ1**.

Second, we observed that using only average sentiment scores of the reviews as an additional feature (P3 combination), both error metrics are reduced in comparison with pure model for all datasets. For instance, in case of Amazon Video Games, RMSE and MAE value of the model with P2 combination is reduced 1.13 and 0.79 respectively. This indicates that pure model for rating prediction task improves on using average sentiment scores of the product reviews. This addresses second part of research question, **RQ2**.

Next, we observe that using contextual embedding of the reviews and presence of negations as an additional feature (P6 combination), both error metrics are reduced in comparison with pure model as well P2 combination (that uses only contextual embedding) for all datasets. For instance, in case of Amazon Video Games, using only contextual embedding RMSE and MAE value is reduced to 1.15 and 0.8 respectively. Also, these values of both error metrics are less than the RMSE and MAE value of model with P2 combination (RMSE as 1.16 and MAE as 0.82). This indicates that pure model for rating prediction task is improved by using negations in the product

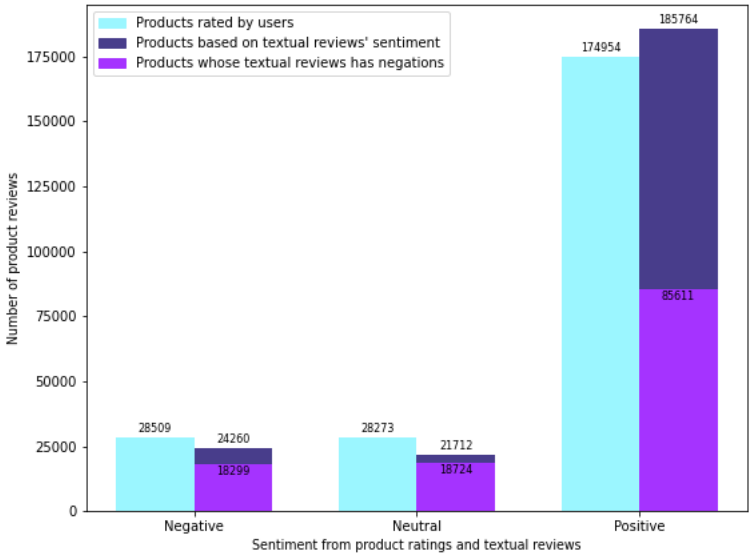

Figure 11. Shows products classified based on ratings and average sentiment scores as well as number of negations in the reviews (reviews classified as negative, positive & neutral based on average sentiment scores) for Amazon Video Games.

reviews in combination with contextual embedding vectors. This addresses second part of research question, **RQ3**.

For Datafiniti Hotel Reviews dataset, RMSE error values of the recommendation model are least in case of P4 combination (about 1.01) i.e. aggregated vector of ratings, contextual embedding of the reviews and average sentiment scores of the reviews and also, in case of P5 combination (about 1.01) i.e. aggregated vector of ratings, contextual embedding of reviews, average sentiment scores of reviews and presence of negations. Although, MAE value is least for P5 combination (about 0.79), MAE value of P4 combination (about 0.8) is 0.01 value more than P5 combination. So, we can say that both P4 and P5 combinations performs well when compared with other combinations.

For Amazon Digital Music dataset also, P4 combination i.e. aggregated vector of ratings, contextual embedding of reviews and average sentiment scores of reviews performs best as both error values (RSME as 1.01 and MAE as 0.7) are least in comparison to other predictor combinations. Also, we observe that RMSE value for P5 combination (about 1.02) is 0.01 more than P4 combination whereas MAE value for P4 is same as P5 (about 0.7).

The evaluation of Amazon Video Games dataset is similar to Datafiniti Hotel Reviews dataset. RMSE error values for P4 (about 1.09) and P5 (about 1.09) combinations are found to be least in comparison to other predictor combinations. Similarly, MAE values also of P4 (about 0.76) and P5 (about 0.8) differ by 0.01.

From all the combinations, we can conclude that P4 combination i.e. ratings, contextual embedding and average sentiment scores of the product reviews and P5 combination i.e. ratings, contextual embedding, average sentiment scores of product reviews and presence of negations in the product reviews performs the best. This addresses research question, **RQ4**.

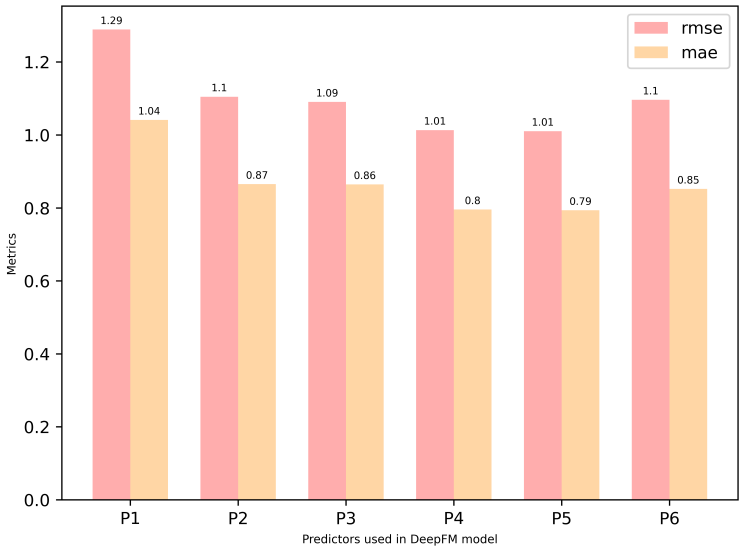

Figure 12. Rating prediction task for Datafiniti Hotel Reviews dataset.

4.3. Experiment 3: Prediction of Ranking Task

In this experiment, we evaluated the proposed recommendation model designed for ranking prediction task where predicted recommendation lists were evaluated instead of predicted ratings. We compare the predicted recommendation list with the desired recommendation list of each user which was created by gathering all items consumed by users.

For each combination of predictors, whole dataset was split into train and test set. The model was trained on training set (80% of whole dataset) with different combination of predictors to identify which combination of predictors performs best and whether using additional features/predictors such as contextual sentence embedding vectors improves the pure model (P1 combination i.e. only using ratings to train the model). Next, trained models of each combination of predictors were evaluated on testing set (20% of whole dataset) using four metrics: Precision@N, Recall@N, MAP@N, and NDCG@N. Same combination of predictors are utilised as in rating prediction task: P1, P2, P3, P4, P5, and P6. Table 2 lists different aggregated vectors considered for modeling and evaluation. The results of all metrics for each combination of features for Datafiniti Hotel reviews, Amazon Digital Music and Amazon Video Games datasets are shown in Figure 15, Figure 16 and Figure 17 respectively. Each figure consists of four plots for each metrics. We have evaluated metrics with different recommendation lists (N) as 5, 10, 15, 20 and 25.

In precision plots of all datasets, we observe that precision value for all predictor combinations decreases as number of recommendations (N) considered for computing the metrics increases. Except for Amazon Digital dataset, where precision neither decreases or increases, it is almost same.

In MAP plots of Hotel reviews dataset, we observe that MAP values for all predictor combinations slightly increases as number of recommendations (N) considered for computing the metrics increases. However, In MAP plots of Amazon Video

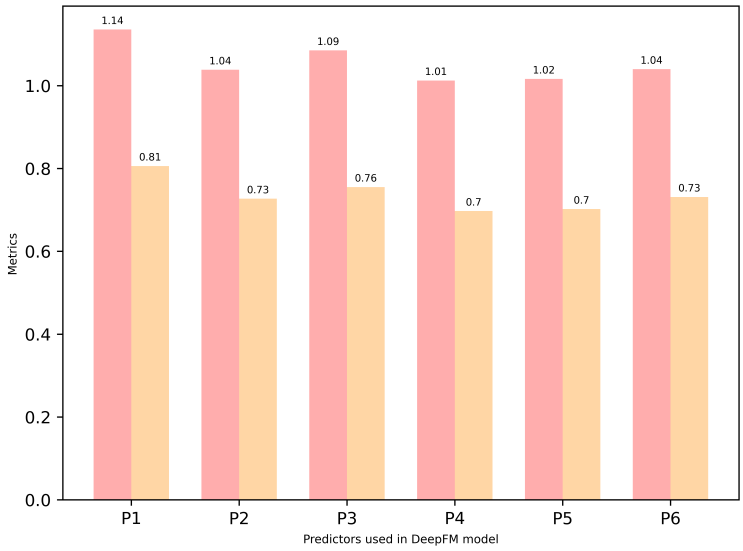

Figure 13. Rating prediction task for Amazon Digital Music dataset.

Games dataset, MAP values for all predictor combinations decreases as number of recommendations (N) considered for computing the metrics increases. Also, for Amazon Video Games dataset, MAP values for all predictor combinations decreases as number of recommendations (N) considered for computing the metrics increases.

In recall plots of all datasets, we observe that recall values for all predictor combinations increases or are almost same as number of recommendations (N) considered for computing the metrics increases. For example, for Amazon Video Games dataset, recall values for majority of combinations are almost same.

In NDCG plots of all datasets, we observe that recall values for all predictor combinations increases as number of recommendations (N) considered for computing the metrics increases. Except for Amazon video Games dataset, where NDCG decreases.

Comparison of combinations of different predictors: Firstly, we observe that values of all metrics at N, for P2 combination i.e., only contextual embedding of the product reviews are used with ratings in the model, are more than all metrics values of pure model. This is observed for all datasets. Hence, this shows that when contextual embedding of the product reviews are utilised as additional predictor the recommendation model improves. This addresses our first part of first research question **(RQ1)** that contextual embedding of the product reviews have impact on ranking type of recommendation.

Secondly, we observe that values of all metrics at N, for P3 combination i.e., only average sentiment scores of product reviews are used with ratings in the model, are more than all metrics values of the pure model. This is observed for all datasets. Hence, this shows that when average sentiment scores of product reviews are utilised as additional predictor, the recommendation model improves. This addresses our first part of second research question **(RQ2)** that average sentiment scores of textual product reviews have impact on ranking type of recommendation.

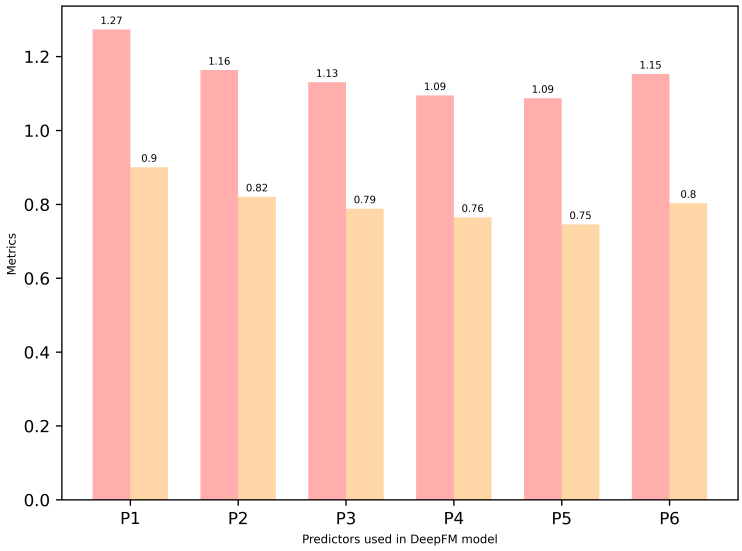

Figure 14. Rating prediction task for Amazon Video Games dataset.

Next, we observe that values of all metrics at N, for P6 combination i.e., contextual embedding of the product reviews and presence of negations in product reviews are used along with ratings in the model, are more than all metrics at N values of the pure model. This behaviour is observed for Amazon Digital Music and Datafiniti Hotel Reviews datasets whereas in case of Amazon Video Games dataset, values of all metrics@N are same for P6 combination and pure model. Also, it is observed that values of all metrics are same for P2 combination which uses contextual embedding of the product reviews and ratings in the model and P6 combination which uses contextual embedding, ratings, and presence of negations in the model. However, for Amazon Digital Music and Hotel reviews dataset, for P6 combination values of all metrics are less than the values for P2 combination. This shows that use of presence of negation feature with or without contextual embedding of the product reviews doesn't impact the recommendation. However, use of presence of negation feature along with reviews in the model (P6 combination) does improve the pure model as all metrics for P6 combination outperforms pure model. Thus, we can say that to some extent negations does have some effect on recommendation for Amazon Digital Music and Datafiniti Hotel Reviews datasets **(RQ3)**.

The best to worst precision values for all predictor combinations from precision plots for Datafiniti Hotel reviews is P5>P2>P6>P4>P3>P1, for Amazon Digital Music is P2>P6>P4>P5>P3>P1, for Amazon Video Games is P3>P4>P2>P5=P6=P1. We see that pure model (P1) has least precision values. From the sequence we can say that P2 combination i.e. ratings along with contextual embedding of the product reviews (as additional information) results in better precision value for all datasets. As it is highest for Digital Music dataset and second and third highest for Hotel Reviews and Video Games datasets respectively.

The best to worst recall values for all predictor combinations sequence for Datafiniti Hotel reviews is P5>P2>P6>P4>P3>P1, for Amazon Digital Music is P2>P6>P4>P3>P5>P1, for Amazon Video Games is P3>P2>P4>P5= P6=P1. From

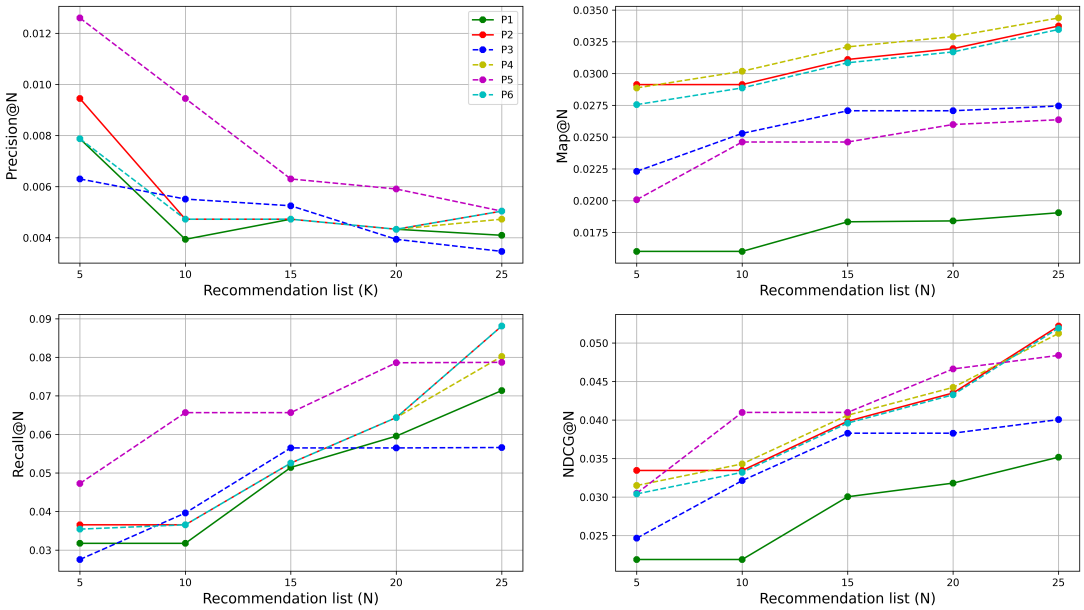

Figure 15. Result of ranking prediction task for Datafiniti Hotel Reviews dataset.

the sequences, it can be said that P2 combination i.e. contextual embedding of the product reviews along with ratings used in model yields better recall values.

The best to worst MAP values for all predictor combinations sequence for Datafiniti Hotel reviews is P4>P2>P6>P3>P5>P1, for Amazon Digital Music is P2>P6>P4>P3>P5>P1 and for Amazon Video Games is P3>P4>P2>P5=P6=P1. Overall, from the sequences of all datasets, it can be said that P4 combination (embedding vectors and average sentiment scores) and P2 (embedding vectors) yields better MAP values.

The best to worst NDCG values for all predictor combinations sequence for Datafiniti Hotel reviews is P5>P4>P2>P6>P5>P1, for Amazon Digital Music is P2>P6>P4>P5>P3>P1, for Amazon Video Games is P3>P4>P2>P5=P6=P1. Overall, from the sequences of all datasets, it can be said that P4 combination (embedding vectors and sentiment scores) and P2 (embedding vectors) yields better NDCG values.

From above observations, firstly, we can conclude that hybrid recommendation model using features such as contextual embedding, sentiment scores and presence of negations in the product reviews performs better than pure model (where only user preference is considered from ratings). Second that from all predictor combinations, P2 (embedding vectors along with ratings) and P4 (contextual embedding vectors and average sentiment scores along with ratings) combinations outperformed other combinations. This addresses the fourth research question of the thesis **(RQ4)**.

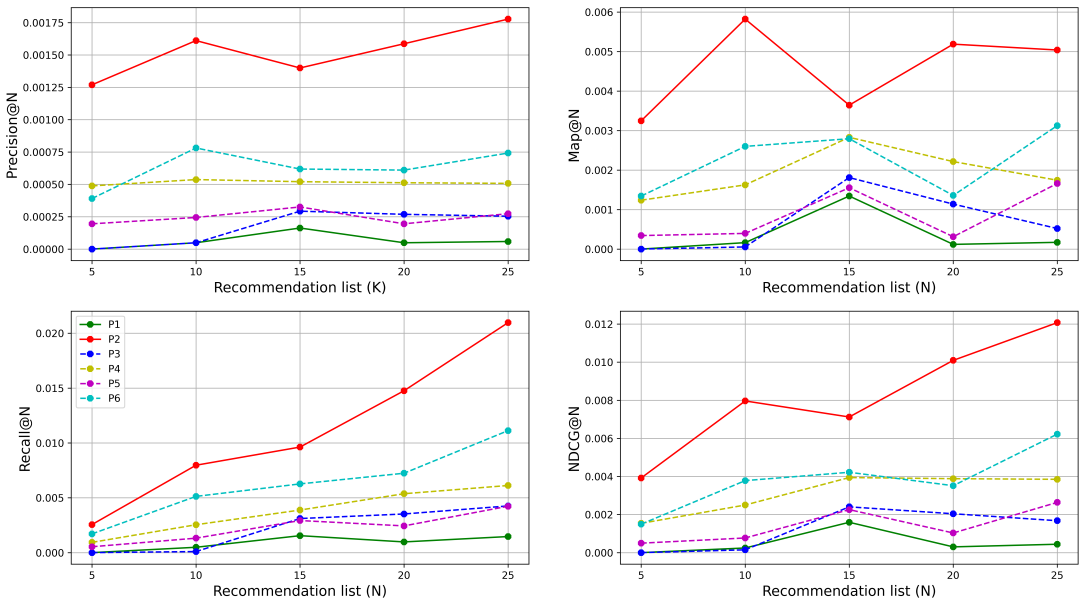

Figure 16. Result of ranking prediction task for Amazon Digital Music dataset.

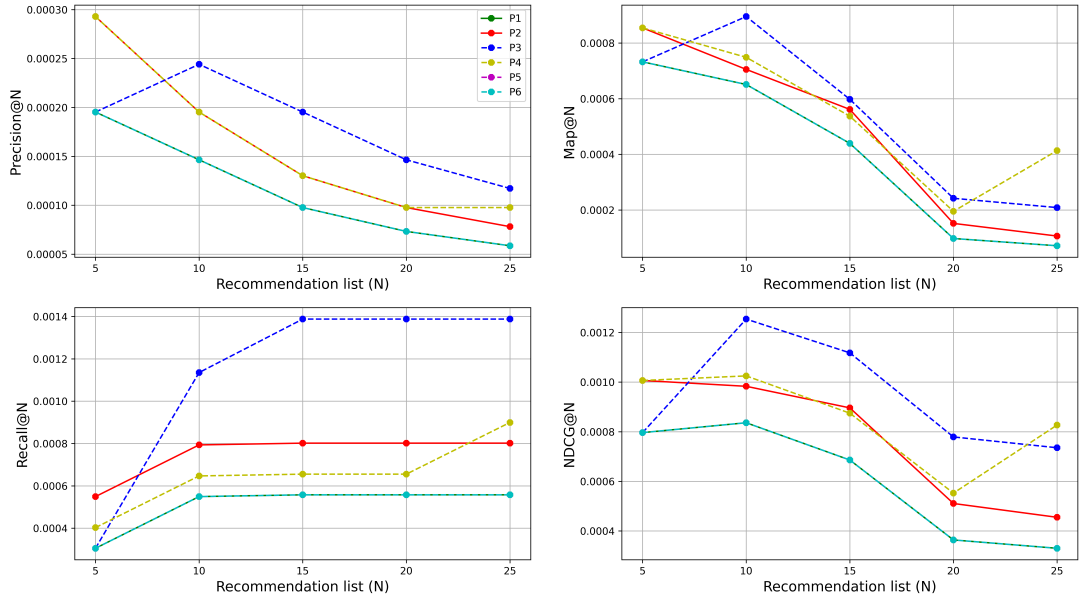

Figure 17. Result of ranking prediction task for Amazon Video Games dataset.

5. SUMMARY

In this master's thesis, we explored different approaches in Natural Language Processing from state-of-the-art Deep Learning models like DeepFM and BERT to Rule-Based approaches like VADER. In the thesis, we proposed a hybrid recommendation model which uses additional information about users and products from textual product reviews such as their contextual sentence embedding, sentiment scores and label information of whether negations are present in the reviews or not. Chapter 3 described the overall method utilised to prepare/extract the features for modeling. Chapter 4 provided the results and analysis of the three experiments: Exploratory Data Analysis, Prediction of Rating Task and Prediction of Ranking Task. These experiments' analysis helped in addressing the four research questions described in chapter 1 (Introduction). From our results and analysis of the experiment 1, we conclude following three points:

1. In the all the datasets considered, large number of products based on ratings assigned by user corresponded to positive rated products as well as distribution of sentiment scores from VADER of product's reviews showed that majority of samples in datasets are observed as positive sentiment.

2. In all datasets, we observed that number of products whose sentiment was classified based on average sentiment scores using VADER almost aligned with the number of products whose sentiment was classified based on product ratings given by users indicating that VADER average sentiment scores of text can be used in sentiment analysis of different domains.

3. In all datasets, we observed that percentage of negations in product reviews classified as neutral and negative sentiment was higher than percentage of negations found in product reviews classified as positive sentiment.

From our results and analysis of the experiment 2 i.e. utilising proposed model which predicts desired ratings of the products, we conclude following points:

1. For all datasets, we observed that when contextual sentence embedding of product reviews from BERT was utilised as an additional predictor to the proposed recommendation model (DeepFM), predictions of desired ratings improved as compared to the recommendation model which utilised only ratings, indicating that embedding vectors of product reviews does effect recommendation based on rank scores positively. **RQ1**

2. For all datasets, we found that when average sentiment scores of product reviews was utilised as an additional predictor to the proposed model, predictions of desired ratings improved as compared to the recommendation model which utilised only ratings, indicating that sentiments from product reviews does effect recommendation based on rank scores positively. **RQ2**

3. For all datasets, we observed that use of presence/absence of negation feature with contextual embedding of the product reviews in combination with ratings in the proposed model improved the prediction of desired ratings as compared to

the recommendation model which utilised contextual embedding and ratings, indicating that linguistic cue like presence/absence of negations have some impact on predicting desired ratings. **RQ3**

4. For all datasets, we found that out of all predictor combinations, two combinations: first, contextual embedding of product reviews along with ratings; second, contextual embedding of the product reviews and average sentiment scores and presence/absence of negation in the product reviews along with ratings combinations outperformed other combinations proving to be the best combination to enhance predicting desired ratings. **RQ4**

From our results and analysis of the experiment 3, we conclude following points:

1. For all datasets, we observed that when contextual sentence embedding of product reviews from BERT was utilised as an additional predictor to the proposed recommendation model (DeepFM), recommendations based on prediction of rank scores improved with respect to the recommendation model which utilised only ratings, indicating that embedding vectors of product reviews does effect recommendation based on rank scores positively. **RQ1**

2. For all datasets, we found that when average sentiment scores of product reviews was utilised as an additional predictor to the proposed model, recommendations based on prediction of rank scores improved with respect to the recommendation model which utilised only ratings, indicating that sentiments from product reviews does effect recommendation based on rank scores positively. **RQ2**

3. For Amazon Video Games dataset, we observed that use of presence/absence of negation feature with or without contextual embedding of the product reviews in combination with ratings in the proposed model doesn't improve prediction of rank scores but perform equally. However, for Amazon Digital Music and Datafiniti Hotel Reviews datasets, use of presence/absence of negation feature with contextual embedding vectors in combination with ratings in the proposed model does slightly improve prediction of rank scores with respect to the recommendation model which utilised contextual embedding vectors and ratings, indicating that linguistic cue like presence/absence of negations to some extent impact prediction of rank scores. **RQ3**

4. For all datasets, we found that out of all predictor combinations, two combinations: first, contextual embedding of product reviews along with ratings; second, contextual embedding of product reviews and average sentiment scores along with ratings combinations outperformed other combinations proving to be the best combination to enhance recommendations based on rank scores. **RQ4**

In conclusion, it was observed that using either contextual embedding vectors or average sentiment scores as an extra feature to the proposed recommendation DeepFM model improved prediction of desired ratings and recommendations based on rank scores with respect to the recommendation model which utilised only ratings. However, using both contextual embedding vectors and average sentiment scores together further enhances the performance of both prediction rating and

ranking prediction tasks. Also, using presence/absence of negation in the product reviews as a feature along with contextual embedding vectors and ratings to the recommendation model improved prediction of desired ratings with respect to the recommendation model which utilised contextual embedding vectors and ratings. Whereas using presence/absence of negation in the product reviews as a feature along with contextual embedding vectors and ratings in the model to some extent improved the recommendation based on prediction of rank scores with respect to the recommendation model which utilised contextual embedding and ratings. Hence, we infer that proposed hybrid recommendation system which makes use of multiple features derived from textual product reviews yields better predicted ratings and recommendations based on predicted scores.

The work presented here can be expanded by enhancing the proposed recommendation system by including user's social network to find users with similar interests. Also, the proposed method can be compared and evaluated with other state-of-the-art recommendation models and data from other domains to further generalise the use of the proposed model.

6. REFERENCES

[1] Wikipedia, Recommender system. https://en.wikipedia.org/wiki/Recommender_system.

[2] Hadash G., Shalom O.S. & Osadchy R. (2018) Rank and rate. In: Proceedings of the 12th ACM Conference on Recommender Systems, ACM. URL: https://doi.org/10.1145%2F3240323.3240406.

[3] Tugend A. (2010) Too many choices: A problem that can paralyze. The New York Times 26.

[4] Gupta J. & Gadge J. (2015) Performance analysis of recommendation system based on collaborative filtering and demographics. In: 2015 International Conference on Communication, Information Computing Technology (ICCICT), pp. 1–6.

[5] Adomavicius G. & Tuzhilin A. (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering 17, pp. 734–749.

[6] Karthik R. & Ganapathy S. (2021) A fuzzy recommendation system for predicting the customers interests using sentiment analysis and ontology in e-commerce. Applied Soft Computing 108, p. 107396.

[7] Dang C.N., Moreno-García M.N. & Prieta F.D.l. (2021) An approach to integrating sentiment analysis into recommender systems. Sensors 21, p. 5666. URL: http://dx.doi.org/10.3390/s21165666.

[8] Wikipedia, Sentence embedding. https://en.wikipedia.org/wiki/Sentence_embedding.

[9] Hutto C. & Gilbert E. (2014) Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: Proceedings of the international AAAI conference on web and social media, vol. 8, vol. 8, pp. 216–225.

[10] Devlin J., Chang M.W., Lee K. & Toutanova K. (2018), Bert: Pre-training of deep bidirectional transformers for language understanding. URL: https://arxiv.org/abs/1810.04805.

[11] Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M. & Duchesnay E. (2011) Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, pp. 2825–2830.

[12] Guo H., Tang R., Ye Y., Li Z. & He X. (2017), Deepfm: A factorization-machine based neural network for ctr prediction. URL: https://arxiv.org/abs/1703.04247.

[13] Pazzani M.J. & Billsus D. (2007) Content-Based Recommendation Systems, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 325–341. URL: https://doi.org/10.1007/978-3-540-72079-9_10.

[14] Pan Y., Huo Y., Tang J., Zeng Y. & Chen B. (2021) Exploiting relational tag expansion for dynamic user profile in a tag-aware ranking recommender system. Information Sciences 545, pp. 448–464.

[15] Hwang T.G., Park C.S., Hong J.H. & Kim S. (2016) An algorithm for movie classification and recommendation using genre correlation. Multimedia Tools and Applications 75.

[16] Ostuni V.C., Di Noia T., Mirizzi R. & Di Sciascio E. (2014) A linked data recommender system using a neighborhood-based graph kernel. In: M. Hepp & Y. Hoffner (eds.) E-Commerce and Web Technologies, Springer International Publishing, Cham, pp. 89–100.

[17] Ricci F., Rokach L. & Shapira B. (2015) Chapter 1 recommender systems : Introduction and challenges.

[18] Aditya P., Budi I. & Munajat Q. (2016) A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for e-commerce in indonesia: A case study pt x. pp. 303–308.

[19] Herlocker J.L., Konstan J.A., Terveen L.G. & Riedl J.T. (2004) Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22, p. 5–53. URL: https://doi.org/10.1145/963770.963772.

[20] Chen R., Hua Q., Chang Y.S., Wang B., Zhang L. & Kong X. (2018) A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. IEEE Access 6, pp. 64301–64320.

[21] Jalili M., Ahmadian S., Izadi M., Moradi P. & Salehi M. (2018) Evaluating collaborative filtering recommender algorithms: A survey. IEEE Access 6, pp. 74003–74024.

[22] Aramanda A., Md Abdul S. & Vedala R. (2021) A comparison analysis of collaborative filtering techniques for recommender systems in lecture notes in electrical engineering 698. Springer: Singapore, pp. 87–95. URL: https://link.springer.com/content/pdf/10.1007/978-981-15-7961-5.pdf.

[23] Gong S., Ye H. & Tan H. (2009) Combining memory-based and model-based collaborative filtering in recommender system. 2009 Pacific-Asia Conference on Circuits, Communications and Systems , pp. 690–693.

[24] Burke R. (2002) Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12. URL: https://doi.org/10.1023/A:1021240730564.

[25] Lalita S. & Anju G. (2013) A survey of recommendation system: Research challenges. International Journal of Engineering Trends and Technology (IJETT) published by seventh sense research group. Volume-4 Issue-5. URL: http://ijettjournal.org/archive/ijett-v4i5p132.

[26] Khusro S., Ali Z. & Ullah I. (2016) Recommender systems: Issues, challenges, and research opportunities. In: K.J. Kim & N. Joukov (eds.) Information Science and Applications (ICISA) 2016, Springer Singapore, Singapore, pp. 1179–1189.

[27] Wankhade M., Rao A.C.S. & Kulkarni C. (2022) A survey on sentiment analysis methods, applications, and challenges. Artificial Intelligence Review , pp. 1–50.

[28] Miller G.A. (1995) Wordnet: a lexical database for english. Communications of the ACM 38, pp. 39–41.

[29] Wei J., He J., Chen K., Zhou Y. & Tang Z. (2017) Collaborative filtering and deep learning based recommendation system for cold start items. Expert Systems with Applications 69, pp. 29–39.

[30] Herce-Zelaya J., Porcel C., Bernabé-Moreno J., Tejeda-Lorente A. & Herrera-Viedma E. (2020) New technique to alleviate the cold start problem in recommender systems using information from social media and random decision forests. Information Sciences 536, pp. 156–170.

[31] Bonta V. & Janardhan N.K.N. (2019) A comprehensive study on lexicon based approaches for sentiment analysis. Asian Journal of Computer Science and Technology 8, pp. 1–6.

[32] Baccianella S., Esuli A. & Sebastiani F. (2010) SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), European Language Resources Association (ELRA), Valletta, Malta. URL: http://www.lrec-conf.org/proceedings/lrec2010/pdf/769_Paper.pdf.

[33] Pennebaker J.W., Francis M.E. & Booth R.J. (2001) Linguistic inquiry and word count: Liwc 2001. Mahway: Lawrence Erlbaum Associates 71, p. 2001.

[34] Stone P.J., Dunphy D.C. & Smith M.S. (1966) The general inquirer: A computer approach to content analysis. .

[35] Elbagir S. & Yang J. (2019) Twitter sentiment analysis using natural language toolkit and vader sentiment. In: Proceedings of the international multiconference of engineers and computer scientists, vol. 122, vol. 122, p. 16.

[36] Pano T. & Kashef R. (2020) A complete vader-based sentiment analysis of bitcoin (btc) tweets during the era of covid-19. Big Data and Cognitive Computing 4. URL: https://www.mdpi.com/2504-2289/4/4/33.

[37] Mustaqim T., Umam K. & Muslim M. (2020) Twitter text mining for sentiment analysis on government's response to forest fires with vader lexicon polarity detection and k-nearest neighbor algorithm. In: Journal of Physics: Conference Series, vol. 1567, IOP Publishing, vol. 1567, p. 032024.

[38] Borg A. & Boldt M. (2020) Using vader sentiment and svm for predicting customer response sentiment. Expert Systems with Applications 162, p. 113746. URL: https://www.sciencedirect.com/science/article/pii/S0957417420305704.

[39] Feldman R. & Dagan I. (1995) Knowledge discovery in textual databases (kdt). KDD'95, AAAI Press, p. 112–117.

[40] Cloud I.E. (2020), Text mining. https://www.ibm.com/cloud/learn/text-mining.

[41] Jia L., Yu C. & Meng W. (2009) The effect of negation on sentiment analysis and retrieval effectiveness. In: Proceedings of the 18th ACM conference on Information and knowledge management, pp. 1827–1830.

[42] Councill I., McDonald R. & Velikovich L. (2010) What's great and what's not: learning to classify the scope of negation for improved sentiment analysis .

[43] Reitan J., Faret J., Gambäck B. & Bungum L. (2015) Negation scope detection for twitter sentiment analysis. In: Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 99–108.

[44] Chapman W., Bridewell W., Hanbury P., Cooper G. & Buchanan B. (2001) A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics 34, pp. 301–310.

[45] Mehrabi S., Krishnan A., Sohn S., Roch A.M., Schmidt H., Kesterson J., Beesley C., Dexter P., Schmidt C.M., Liu H. et al. (2015) Deepen: A negation detection system for clinical text incorporating dependency relation into negex. Journal of biomedical informatics 54, pp. 213–219.

[46] Sohn S., Wu S. & Chute C.G. (2012) Dependency parser-based negation detection in clinical narratives. AMIA Summits on Translational Science Proceedings 2012, p. 1.

[47] Khandelwal A. & Sawant S. (2019), Negbert: A transfer learning approach for negation detection and scope resolution. URL: https://arxiv.org/abs/1911.04211.

[48] World D., Hotel Reviews dataset from Datafiniti's Business dataset. https://data.world/datafiniti/hotel-reviews.

[49] McAuley J., Amazon Product data. http://jmcauley.ucsd.edu/data/amazon/index_2014.html/.

[50] Wolf T., Debut L., Sanh V., Chaumond J., Delangue C., Moi A., Cistac P., Rault T., Louf R., Funtowicz M., Davison J., Shleifer S., von Platen P., Ma C., Jernite Y., Plu J., Xu C., Le Scao T., Gugger S., Drame M., Lhoest Q. & Rush A. (2020) Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, pp. 38–45. URL: https://aclanthology.org/2020.emnlp-demos.6.

[51] Glen S., Explained variance / variation from statisticshowto.com: Elementary statistics for the rest of us! URL: https://www.statisticshowto.com/explained-variance-variation/.

[52] Rosenthal J.A. (2011), Statistics and data interpretation for social work. URL: https://eric.ed.gov/?id=ED578967.

[53] Guo G., Zhang J., Sun Z. & Yorke-Smith N. (2015) Librec: A java library for recommender systems. In: A.I. Cristea, J. Masthoff, A. Said & N. Tintarev (eds.) UMAP Workshops, CEUR Workshop Proceedings, vol. 1388, CEUR-WS.org, CEUR Workshop Proceedings, vol. 1388. URL: http://dblp.uni-trier.de/db/conf/um/umap2015w.html#GuoZSY15.

[54] Education I.C. (2020), Natural language processing (nlp). URL: https://www.ibm.com/cloud/learn/natural-language-processing.

[55] Hurwitz J. & Kirsch D. (2018) Machine Learning for dummies, John Wiley & Sons, Inc. pp. 13–18. URL: https://www.ibm.com/downloads/cas/GB8ZMQZ3.

[56] Team Q.E. (2020), Rule-based vs machine learning approach. URL: https://qualitastech.com/blog/image-processing/differences-between-machine-learning-and-rule-based-systems/#rulebasedapproach.

[57] Dorash M. (2017), Machine learning vs. rule based systems in nlp. URL: https://medium.com/friendly-data/machine-learning-vs-rule-based-systems-in-nlp-5476de53c3b8.

[58] Sun B. (2019), Deepfm - combining fm and neural nets. https://bangdasun.github.io/2019/05/27/49-deepfm-combine-fm-and-neural-nets/. [Paper reading] DeepFM: A Factorization - Machine based Neural Network for CTR Prediction.

[59] Chaoran (2019), Implement deepfm model in keras. https://6chaoran.wordpress.com/2019/01/03/implement-deepfm-model-in-keras/.

[60] Alammar J. (2021), The illustrated bert, elmo, and co. (how nlp cracked transfer learning). URL: http://jalammar.github.io/illustrated-bert/.

[61] Swarnkar N. (2020), Vader sentiment analysis: A complete guide, algo trading and more. https://blog.quantinsti.com/vader-sentiment/.

[62] JJ (2016), Mae and rmse — which metric is better? https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d.

[63] Wang B. (2021), Ranking evaluation metrics for recommender systems. https://towardsdatascience.com/ranking-evaluation-metrics-for-recommender-systems-263d0a66ef54.

[64] Wikipedia, Information retrieval. https://en.wikipedia.org/w/index.php?title=Information_retrieval&oldid=793358396#Average_precision.

7. APPENDICES

Appendix 1

The works of thesis project is included in following url:
https://github.com/VandanaYadav24/MasterThesis_repo