Recipe Rating Prediction

Ketan Malempati
Master's in Artificial Intelligence
San Jose State University
San Jose, USA
ketan.malemapti@sjsu.edu

Pruthvi Raj Kotigari
Master's in Artificial Intelligence
San Jose State University
San Jose, USA
pruthviraj.kotigari@sjsu.edu

Rahul Ragava Peela
Master's in Artificial Intelligence
San Jose State University
San Jose, USA
rahulraghava.peela@sjsu.edu

Dev Savsani
Master's in Artificial Intelligence
San Jose State University
San Jose, USA
dev.savsani@sjsu.edu

***Abstract*—This project aims to build a food recommendation system that predicts ratings based on user interactions with recipes and recipe details. The dataset used in this project is sourced from Kaggle and contains a large number of user reviews and ratings for different food items. Various techniques and models, including GloVe, TF-IDF, BM25, Neural Networks, and Faiss, are being explored to calculate similarities between different food items and generate predictions for new food items. The objective is to build a robust recommendation system that can help users discover new food items they are likely to enjoy and provide valuable insights into user preferences and behavior.**

***Index Terms*—Glove, TF-IDF, BM25, Faiss, Neural Networks, Recommender System, Spacy, Sentiment analysis.**

I. Introduction

The abundance of food choices available today can make it difficult for consumers to decide what to eat. To address this problem, food recommendation systems have been developed to help users discover new food items that match their preferences. A food recommendation system predicts a user's preference for a food item based on their past interactions with similar items.

This project focuses on building a food recommendation system that predicts ratings based on user interactions with food items. The dataset used in this project contains user reviews and ratings for different food items, recipe steps, and descriptions and dates of interaction. Various models and techniques are being explored, including GloVe, TF-IDF, BM25, Neural Networks, and Faiss, to calculate similarities between different food items and generate predictions for new food items.

The goal of this project is to build a robust recommendation system and to explore different methodologies that can predict recipes to users effectively and by using relevant amounts of resources (CPU process time, RAM, Memory).help users discover new food items they are likely to enjoy, while also providing valuable insights into user preferences and behaviour. Such a system can benefit food companies and retailers by better understanding their customers' preferences and providing personalized recommendations. Overall, this project has the potential to enhance the user experience in the food industry and provide more satisfying and personalized recommendations for users.

II. System Design and Implementation details

A. Algorithm selection

Since the dataset is having several textual features, detailed descriptions, and ratings, We would like to use Natural Language Processing to tackle this problem. So we are using text similarity as a base and building, evaluating different algorithms in terms of effectiveness and resources used.

B. Semantic analysis Glove

The process of extracting meaning from text is known as semantic analysis. It enables computers to comprehend and interpret sentences, paragraphs, or entire documents by analyzing the grammatical structure and identifying relationships between individual words in a given context.

The GloVe is a technique for producing vector representations of words using unsupervised learning. Training is done using corpus-based global word-word co-occurrence statistics, and the resulting representations show off some of the word vector space's fascinating linear substructures. The Glove has a lot of features but for this use case, we are going to use the nearest neighbour methodology of the glove where the Euclidean distance (or cosine similarity) between two-word vectors is an efficient method for determining the linguistic or semantic similarity of the words. This metric's nearest neighbours can sometimes reveal rare but relevant words that are outside the average human's vocabulary. Since recipes contain steps and descriptions in data containing rare and unique words, Glove is a suitable method to compare two recipes.

C. TF-IDF

Term Frequency – Inverse Document Frequency (TF-IDF) is a popular statistical technique utilized in natural language processing and information retrieval to assess a

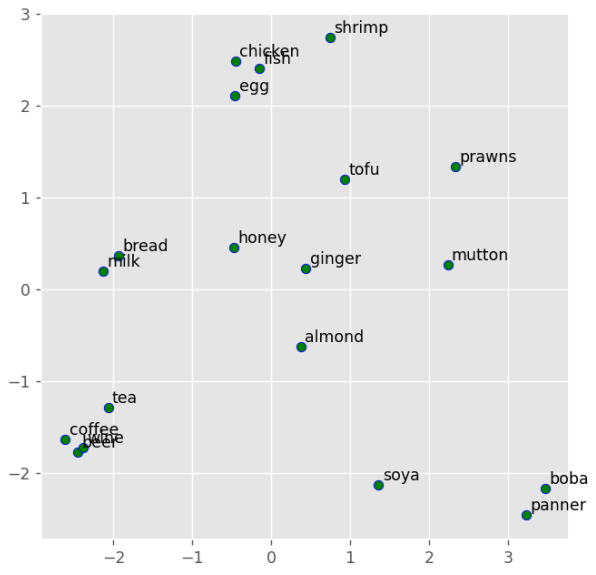

Fig. 1. A scatter plot of different ingredients of recipes using vectors from glove

$$PR(x) = \frac{\sum_{i \epsilon RI} (GloveSimilarity(i,x) * rating(i))}{\sum_{i \epsilon RI} GloveSimilarity(i,x)}$$

Fig. 2. Formula to calculate ratings for a recipe using glove similarity

PR - Predict Probability for a user
x - recipe id
RI - All recipes interacted by the user
GloveSimilarity - Method to find similarity between two recipe ids
rating - rating of a recipe id by the user

term's significance in a document compared to a group of documents, known as a corpus. The technique employs a text vectorization process to transform words in a text document into numerical values that denote their importance. [2]. TF-IDF is faster than Glove but it can not calculate semantic similarity.

$$w_{i,j} = tf_{i,j} \times log(\frac{N}{df_i})$$

Fig. 3. Formula to calculate ratings for a recipe using TF-IDF similarity

tf = number of occurrences of i in j
df = number of documents containing i
N = total number of document

For this algorithm, we first start by creating a dictionary of users and all the recipes they have rated. So, we have a list of the columns of each user. Then performing pre-processing like removing stopwords, and punctuation's. Next, when given a user id and recipe id we calculate the similarities of all the columns of the user's data with the recipe data and the similarities of each column. In the end, we take the average of all the columns and find the weighted average of the rating this would be the predicted rating.

D. BM25

BM25 is a simple Python package and can be used to index the data, tweets in our case, based on the search query. It works on the concept of TF-IDF. [3]. Same as TF-IDF it also can not calculate semantic similarity but is faster than both TF-IDF and Glove.

$$\sum_{i}^{n} IDF(q_i) \frac{f(q_i, D) * (k1 + 1)}{f(q_i, D) + k1 * (1 - b + b * \frac{fieldLen}{avgFieldLen})}$$

Fig. 4. Formula to calculate ratings for a recipe using TF-IDF

E. Faiss

Facebook AI similarity search (FAISS) is an algorithm developed by Facebook for fast and efficient similarity search, especially for large data.it is built with nearest-neighbor search implementations for billion-scale data sets that are some 8.5x faster than the previously reported state-of-the-art, along with the fastest k-selection algorithm on the GPU.

FAISS is a high-performance similarity search algorithm developed by Facebook, which is specifically designed for conducting efficient nearest-neighbor search operations on large datasets. Its advanced architecture allows it to outperform previous state-of-the-art algorithms by a factor of 8.5x. Additionally, FAISS boasts the fastest k-selection algorithm for GPUs, making it an ideal tool for conducting similarity search operations on billion-scale datasets.

The FAISS algorithm adopts a two-pronged approach for similarity search. Firstly, it employs Euclidean distance to identify the list of database objects that are closest to a given query vector. Secondly, it utilizes dot product to identify the list of database objects that exhibit the highest similarity with the query vector. Faiss is optimized to ensure efficient memory usage and speedy operations. Faiss further distinguishes itself by offering a state-of-the-art GPU implementation for the most pertinent indexing methods, thereby ensuring superior performance and scalability.

In Faiss, indexing methods are denoted using a string representation. The string contains several elements that specify the pre-processing steps to be applied to vectors, the partitioning mechanism for the database, and the encoding component for vectors, which involves the use of a product quantizer (PQ) to generate 20-byte codes. This allows Faiss to achieve an efficient memory usage of below 30 GB of RAM, inclusive

of overheads. Once the index type has been selected, Faiss can process up to 1 billion vectors, which are then added to the index. The index can either be stored on disk or used immediately, with searches and additions/removals to the index being performed concurrently.

For the food recommendation system we've used a food review dataset containing 1132198 reviews. In this case, we've used the sentence transformer model 'paraphrase-distilroberta-base-v1' to convert text to embeddings of dimension of 768. We've used a product quantizer, which provides a large number of reproduction values without increasing the processing cost. Each sub-vector is quantized with its own quantizer, yielding the tuple The sub-quantizers have 256 reproduction values, to fit in one byte.

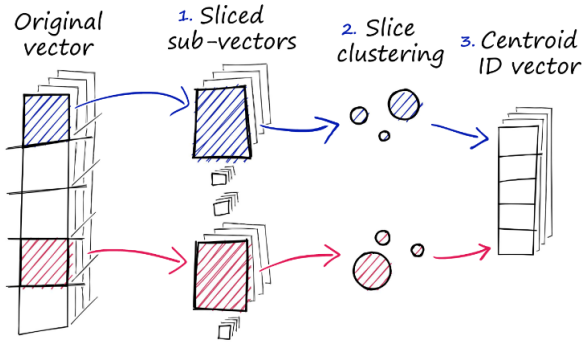

Fig. 5. Three steps of product quantization

As part of our efforts to improve our indexing methods, we have explored the use of vector compression techniques in Faiss, specifically through the use of Product Quantization (PQ). PQ involves a three-step process for approximating similarity operations on compressed vectors in large datasets. Firstly, the original vector is split into multiple subvectors. Secondly, we perform a clustering operation on each set of subvectors, thereby creating several centroid's for each sub-vector set. Finally, we replace each sub-vector in our vector of sub-vectors with the ID of its nearest set-specific centroid. This approach allows us to effectively compress our vectors while still achieving the desired levels of similarity search accuracy.

We have a dataset of embeddings. we've used a special data structure in Datasets called a FAISS index. The basic idea behind FAISS is to create a special data structure called an index that allowed us to find which embeddings are similar to an input embedding. We have performed queries on this index by doing a nearest neighbor lookup with the food Dataset to find similar vectors and corresponding rating for user recommendation.

F. Neural Networks

Neural networks are a type of machine learning model inspired by the structure and function of the human brain. They consist of layers of interconnected "neurons" that process input data and produce output predictions or classifications.

The input data is passed through the network, where it is transformed and manipulated by the neurons in each layer. The neurons are connected by weights, which determine the strength and direction of the signal between them. During training, the weights are adjusted to optimize the network's ability to make accurate predictions or classifications.

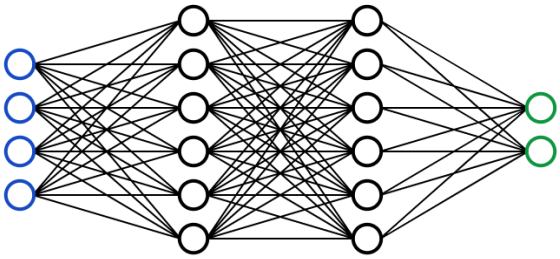

Fig. 6. General Neural Network

A TfidfVectorizer object is created with a maximum of 1000 features. This object is used to convert the description column of the "FoodRecipes" Data Frame into a matrix of TF-IDF features. This method fits the vectorizer on the documents and creates the term-document matrix, which represents the TF-IDF features for each document. LSA is used to create embeddings.

Latent Semantic Analysis (LSA) is a technique for generating a low-dimensional representation of high-dimensional data, typically used for natural language processing tasks such as document classification, topic modelling, and information retrieval.

LSA is based on the idea that words that appear in similar contexts are likely to have similar meanings. The method starts by creating a term-document matrix, which represents the frequency of each word in each document. This matrix is then transformed using singular value decomposition (SVD), a linear algebra technique that can factorize a matrix into a set of orthogonal basis vectors and their corresponding singular values.

A TruncatedSVD instance is created with n_components set to 100. This is the desired number of components for the LSA model.

We performed LSA on the TF-IDF features of the description column of the "FoodRecipes" DataFrame to generate a lower-dimensional representation of each document, which can be used for various machine learning tasks such as clustering, classification, or recommendation.

Next, create embedding for user_id, and recipe_id using the embedding of TensorFlow. We create these embeddings as the neural network must consider the user_id and Recipe_id as separate categorical entities rather than being just a number as the ratings correspond to each user's particular recipe.

Similarly create one more embedding merging the user_id, recipe_id, and the description embedding. Now, use these

embedding layers and send them to the neural network to train with ”relu” activation, and ”adam” optimizer.
Doing the train-test split with a ratio of 4:1 and fitting the neural network on the train data. Predict the rating for the user and recipes accordingly. Test on the test set and calculate the RMSE.

G. Technologies and Tools used

HPC - For faster computations.
Numba - To run code on GPU for faster processing.
Multi-Processing - To run code on multiple cores of CPU.
Python - To use several modules that are mentioned below.
Jupyter Notebook - To compile and run Python code and to display visualizations and results of respective code.
seaborn, matplotlib - To generate visualization and for EDA.
NumPy - To calculate cosine similarity (dot products) and for array calculations.
Spacy, Glove - Used for word embeddings using pre-trained models.
Pandas - Data frame handling, visualizations, pre-processing.
transformer, Tensorflow, Keras - Model training and to use pre-trained and fine-tuned models.

H. System design/architecture/data flow

We use the following data flow for evaluating TF-IDF, BM25, Spacy and Glove. After doing EDA on the data set it was observed that the data is not balanced and there are a lot of ratings with a rating of 5. Since we are using word embeddings and cosine similarities, it is not a good idea to do under-sampling, so sentiment analysis is used on reviews and the high ratings with negative ratings, and low ratings with positive reviews are filtered out from the data.
The cleaned data is grouped on the user id and new features are created with user ratings, descriptions, steps and tags.
For each item and user pair present in the test dataset, we are calculating similarity measures for each feature as shown in the following Figure 7. These ratings and similarity values are then given to the formula mentioned in Figure 2 to predict ratings for a user-item pair. This step is done for all user-item pairs. Since the number of calculations to be done are humongous we are running the above steps of Python code on a GPU using CUDA on multiple threads.

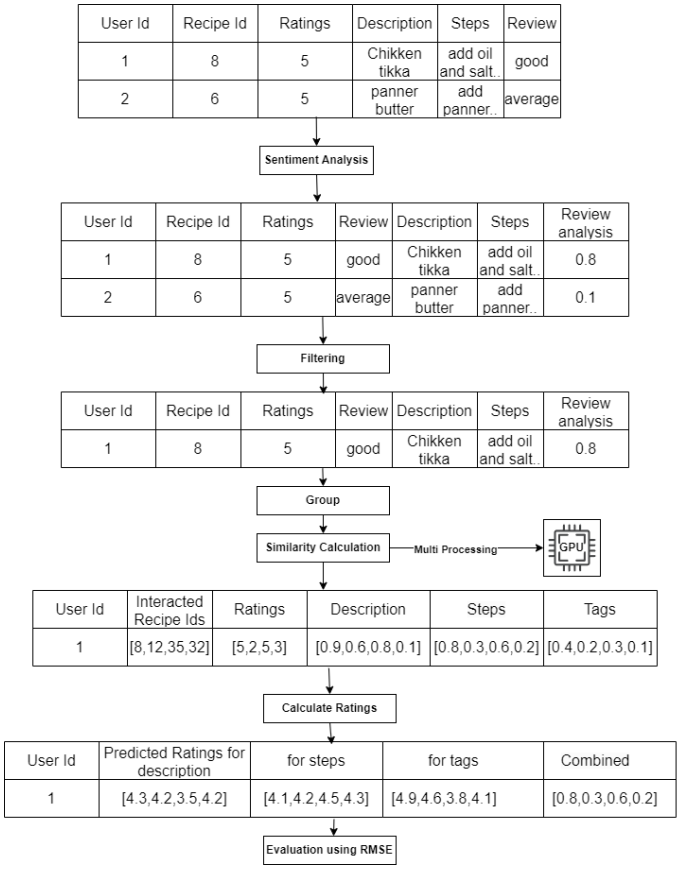

Fig. 7. Data flow using any similarity measure (Glove, TF-IDF, BM25, Spacy)

III. Experiments / Proof of concept evaluation

A. Dataset

Name - Food.com Recipes and Interactions
Source - https://www.kaggle.com/code/ngohoantamhuy/food-recommendation-systems
The dataset basically consists of the users, recipes, and their corresponding ratings, and reviews. It also consists of other metadata which is related to the recipes of the food such as recipes description, tags, steps involved, etc.
1. RAW_interactions.csv - user_id(int), recipe_id(int), date(int), rating(int), review(text).
Number of rows - 1125284
Number of columns - 5
2. RAW_recipe.csv - name(text), id(int), minutes(int), contributor_id(int), submitted(int), tags(text), nutrition(text), n_steps(int), steps(text), description(text), ingredients(text), n_ingredients(int).
Number of rows - 231638
Number of columns - 12
3. PP_users - techniques(int), items(int), n_items(int), ratings(int), n_ratings(int).
Number of rows - 25066
Number of columns - 6
4. PP_recipes - id(int), i(int), name_tokens(list¡int¿), step_tokens(list¡int¿), techniques(list¡int¿), calorie_level(int), t_ids(list¡int¿).
Number of rows - 178091
Number of columns - 8
5. interactions_validation - user_id(int), recipe_id(int), date(date), rating(int), u(int), i(int)
Number of rows - 536146
Number of columns - 6
6. interactions_train - user_id(int), recipe_id(int), date(date), rating(int), u(int), i(int)
Number of rows - 537439
Number of columns - 6
7. interactions_test - user_id(int), recipe_id(int), date(date), rating(int), u(int), i(int)

Number of rows - 538038
Number of columns - 6

B. Pre-Processing/EDA

We are using 2 tables of raw interactions and raw recipes. We have visualised all the columns in these tables and we can see that reviews and descriptions have Nan values. When we look at the bar charts and box plots for the columns we can see that majority of the data are close to each other but there are some outliers. When we look at the rating distribution we can see that the majority of the ratings are 5 so this may affect our predictions.

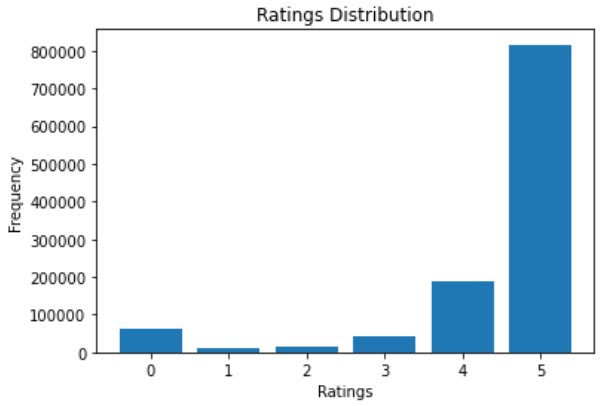

Fig. 8. Ratings Distribution

Sentimental Analysis is being used to find the sentiment of the reviews as it should be closely related to the ratings. We are using the Twitter-roBERTa-base model for finding the sentiment. It is trained on 58M tweets and fine-tuned. It finds the probability for each label and returns the name of the label with the highest probability. We then are using this sentiment data for training neural networks.

C. Evaluation methodology

We are using Mean Squared Error to find the error of each model. We also use cross-validation methodology in grid search for fine-tuning baseline SVD.
Used train-test split for the neural networks with the ratio being 4:1. Calculating the RMSE on the test data.

D. Graphs showing different parameters

From the charts [Fig.9 and Fig.10], we can see that Glove and Spacy as they use semantic scores and we get more information from them. TF-IDF and BM25 give the worst results because they are only based on simple word document similarity.

IV. Discussion and Conclusions

We made the decision on concentrating on the textual columns more and concentrated on the similarities, in the end, it worked out well since the models such as Glove and Spacy are giving out better results. The main difficulty was to create embeddings and perform sentiment analysis since

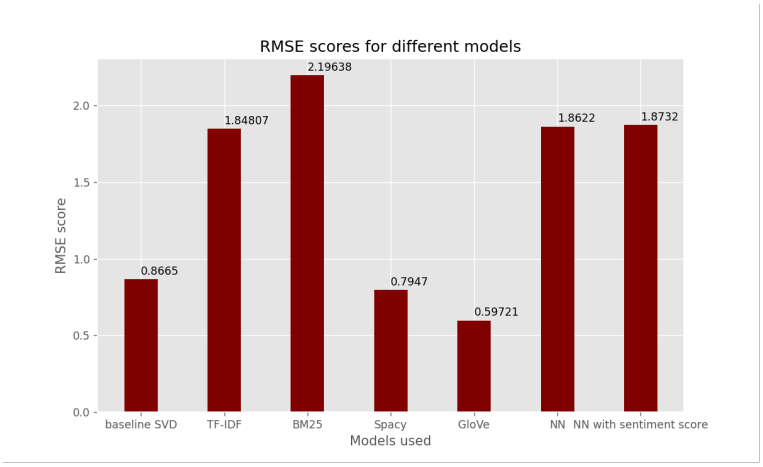

Fig. 9. Graph showing RMSE scores of different models

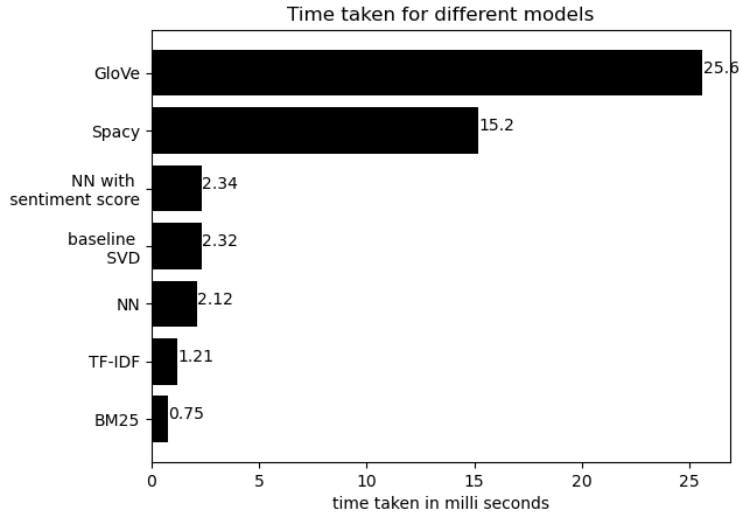

Fig. 10. Time taken to predict ratings in milliseconds for each model

there are a lot of textual values with an average length of 20 words per row. Handling the imbalance in the data was another difficulty but we tackled it using sentiment analysis. The arithmetic throughput and memory bandwidth of GPUs are well into the teraflops and hundreds of gigabytes per second. However, implementing algorithms that approach these performance levels is complex and counter-intuitive. By using FAISS Algorithm we presented the algorithmic structure of similarity search methods that achieves near-optimal performance on GPUs. FAISS makes it possible to do a similarity search in less time than a CPU would take to do this approximately. Although the Sentiment analysis was used as an embedding to the neural network it didn't perform well, as most of the reviews, were classified to be neutral.

V. Project Plan / Task Distribution

Ketan Malempati - EDA, Sentimental Analysis, TF-IDF, BM25.

Pruthvi Raj Kotigari - EDA, Glove, SVD, Spacy
Rahul Raghava Peela - EDA, Neural Networks, Neural Networks with sentimental analysis
Dev Savsani - EDA, FAISS
Everyone performed the respective tasks that were assigned. We were initially taught of using one model each but later wanted to experiment more and analyze other models so we did extra methodologies such as sentiment analysis, spacy, bm25.

REFERENCES

[1] M. Cory. "TF IDF." https://towardsdatascience.com/natural-language-processing-feature-engineering-using-tf-idf-e8b9d00e7e76 (accessed Apr. 21, 2023).
[2] "Understanding TF-IDF." Sefidian.com. http://www.sefidian.com/2022/07/28/understanding-tf-idf-with-python-example/ (accessed Apr. 21, 2023).
[3] "Rank-BM25: A two line search engine." pypi.org. https://pypi.org/project/rank-bm25/ (accessed Apr. 21, 2023).
[4] "Twitter-roBERTa-base for Sentiment Analysis." huggingface.co. https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment (accessed Apr. 21, 2023).
[5] Loana. "Latent Semantic Analysis: intuition, math, implementation." https://towardsdatascience.com/latent-semantic-analysis-intuition-math-implementation-a194aff870f8 (accessed Apr. 23, 2023).
[6] J. Pennington, Glove: Global vectors for word representation. [Online]. Available: https://nlp.stanford.edu/projects/glove/. [Accessed: 30-Apr-2023].