Flavour Fusion Report

***Abstract*—A problem many people face when they have limited ingredients at home and feel discouraged to cook, leading them to order food online. The solution proposed is a system that recommends dishes based on the ingredients that the user has at hand, promoting healthy eating habits and financial management skills, and introducing users to new recipes and cuisines. The report outlines the system's development process, including data collection, algorithm design, and presents the results of its evaluation through user testing data. Ultimately, the report emphasizes the potential of technology to support home cooking and promote healthy eating habits.**

I. Introduction

Whenever we want to make food at home we get discouraged due to not knowing the recipes. At the end we decide to order food from online outlets. Our product Flavour Fusion tries to reduce this problem by providing and suggesting the recipes with the ingredients you have available. The suggestion of these recipes is given based on the ingredients that you provide and some popular recipes that you have rated. We use these information to predict the best dishes for you. The system we propose also recommends based on the nutrition setting you choose. If you want more healthy food with less emphasis on taste then we suggest you food items based on that.

Te use machine learning techniques like Word2Vec and Deep neural networks to identify the best recipes based on the user preferences. The the system uses the description of recipes with ingredients list to make the recipes embedding. For inference and testing we take the user embedding based on the recipes that they rate and use that system to suggest the top recipes for you. The system then filters out the recipes based on the ingredients that you provide.

II. Dataset Creation

Dataset we used was publicly available dataset on kaggle.com. The dataset is based on the information and user ratings based on Food.com website. Dataset had 180K+ recipes and 700K+ recipe reviews. Covering about 18 years of user interactions. This data set is very huge and good for data analysis. We applied complex models on this large dataset to get our results.

We do not require any data creation as the dataset provided in [1] is enough for any practical purposes and we did not require any further dataset.

III. Preprocessing Steps

The basic preprocessing steps that we took on the dataset are -

- The users didn't give enough ratings to many of the recipes. A lot of the recipes had very less number of user reviews thus making them harder to analyze or include in the models based on the ratings. So we decided to take the top 20 most commonly rated recipes as the user representation. This gives us sufficient metric to run the model. So every user is just a embedding of the 20 most commonly rated recipes.
- Most of the features such as ingredients list or the description of the recipes that was taken into account is in the text form thus we did not perform any kind of analysis on them.
- The rating given by each user is a feature that we can analyze statistically. Thus we used standard scaler on the rating data as the data was skewed. The rating were very highly inclined over the rating 5 (rating is given b/w 1-5). So we applied standard scaler in order to make the mean of the ratings to be 0. Please refer to the graphs above.

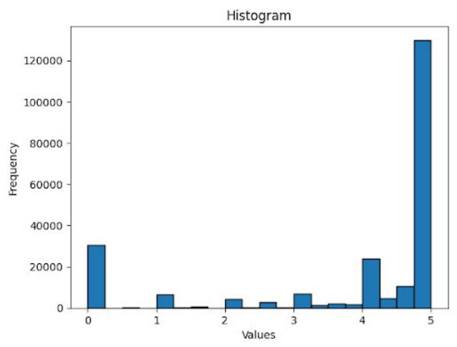


Fig. 1. Graph before using standard scaler

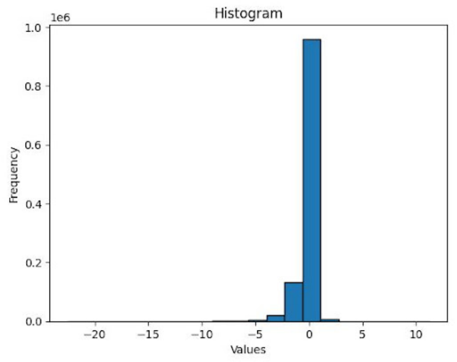


Fig. 2. Graph after using standard scaler

IV. Methods

The Food.com dataset is a comprehensive collection of recipe and user rating information that can be used to build

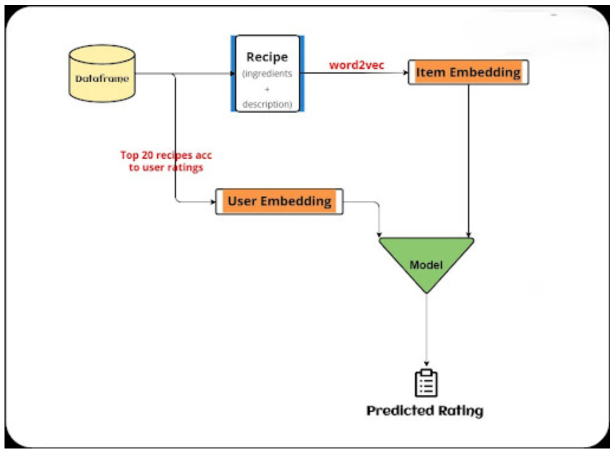
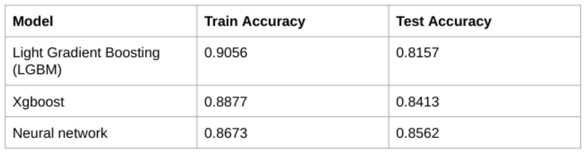
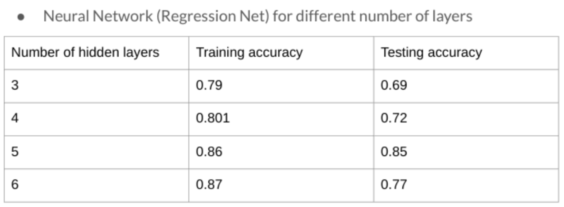

Dataframe
Recipe (ingredients + description)
word2vec
Item Embedding
Top 20 recipes acc to user ratings
User Embedding
Model
Predicted Rating

Fig. 3. Pipeline



Model	Train Accuracy	Test Accuracy
Light Gradient Boosting (LGBM)	0.9056	0.8157
Xgboost	0.8877	0.8413
Neural network	0.8673	0.8562

Fig. 4. Neural Network Comparison with trees



- Neural Network (Regression Net) for different number of layers

Number of hidden layers	Training accuracy	Testing accuracy
3	0.79	0.69
4	0.801	0.72
5	0.86	0.85
6	0.87	0.77

Fig. 5. Neural Network Comparison

a recipe recommendation system. The dataset contains over 180k recipes, each with a corresponding rating from users who have tried the recipe. The goal of this recommendation system is to provide users with personalized recipe recommendations based on their past preferences.

To achieve this, the first step is to filter the dataset by taking the top 20 rated recipes from all recipes. This approach is taken as the number of user ratings available for all recipes is very low, but for the top 20, it is reasonable. The next step is to extract the ingredients and description data from the recipe, which will be used to create item embeddings.

Once the recipe data has been preprocessed, the next step is to create user embeddings. To achieve this, we ask the user to rate their top 20 favorite recipes. These 20 ratings will be used as the user representation. This user representation is then used to compare against other recipe embeddings in order to create a personalized ranking of recipe recommendations for the user.

A. Neural Networks

The user representation and the recipe embeddings are then appended and passed through a deep neural network. This network is responsible for predicting a rating for an input recipe. This process is repeated for all the recipes that the user has rated, and the model is trained on this data.

We train the model for all the recipe that the user has rated. So we append the user representation with the embeddings of the recipe that are actually rated by that particular user. This creates a rating for user and recipe.

After the model has been trained, the next step is to use it for inference. This is done by taking all the recipes in the dataset and comparing their embeddings to the user embedding. The similarity between the embeddings is used to create a personalized ranking of recipe recommendations for the user. For every recipe we get the ratings as we predict using the embeddings of user appended to every recipe embedding. Then we get score related to every embedding which we use to rank the recipe.

The final step in the process is to filter the recipe recommendations based on the ingredients provided by the user. This ensures that the recommendations are relevant and feasible for the user based on their available ingredients.

B. Why neural Networks (Comparision with tree based methods)

Tree-based algorithms might have problems capturing complicated relationships in the data since they split the data using a set of straightforward if-then rules. Tree-based algorithms typically overfit the data, which results in lower testing accuracy even if they can have better training accuracy. Contrarily, neural networks are better able to generalise to new data, which makes them more appropriate for recommendation systems that aim to deliver precise suggestions for novel, untested recipes.

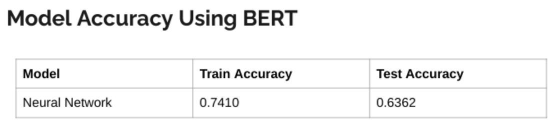
Additionally, user ratings may be included into the neural network-based algorithms used in the recipe recommender as part of the learning process, which can increase the precision of the suggestions. In contrast, user ratings cannot be easily included into the model using tree-based algorithms. Overall, neural networks are a superior option than tree-based algorithms for the recipe recommender since they can learn complicated correlations and generalise effectively to new data, as well as because they can take into account user evaluations.

C. Why we did not use BERT

We did the experimentation with using transformer like BERT instead of using word2vec. But the result we got was against transformer. The code we used for transformers is -

```
from transformers import AutoTokenizer,
    AutoModel

tokenizer = AutoTokenizer.from_pretrained
    (”bert-base-uncased”)
```


Model Accuracy Using BERT

Model	Train Accuracy	Test Accuracy
Neural Network	0.7410	0.6362

Fig. 6. Using Bert instead of word2vec

```
model = AutoModel.from_pretrained("bert-
    base-uncased")
df1['ingredients_str'] = df1['ingredients
    '].apply(lambda x: ' '.join(x))
df1['text'] = df1['ingredients_str'] + '
    ' + df1['description']
def tokenize(text):
    if isinstance(text, str) and len(text
        ) > 0:
        return tokenizer.encode(text,
            add_special_tokens=True)
    else:
        return []
df1['tokens'] = df1['text'].apply(
    tokenize)
df1['tensors'] = df1['tokens'].apply(
    lambda x: torch.tensor(x))
```

The above code actually gives a bad result

Word2Vec and transformers like BERT are both popular methods for generating word embeddings. However, in the given pipeline for recipe recommendation, it can be argued that Word2Vec is a better choice.

The reason for this is that Word2Vec is specifically designed to capture the semantic relationships between words, which is precisely what is needed for this task. Word2Vec is a shallow neural network that is trained to predict a word given its neighboring words in a corpus. In the process, it learns to represent words as vectors in a high-dimensional space, such that words with similar contexts have similar vectors.

On the other hand, transformers like BERT are designed to capture contextual information in text. They use a deep neural network with attention mechanisms to generate word embeddings that take into account the context in which the word appears. While this is useful for many natural language processing tasks, it may not be necessary for recipe recommendation, where the interconnections between ingredients and recipe descriptions are more important than their precise context.

Another advantage of Word2Vec is that it is computationally less expensive than transformers like BERT. Training a BERT model requires large amounts of computational resources and can take a long time. Word2Vec, on the other hand, can be trained relatively quickly and with less computational resources.

In addition, Word2Vec is a more interpretable method compared to transformers. The resulting word embeddings have a clear geometric interpretation in a high-dimensional space, which makes it easier to understand the relationships between words. On the other hand, the internal workings of transformers are more complex and less interpretable.

Overall, while transformers like BERT have shown impressive performance in many natural language processing tasks, for the given pipeline of recipe recommendation, Word2Vec is a better choice due to its ability to capture semantic relationships between words, computational efficiency, interpretability, and suitability for the task at hand.

V. Novelty

Our recipe predictor aims to provide personalized recipe recommendations to users based on their preferences, dietary choices, and time constraints. To enhance the novelty of our idea, we have incorporated four additional features to make our predictor more useful and versatile.

Firstly, we have added a nutrition score for each recipe, which takes into account the nutritional value of the ingredients used in the recipe. This score is calculated using a predefined algorithm that considers factors such as calorie content, protein, carbohydrates, and fats. The nutrition score is then used to rank the predicted recipes, which ensures that users get healthy and nutritious recommendations.

Secondly, we have introduced the concept of diet choices, where users can select their dietary preferences such as healthy, balanced, or high-protein diets. Based on the selected diet choice, a weighted addition is made to the nutrition score of the recipes, which favors the nutrients that are important for the chosen diet. For example, for a healthy diet, the weights are more focused on protein than carbohydrates, while for a balanced diet, the weights are evenly distributed.

Thirdly, we have added a nutrition sensitivity meter, which allows users to adjust the level of emphasis on nutrition in the recipe recommendations. The sensitivity meter can take values from 0 to 1, where 0 means the recommendations are based solely on the deep neural network, while 1 means the recommendations are focused entirely on nutrition. A value of 0.5 gives a balance between the two.

Lastly, we have included a filter for time taken to make the dish, where users can specify the time they have available to make the recipe. The database contains information about the time it takes to make each recipe, and thus the predictor can filter out recipes that take too long to prepare, ensuring that users get recipes that can be made within their available time frame.

Overall, these four features make our recipe predictor more personalized and useful for users, providing them with healthy and nutritious recipe recommendations that cater to their dietary preferences and time constraints. It ensures that users can get the most out of their cooking experience, whether they are looking for a quick meal or a nutritious one.

VI. Result Analysis

We are assuming that user have all the ingredients to make these dishes (i.e. there is no filtering of recipes based on ingredients) In case user wants to add their list of ingredients,

they can do that. User with no prefrences:
newuser = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] Nutrition = 0.
The top 20 dishes recommended was: ['jilllyray s cinnamon rolls', 'pigs in a blanket', 'hot roasted turkey mountain', 'italian sausage onion pizza', '3 ingredient garlic bread', 'crescent wrapped brie', 'garlic dill pickled cucumbers gherkins', 'oven fried bacon no mess no cleanup', 'sausage biscuit balls', 'ham cheese crescent roll ups', 'zepz', 'slow roasted bell peppers', 'ketchup easy quick', 'light south east asian zucchini bake', 'power shake for breakfast', 'turkey taco salad', 'lidia s sausage peppers', 'butterflied grilled chicken with curry and cumin', 'smoked bacon onion sweet cornbread', 'pepperoni stromboli pepperoni bread']
Obsercation: We can say that this dishes are the most neutral dishes which are liked by all kind of people. Let us assume that user only likes the sweet dishes in the 20 dishes. In this case user input will be:
newuser = [5,0,0,0,5,0,0,5,0,0,0,0,5,0,5,0,0,0,0,0], Nutrition = 0.
The top 20 dishes recommended was: ['sugar strawberries', 'roasted fresh chilies like poblanos jalapenos bell peppers', 'dressed egg salad', 'smoked fattie', 'manchego chorizo puffs', 'biscuits in a nuwave', 'maple candied sweet potatoes', 'corn chorizo tacos', 'pesto pinwheels', 'hot roasted turkey mountain', 'italian sausage onion pizza', 'shrimp crescent appetizers', 'fried bacon in the oven', 'caramelized onion gorgonzola pizza', 'sweet pepper relish', 'chorizo sausage chihuahua dogs', 'mildreds s cranberries', 'carrot and celery juice', 'grilled sweetcorn', 'jordanian herbal tea']
Observation: Giving higher rating to sweet dishes gives a bias towards sweet dishes. And we can notice this observation in the output also. All these dishes are mostly sweet and have higher position than other dishes.
Let us assume that user likes all the dishes:
newuser = [5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5] The top 20 dishes recommended was: ['pigs in a blanket', 'biscuits in a nuwave', 'sizzler s cheese toast', 'blueberry syrup pancakes nigella lawson', 'bacon bits', 'sugar strawberries', 'st vincent sweet potato chips', 'raspberry iced tea', 'jilllyray s cinnamon rolls', 'baked havarti', 'polish sausage in tomato sauce kielbasa w sosie pomidorowym', 'fantasy island cocktail', 'zepz', 'hot pecan peas', 'perfect pineapple pops', 'old fashioned vinegar taffy', 'grape freezer jam', 'fruity fruit dip', 'grilled blue cheese sandwich with seared grapes', 'basic millet']
Observation: Giving all the dishes 5's does not give that much variance to the model. Therefore it have many dishes common with the user giving 0 rating.
Now let us give ingredient list to the model. List contains ingredient for chicken and egg recipes and many other common ingredients.
newuser = newuser = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] :
The top 20 recommendation : ['a quick and different fried fish recipe',' easy rambled eggs ' , 'fluffy oven scrambled eggs' , ' the indian butter chicken' like campbells , ' gluten free tomato 'fried eggs' , 'half and half substitute' , best scrambled eggs' , 'munavoi finnish egg butter', 'homemade whipped for a crowd', ' simple sc soup', 'fried eggs with butter ' , ' perfect onion ukrainian hard boiled eggs russian' technique', 'oven scrambled eggs ' sweating onions '] Observation : This recommends most common and neutral dishes that can be made by eggs and chicken.

References

[1] Food.com Recipes and Interactions Crawled data from Food.com (GeniusKitchen) online recipe aggregator https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions