ELL409 MACHINE INTELLIGENCE AND LEARNING

PROJECT 4 CUISINE PREDICTION

Kamal Nayan (2022EE31195)[1] and Raaginee D Turki(2022EE11681)[2]

[1]Department of Electrical Engineering, Indian Institute of Technology, Delhi
[2]Department of Electrical Engineering, Indian Institute of Technology, Delhi

November 24, 2024

1 Problem Statement

The primary goal of this project is to develop a machine-learning model capable of **classifying the cuisine type of a recipe based on its list of ingredients.** The specific points addressing the problem statement are as follows:

1. *Classification Challenge*: The task involves recognizing and categorizing recipes into different cuisine types, such as Indian, Mexican, Moroccan, and others, solely based on their ingredient composition.

2. *Diversity of Cuisines and Data Representation*: Given the wide variety of global cuisines, the model must differentiate between numerous cuisines with potentially overlapping ingredients or having specific ingredients, making the classification task complex. Our model will preprocess the data in order to address these issues.

3. *Performance Metrics*: The model's success will be evaluated based on its accuracy defined as:

$$Accuracy = (tp + tn)/(tp + tn + fp + fn) \quad (1)$$

2 Proposed Solution

The project will leverage Artificial Neural Networks (ANNs) [1]to address this classification challenge, which is well-suited for identifying patterns in complex datasets. The implementation will utilize popular machine learning frameworks like TensorFlow or PyTorch, providing the flexibility and efficiency needed for model training and evaluation.

The data (JSON format) includes recipe IDs, cuisines (training set), and ingredient lists. Ingredients are preprocessed into numerical vectors (here we used TF-IDF). The ANN model has an input layer (ingredients), hidden layers (feature extraction), and an output layer (cuisine prediction).

The model is trained on the training dataset using cross-entropy loss and optimization techniques(Adam). A validation split ensures model performance is robust.

3 Experiments Conducted

3.1 Data Preprocessing

3.1.1 Cuisine Count Table

Cuisine	Count
Italian	5910
Mexican	4784
southern US	3279
Indian	2249
French	2009
Chinese	1980
Cajun Creole	1159
Thai	1146
Japanese	1055
Greek	875
Spanish	748
Moroccan	630
Vietnamese	618
Korean	609
British	586
Filipino	553
Irish	485
Jamaican	408
Russian	348
Brazilian	343

Table 1: Cuisine distribution table.

Number of recipes in the dataset: 29,474
Types of cuisine in the dataset: 20

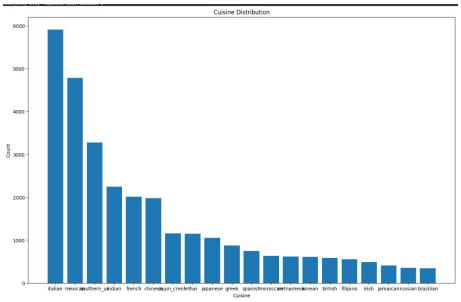

Figure 1: Initial dataset representation

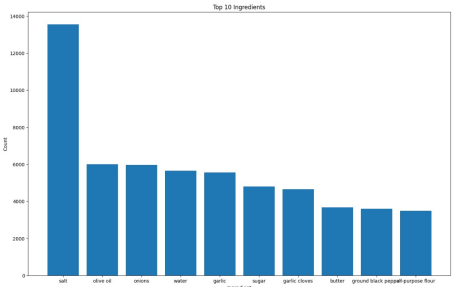

Figure 2: Top 10 ingredients in the dataset

Through plotting the data, we realized that some ingredients, like salt, onions, water, etc., are present in a lot of recipes, and therefore, we applied the following strategies:

1. TF-IDF: TF-IDF[3] provides a more informative feature representation than raw one-hot encoding or bag-of-words, enabling the ANN to better capture relationships between ingredients and cuisines.

2. Data Augmentation:

   Minority Sampling: If specific cuisines (e.g., Indian or Mexican) are overrepresented in the dataset, data augmentation creates synthetic examples for underrepresented cuisines, ensuring the ANN learns equally for all classes.

   Increasing Training Data: Data augmentation artificially increases the dataset size, providing the ANN with more samples to learn patterns, especially when training data is limited or imbalanced.

3.2 Activation function for Neural Network

Leaky ReLU [4]is a variation of ReLU that allows a slight, non-zero gradient for negative inputs.
Why it was used:

1. Prevents dying neurons by maintaining small gradients for negative inputs.

2. Improves learning in deep networks, especially when dealing with sparse input data like ingredient embeddings or TF-IDF vectors.

3. More robust to vanishing gradients than sigmoid, which squashes values into the (0,1) range, leading to slower training.

3.3 When to stop training and Overfitting issues

1. Why Early Stopping Was Used:

   (a) Prevention of Overfitting[5] :

   Early stopping monitors validation performance during training. Training halts when validation accuracy stops improving for a set number of epochs. This ensures the model does not overfit the training data, which is critical when dealing with high-dimensional inputs like TF-IDF vectors or sparse data.

   (b) Efficient Resource Usage:

   Saves computational resources by avoiding unnecessary training epochs, especially when the model has already reached its optimal performance.

   Early stopping restores the model weights to the epoch where validation performance was optimal, improving generalisation on unseen test data.

2. Batch Normalization: By normalizing the input to each layer to have zero mean and unit variance, BN ensures that the distribution of activations remains stable throughout training, leading to faster convergence.

4 Discussion

4.1 Training loss, Training Accuracy, Validation accuracy

Training loss: 0.3100
Training Accuracy: 90.06
Validation Loss: 0.7843
Validation Accuracy: 81
Training will stop at epoch 13.

5 Conclusion

In this project, we successfully developed an Artificial Neural Network (ANN) model to predict a given recipe's cuisine based on its ingredient

```
Epoch 1, Train_Loss: 0.7918, Train Acc: 77.06%,Val Loss: 0.6467 Val Acc: 81.12%
Epoch 2, Train_Loss: 0.6048, Train Acc: 81.91%,Val Loss: 0.6393 Val Acc: 81.55%
Epoch 3, Train_Loss: 0.5459, Train Acc: 83.42%,Val Loss: 0.6260 Val Acc: 82.15%
Epoch 4, Train_Loss: 0.5056, Train Acc: 84.67%,Val Loss: 0.6466 Val Acc: 81.71%
Epoch 5, Train_Loss: 0.4710, Train Acc: 85.53%,Val Loss: 0.6321 Val Acc: 82.01%
Epoch 6, Train_Loss: 0.4397, Train Acc: 86.19%,Val Loss: 0.6624 Val Acc: 81.44%
Epoch 7, Train_Loss: 0.4166, Train Acc: 87.04%,Val Loss: 0.6536 Val Acc: 82.23%
Epoch 8, Train_Loss: 0.3904, Train Acc: 87.73%,Val Loss: 0.6736 Val Acc: 81.89%
Epoch 9, Train_Loss: 0.3720, Train Acc: 88.24%,Val Loss: 0.6919 Val Acc: 81.67%
Epoch 10, Train_Loss: 0.3552, Train Acc: 88.58%,Val Loss: 0.7030 Val Acc: 81.91%
Epoch 11, Train_Loss: 0.3435, Train Acc: 88.96%,Val Loss: 0.7154 Val Acc: 81.49%
Epoch 12, Train_Loss: 0.3234, Train Acc: 89.55%,Val Loss: 0.7390 Val Acc: 81.91%
Epoch 13, Train_Loss: 0.3100, Train Acc: 90.06%,Val Loss: 0.7843 Val Acc: 81.00%
Epoch 14, Train_Loss: 0.3042, Train Acc: 90.04%,Val Loss: 0.7983 Val Acc: 80.34%
Epoch 15, Train_Loss: 0.2924, Train Acc: 90.52%,Val Loss: 0.8148 Val Acc: 81.02%
Epoch 16, Train_Loss: 0.2822, Train Acc: 90.91%,Val Loss: 0.8361 Val Acc: 81.04%
Epoch 17, Train_Loss: 0.2809, Train Acc: 90.97%,Val Loss: 0.8315 Val Acc: 80.88%
```

Figure 3: Training loss, Training Accuracy, Validation accuracy on 20 epochs

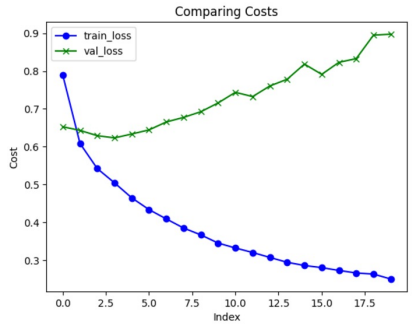

Figure 4: Comparing Validation and Training Loss

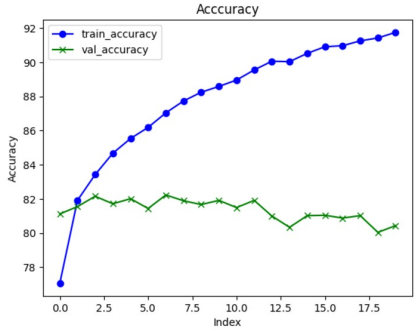

Figure 5: Comparing Validation and Training Accuracy

list. By leveraging frameworks such as TensorFlow or PyTorch, we implemented a robust machine learning pipeline that included preprocessing the dataset, feature extraction using techniques like TF-IDF, and training the model with strategies to prevent overfitting, such as early stopping and data augmentation.

The model was evaluated using the overall accuracy metric, and we achieved consistent performance by balancing the training and validation phases. Leaky ReLU activation functions ensured efficient learning while avoiding the vanishing gradient and dying neuron problems. Data augmentation also helped address class imbalances, leading to better generalization across diverse cuisines.

With these efforts, we were able to generate predictions for the test set in the required format, achieving all deliverables for this assignment. The project demonstrates the power of ANNs in solving multi-class classification problems and provides a scalable framework for handling text-based inputs in future culinary or similar applications.

For the code and dataset refer to this GitHub repository: https://github.com/RDT3612/ELL409-CUISINE-PREDICTION

References

[1] LeCun, Yann, et al. "Deep learning." Nature 521.7553 (2015): 436-444.

[2] Abadi, Martín, et al. "TensorFlow: Large-scale machine learning on heterogeneous systems." (2015).

[3] Ramos, Juan. "Using TF-IDF to determine word relevance in document queries." Proceedings of the first instructional conference on machine learning. Vol. 242. No. 1. 2003.

[4] Xu, Bing, et al. "Empirical evaluation of rectified activations in convolutional network." arXiv preprint arXiv:1505.00853 (2015).

[5] Caruana, Rich, et al. "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping." Advances in neural information processing systems. 2001.