Recipe-to-Chef Prediction Using Natural Language Processing

Ricardo de Jesus Vicente Tavares

Group 1, Instituto Superior Técnico, Universidade de Lisboa

October 2025

1 Introduction

This project tackles the task of automatically linking a recipe to its chef, a challenging problem due to noisy and unbalanced data [4]. We compare classical and deep learning approaches, focusing on how textual features—such as descriptions, steps, and ingredients—contribute to chef classification, and on developing efficient, accurate prediction models.

2 Models

The preprocessing converted the raw CSV into a simplified dataset with only two columns: `chef_id` and `recipe`. List-like fields were parsed and flattened into plain text with consistent formatting. The `steps` lists, in particular, were processed by explicitly inserting the order of the steps (1., 2., 3., ...). Each `recipe` was built by concatenating cleaned fields into a single string separated by semicolons. We implemented and compared three text classification models:

- **Support Vector Classifiers (SVC)** [1]: Recipes were converted into TF-IDF vectors and classified using a linear-kernel SVC.
- **BiLSTM classifiers** [3]: Pretrained `GloVe` embeddings were used to initialize the embedding layer, feeding into a bidirectional LSTM with two layers and dropout.
- **BERT-based classifiers** [2]: Recipes were tokenized with `BertTokenizerFast`, and labels were integer-encoded using `LabelEncoder`.

In addition to baseline versions, each model was also trained with a *data augmentation* variant. The original dataset is indeed unbalanced, with a strong predominance of class 4470 over the others (Appendix: Fig. 3). The augmentation consisted of synonym replacement using `WordNet`, applied only to the training set in order to balance class distributions and increase data diversity. This approach was chosen with particular care to avoid altering ingredient quantities, as these numerical values may reveal distinctive patterns and stylistic choices characteristic of each chef. Thus, during this process, numeric quantities were temporarily masked to avoid altering measurements or amounts in the recipes.

3 Experimental Setup

3.1 Datasets

The original dataset, contained in the `train.csv` file, was first preprocessed and shuffled. After this step, the data was split in a stratified manner into 70% training, 15% validation, and 15% test sets. Each subset was then exported as a separate CSV file to be used by the different algorithms.

3.2 Metrics

Model performance was evaluated primarily using accuracy, computed on both validation and test sets. For BERT-based models, accuracy was obtained via a custom `compute_metrics` function in the Hugging Face `Trainer`, while for BiLSTM and SVC models it was calculated using `sklearn.metrics.accuracy_score`.

3.3 Parameters, Hyperparameters

SVC models used TF-IDF vectors with a maximum of 5000 features and a linear kernel with probability estimates enabled. **BiLSTM** models used pretrained `GloVe` embeddings with 100 dimensions, sequence length of 200 tokens, hidden size of 256, two layers, dropout of 0.3, batch size of 32, and 15 epochs with `Adam` optimizer (learning rate 1×10^{-3}). For **BERT-based classifiers**, we used the `bert-base-uncased` model with a maximum sequence length of 512 tokens, learning rate of 2×10^{-5}, batch size of 8, weight decay of 0.01, and 5 training epochs.

4 Results

Figure 1 shows that the best results were achieved by the BERT model with data augmentation (92.9% accuracy on the validation set and 94.0% accuracy on the test set). Figure 2 presents the results of the class-wise inference results of the best model on the test set. The best predicted classes are `chef_id` 6357 (98.3%), 4470 (97.8%), and 5060 (96.2%).

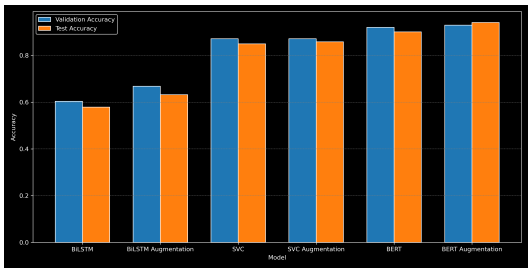

Figure 1: Validation vs. test accuracy per model.

5 Discussion

The input dataset (`train.csv`) consists of 2,999 samples (recipes). In addition to the label `chef_id`, which is an integer identifying a chef, it includes seven features: `recipe_name`, a string representing the title of the culinary recipe; `date`, possibly indicating the date of publication or presentation; `tags`, an array of strings describing the essential characteristics of the recipe; `steps`, another array of strings explaining the preparation steps; `description`, a string containing information, warnings, suggestions, and often *subjective* comments, which can introduce noise or even cause data leakage; `ingredients`, an array with one string per ingredient; and `n_ingredients`, an integer eventually derived from the length of the `ingredients` feature. These features contain fundamental elements for identifying the respective chef, namely identifiers (`tags` and `description`), culinary techniques, and the types and quantities of ingredients (`steps` and `ingredients`).

The dataset exhibits several common data issues, including inconsistent formatting in lists where `tags`, `steps`, and `ingredients` have mixed delimiters or extra characters, quotation and escaping errors from CSV exports, and typos or misspellings (e. g., `water\n`, `butter 13x9x2\"\"`, `transluscent` instead of translucent, `jalapeo` instead of `jalapeño`, `Salt & Pepper` vs. `salt & pepper`, `1 / 4 inch` vs. ¼ `inch`) that can affect NLP models.

Analyzing the results, particularly the confusion matrix, the model mislabels some recipes. For instance, five practical, versatile, homestyle, internationally inspired dishes labelled for chef 3288 were incorrectly assigned to chef 4470, who is mainly practical and creative, focused on quick, accessible homemade comfort food with American and Italian influences.

The model most easily recognized chefs 6357 and 4470 (Fig. 2; App.: Fig. 3). Chef 6357, though less frequent, may reveal a clear, consistent style that aids classification. Chef 4470, more frequent and variable, benefits from many examples that enhance generalization. Overall, both clarity and data volume support learning chef-specific features.

The strong results of the BERT model with data augmentation show effective generalization to unseen data. They confirm BERT's ability to handle noise and inconsistencies while leveraging contextual information to detect subtle stylistic patterns unique to each chef. Moreover, data augmentation did not always mitigate the impact of underrepresented classes, although it did accelerate training (App.: Fig. 4, Fig. 5, Fig. 6, Fig. 7).

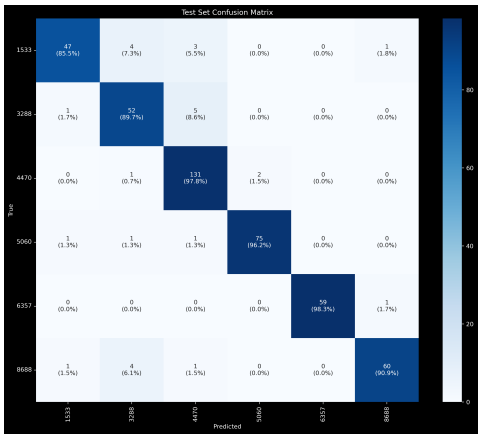

Figure 2: Confusion matrix of model BERT with data augmentation.

6 Future Work

Future work could explore larger pretrained language models, such as GPT or RoBERTa. Incorporating structured recipe features, including ingredient quantities and cooking times, alongside textual data, could enable more robust multimodal modeling. Advanced data augmentation techniques, such as back-translation or contextual paraphrasing, may help increase dataset diversity and balance. Additionally, it is important to understand the contribution of each variable to the model's inference, i.e., a study of feature importance.

Bibliography

References

[1] Constantine Dylan and Abdul Rangkuti. WhatsApp Chatbot Customer Service Using Natural Language Processing and Support Vector Machine. *International Journal of Emerging Technology and Advanced Engineering*, 12:130–136, 03 2022. doi: 10.46338/ijetae0222_15. URL `http://dx.doi.org/10.46338/ijetae0222_15`.

[2] Nadia Gardazi, Ali Daud, Muhammad Malik, Amal Bukhari, Tariq Alsahfi, and Bader Alshemaimri. BERT applications in natural language processing: a review. *Artificial Intelligence Review*, 58, 03 2025. doi: 10.1007/s10462-025-11162-5. URL `https://link.springer.com/article/10.1007/s10462-025-11162-5`.

[3] Ioannis Touloupis. Natural Language Processing and Text Classification with BERT & BiLSTM Model. *Tméma Plerophorikês*, 2024. doi: 10.26267/unipi_dione/4401. URL `https://dione.lib.unipi.gr/xmlui/handle/unipi/16979`.

[4] Apurwa Yadav, Aarshil Patel, and Manan Shah. A comprehensive review on resolving ambiguities in natural language processing. *AI Open*, 2:85–92, 2021. ISSN 2666-6510. doi: 10.1016/j.aiopen.2021.05.001. URL `https://www.sciencedirect.com/science/article/pii/S2666651021000127`.

Appendix A: Extra Figures

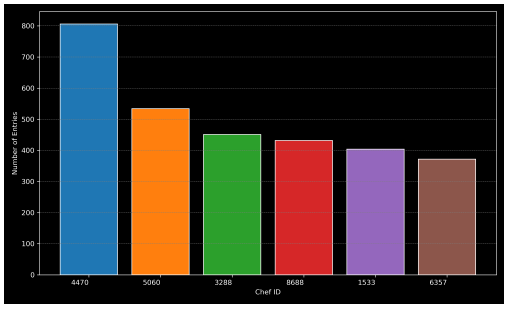

Figure 3: Class distribution in the original dataset.

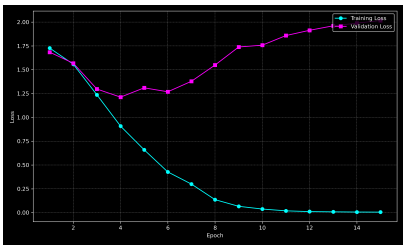

Figure 4: Training and validation loss over epochs for BiLSTM without data augmentation.

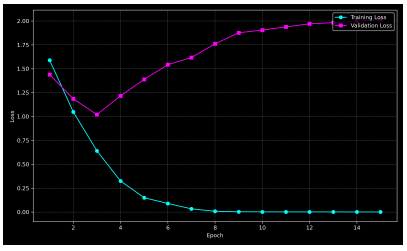

Figure 5: Training and validation loss over epochs for BiLSTM with data augmentation.

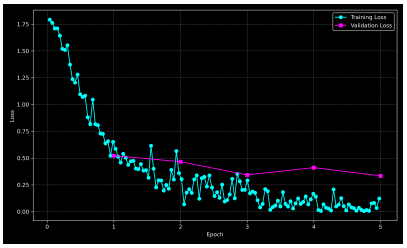

Figure 6: Training and validation loss over epochs for BERT without data augmentation.

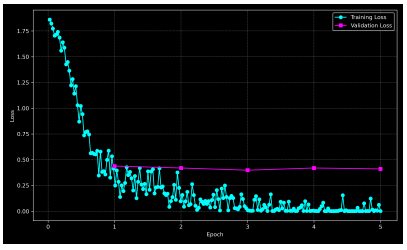

Figure 7: Training and validation loss over epochs for BERT with data augmentation.