Identifying Key Entities in Recipe Data REPORT

1. Problem Statement

Extracting key entities (e.g., ingredients, quantities, and units) from recipe data using Named Entity Recognition (NER) with Conditional Random Fields (CRF).

Objective: To create a structured database of recipes for applications like recipe management systems, dietary tracking apps, or e-commerce platforms.

2. Methodology

2.1 Data Description

Dataset: JSON file containing raw ingredient lists and corresponding NER labels (input and pos).

Key Fields:

- input: Raw ingredient list.
- pos: Corresponding NER labels (e.g., quantity, unit, ingredient).

2.2 Data Preparation

Tokenized the input and pos fields into input_tokens and pos_tokens.

Validated token lengths and cleaned invalid rows.

Split the dataset into 70% training and 30% validation.

2.3 Feature Engineering

Extracted token-level features using spaCy and custom logic:

- Core Features: Token text, lemma, POS tags, dependency relations, etc.
- Quantity and Unit Detection: Used regex patterns and keyword lists.
- Contextual Features: Previous and next tokens, start/end markers.

2.4 Model Training

Trained a CRF model with the following hyperparameters:

- Algorithm: lbfgs
- Regularization: c1=0.5, c2=1.0
- Maximum Iterations: 100

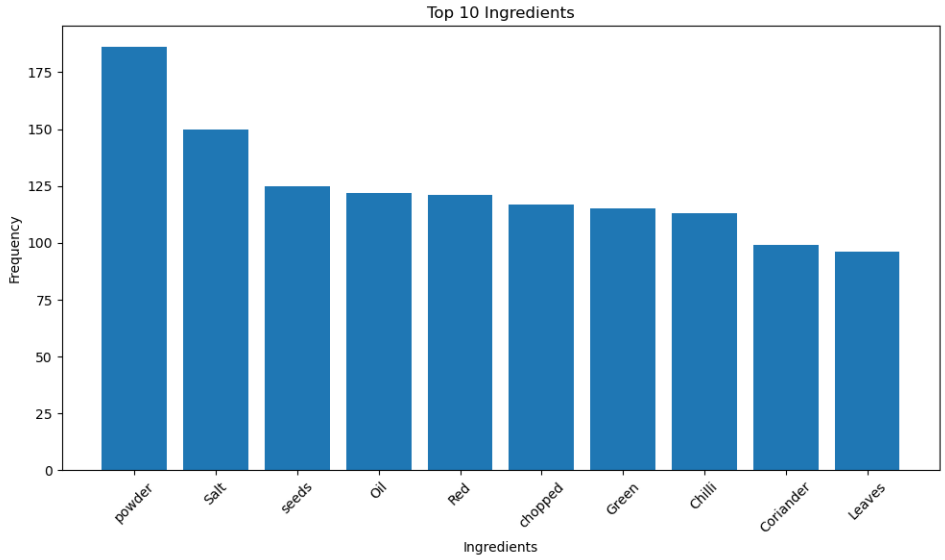
- All Possible Transitions: True

Applied class weights to balance label importance.

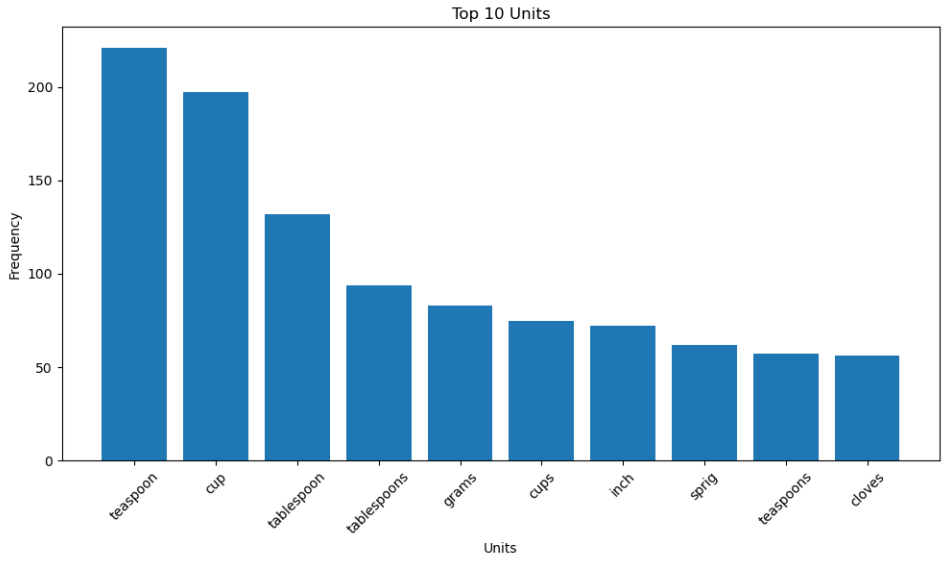
3. Exploratory Data Analysis (EDA)

3.1 Training Dataset

Top 10 Ingredients:



Top 10 Units:



4. Results and Evaluation

4.1 Training Dataset

Flat Classification Report:

Precision, recall, and F1-score for each label (e.g., ingredient, unit, quantity).

High performance for common labels like ingredient, but lower for less frequent labels.

Confusion Matrix:

Shows misclassifications between labels (e.g., confusion between quantity and unit).

```
Confusion Matrix:
Labels: ['unit', 'quantity', 'ingredient']
[[ 562    1    0]
 [   3  655    0]
 [   0    0 3411]]
```

4.2 Validation Dataset

Flat Classification Report:

Similar metrics as the training dataset, with slightly lower performance due to unseen data.

Confusion Matrix:

Highlights areas where the model struggles, such as distinguishing between quantity and unit.


```
Confusion Matrix:
Labels: ['unit', 'quantity', 'ingredient']
[[ 562    1    0]
 [   3  655    0]
 [   0    0 3411]]
```


5. Error Analysis

Misclassified Samples:

Tokens like "little" or "taste" were often misclassified as unit or ingredient.

Common words like "is" were incorrectly tagged as ingredient.

Insights:

1. Confusion Between Labels: Model struggles with ambiguous tokens.

2. Class Weights: Helped balance the model but didn't fully resolve issues with less frequent labels.

3. Contextual Understanding: Errors occurred due to lack of context from surrounding words.

4. Sentence Boundaries: Special markers for start/end helped but weren't always sufficient.

```
Error Analysis by Label:
Label: quantity | Errors: 3 | Class Weight: 7.04
    token true_label  pred_label prev_token next_token  class_weight
0  little   quantity        unit     leaves       Salt      7.039514
1   taste   quantity  ingredient        per        1/2      7.039514
2      is   quantity  ingredient        Pur          2      7.039514
```

6. Key Insights

Strengths:

The CRF model performed well on common labels like ingredient.

Class weights improved balance across labels.

Weaknesses:

Struggled with ambiguous tokens and less frequent labels.

Contextual understanding was limited, leading to misclassifications.

7. Recommendations

Use domain-specific word lists to improve recognition of unit, quantity, and ingredient.

Investigate misclassified tokens (e.g., adjectives, verbs) for further feature engineering.

Explore advanced models like BiLSTM-CRF or transformer-based models for better contextual understanding.

9. Conclusion

Followings are the major insights gathered after evaluating the module on validation data

1. Confusion Between Labels:

The model often mixed up quantities and units. Words like “little” or “taste” were sometimes wrongly tagged.
It also sometimes thought common words like “is” were ingredients, which they are not.

2. Class Weights Helped, But Not Fully:
Giving different importance (weights) to each label helped balance the model.
Still, it had trouble with less common labels like ingredient, so more tuning is needed.

3. Not Enough Context:
Many mistakes happened because the model didn’t understand the meaning from nearby words.
For example, if “taste” came after “for,” it might be easier to tell it’s not a unit or ingredient.

4. Sentence Start and End:
Special markers for the first and last words helped.
But if those words were unclear, the model still sometimes made mistakes.

Enhacement:
Use food-related word lists to help the model know what’s a unit, quantity, or ingredient.
Study the wrong predictions more closely—especially confusing words like adjectives or verbs.