Identify Key Entity in Recipe Data Report

Amit Kumar

1. Problem Statement

The objective of this project is to **automatically identify and tag key entities** in recipe text — specifically:

- **Ingredients** (e.g., tomato, sugar)
- **Units** (e.g., tsp, cup)
- **Quantities** (e.g., 1, few)

This is intended to improve recipe organization, **searchability**, and **user personalization** in platforms like online cooking apps and meal planners, while reducing the burden of manual tagging.

2. Methodology

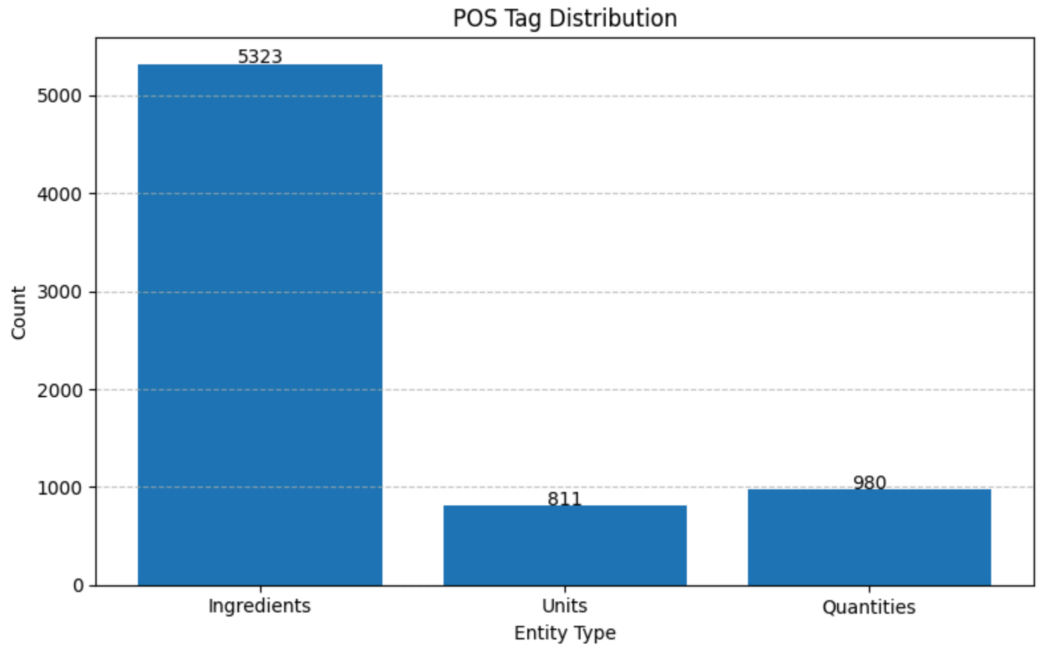
The process followed to address the problem includes:

- Data Ingestion and Preparation
- Exploratory Recipe Data Analysis
- Feature Extraction For CRF Model
- Model Building and Training & Evaluation

3. Exploratory Data Analysis (EDA)

3.1 Class Distribution

The dataset exhibits class imbalance with a higher proportion of ingredient pos tag.



We need to apply class_weight to penalise the ingredient data

3.2 Data Analysis

The analysis of recipe data reveals key facts:

3.3 Techniques Used

- CRF with 'lbfgs' algorithm
- Regex-based quantity patterns (e.g., 1/2, 3.5, few)
- Custom token features (POS tags, dependency tags, casing, digits)
- Class weight simulation for penalizing dominant classes
- Confusion matrix, classification report, and manual error inspection

3.4 Steps

The overall approach followed a standard NER pipeline, using Conditional Random Fields (CRF) to label tokens in recipe text. Major steps included:

- **Data Cleaning & Tokenization**
    - Dropped inconsistent token-tag rows
    - Split text into `input_tokens` and `pos_tokens`
    - Validated token alignment

- **Data Splitting**
    - Dataset split into 70% training and 30% validation
- **Feature Engineering**
    - Used `spaCy` to extract rich lexical and syntactic features
    - Included regex for numbers/fractions, token shape, capitalization, context, etc.
- **Entity Categorization**
    - Labeled tokens with `ingredient`, `unit`, or `quantity` using grouped POS tag

4. Model Building

- **Model**: Conditional Random Field (CRF)
- **Library**: `sklearn-crfsuite`
- **Algorithm**: `'lbfgs'` (quasi-Newton optimization)
- **Regularization**:
    - `c1=0.5` (L1 penalty – feature selection)
    - `c2=1.0` (L2 penalty – overfitting control)
- **Features Used**:
    - Token shape, POS tag, lemma, is_digit, is_title, has_alpha, context words
    - Indicators for units and quantities using regex + keyword dictionaries
- **Custom Class Weights:**
    - Weights were calculated using inverse frequency
    - `ingredient` label was penalized to improve model balance

Model Evaluation

- **Classification Report**:
    - High accuracy for `ingredient` class
    - Moderate performance for `unit` and `quantity` due to variability in expressions
- **Confusion Matrix**:
    - Common misclassifications:
        - `unit` misidentified as `ingredient`
        - `quantity` misidentified due to ambiguous words like “few”, “some”
- **Validation Accuracy:**
    - Accuracy was calculated by comparing flattened predicted and actual labels
    - Misclassified tokens were traced with contextual insights for debugging
- **Error Analysis:**

    - Captured incorrect predictions with surrounding tokens and true vs predicted labels
    - Found patterns of ambiguity, case-sensitivity, and inconsistent token formatting

Key Insights

- **Ingredients are predicted most accurately**, owing to their frequency in training data.
- **Units and quantities** need better normalization and additional training samples.
- **Token context (previous/next word)** is important for accurate predictions.
- Adding semantic features or moving to neural models could improve underperforming labels.
- **Ingredients are predicted most accurately,** with an F1-score of **0.990** due to their frequency and clear context in training data.
- The **quantity** label performs very well (**F1 = 0.987**) but can still be affected by ambiguous terms like "few" or mixed formats like "1/2".
- **Units** show slightly lower performance (**F1 = 0.944**, recall = 0.911) due to their overlapping expressions with ingredients and inconsistent usage (e.g., "tsp", "teaspoon", "spoon").
- The model achieved an overall **accuracy of 98.4%** on the validation dataset.