Analyzing Cooking Recipes to Estimate Preparation Time

Henry Duhaime, Umang Lathia, Logan Light, Christian Kavouras
University of Michigan, EECS 498: Natural Language Processing (Group 17)

Overview

This research aimed to capture the complexity of cooking recipes by analyzing their linguistic structure. After extracting features from a collection of online recipes, we trained a language model to predict the expected preparation time for each training example as a regression problem. On average, our model was accurate within ±48.5% of the correct length of each recipe. Although this significantly outperformed our baseline measurement of predicting the median value in the training set (±125.2%), we expect that better results could be achieved by training on more recipes with high ready in times and by enhancing the set of features.

Introduction

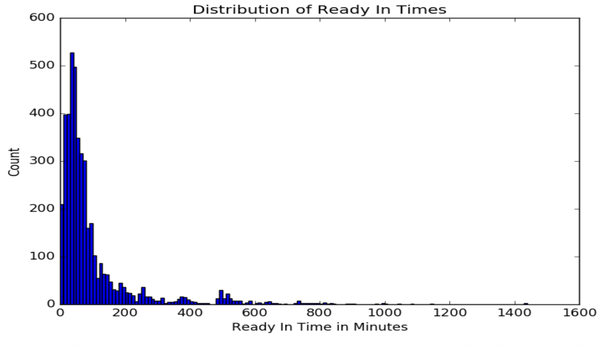
Initially, our research group was curious about the syntactic and semantic composition of cooking recipes. We hypothesized that the broad set of ingredients and common cooking actions could serve as its own language and reveal inherent properties of recipes through further analysis. Recipes tend to include a specific subset of language as they are generally composed exclusively of imperative sentences referencing ingredients as direct objects ("*pour milk*", "*marinate chicken*").

We found that previous work on culinary language models focused on understanding recipes as the aforementioned imperatives [1], machine-interpretable tasks [2], recipe -specific parts-of-speech and term categorization, and synonym pairs for recipe instructions and ingredients [3]. Within the scope of this Natural Language Processing, we sought to determine whether recipe time complexity can be inferred from the language structure of a recipe.

Given a recipe, can we predict the cooking time?

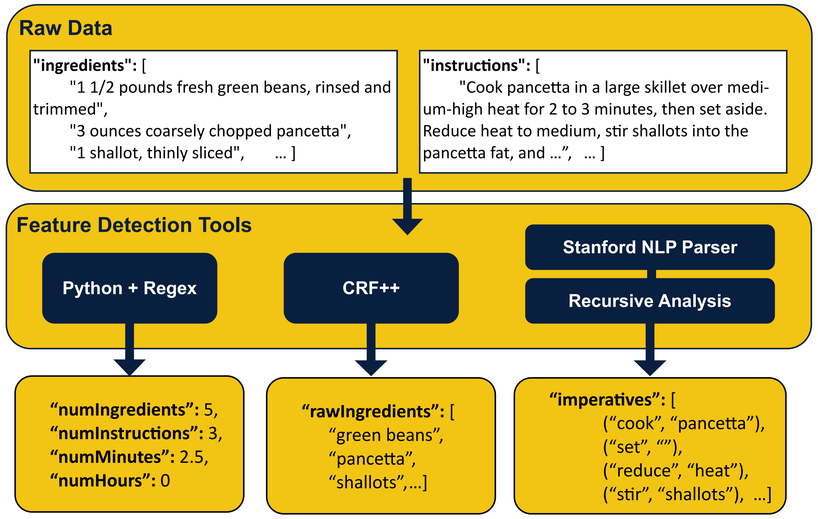
Dataset

We assembled our dataset from over 4,400 recipes extracted from Allrecipes.com using Beautiful Soup 4. In order to be considered, a recipe needed to have the following four fields: ingredients, instructions, "ready-in" time, and a unique recipeID. During the collection process, we often were faced with malformed HTML and inconsistent data entry, which we addressed on a case-by-case basis.



After our first pass over the data, we realized that the distribution of preparation times was skewed significantly to the right. Only a few recipes reached past 48 hours, with one requiring 30 days. To account for this, we limited our training examples to recipes with a preparation time less than 24 hours to mitigate the impact of such outliers on the model.

Features



The diagram above illustrates the features we extracted for our model. Our initial features captured only numerical data about the recipes, such as the number of ingredients or instructions. We then implemented additional parsing to capture and encode time-related phrases such as "3 to 5 minutes" and "a few hours."

To capture information about which raw ingredients were used, we represented them in a bag-of-words model. To extract the raw ingredients from the other data (units, quantities, description), we used a CRF++ tool trained on the NYTimes recipe dataset.

When extracting the imperatives, we initially noticed that many were mistakenly tagged as nouns. To address this, we prepended the pronoun "You" to each clause before tagging it. Thus "cook pancetta" became "you cook pancetta", significantly improving our ability to parse noun-verb relationships.

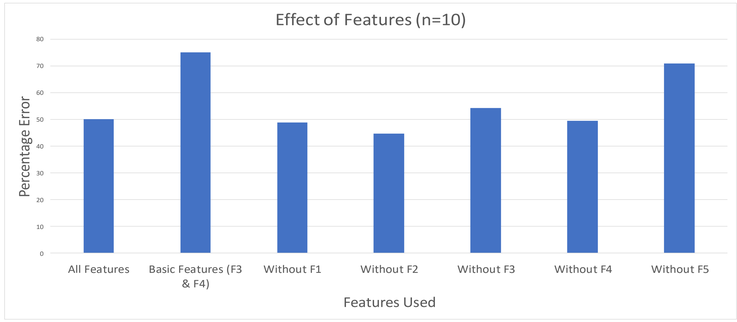
Model

We implemented different regression models to determine which would predict most accurately. We chose regression because we wanted to predict an exact cooking time, instead of classifying into a 'time bucket' SVM was the best by far, guessing 20% of the recipes within 10% of their true time (the threshold we used to determine an 'accurate' prediction).

Our final model used 3 SVM models, each trained on different resamples of the training data. Our prediction was the average of the prediction of these three models, thus allowing each a 'vote' in the true prediction. This allowed us to avoid overfitting and develop a more robust model. We performed hyperparameter tuning and tested the best-tuned model on a validation set using 5-fold cross-validation.

Model (n=10)	Mean % Error	St Dev % Error	R^2	Under 10% Error
Decision Forest	102.02%	304.44	0.554765	14.02%
Gaussian Process	95.49%	24.57	0.001074	0.62%
K Nearest Neighbors	87.73%	226.44	0.637889	13.11%
SVM	48.50%	109.92	0.678725	19.84%

Results



Using the SVM Model with the voting technique described below, we were able to achieve an average error of 48.5%. We chose to use percent error as a metric, as opposed to RMS error or mean difference because we wanted to weigh the error according to the proportion we were off by (i.e. predicting 20 minutes for a 15 minute recipe should be weighed more heavily than predicting 55 minutes for a 60 minute recipe).

Above, we have graphed the percent error on a SVM model using different subsets of the features we extracted. Feature 5, the instruction times in the recipe and feature 3, the number of instructions were the most indicative of cooking time.

F1: Vector of counts of imperatives in recipe
F2: Vector of counts of ingredients in recipe
F3: Number of instructions in recipe
F4: Number of unique ingredients in recipe
F5: Instruction times in recipe

Conclusion

Through our analysis of recipe data, we have learned the problem of predicting preparation time to be complex. However, the results of our model demonstrate that the language structure of recipes informs the preparation time, and more broadly, the complexity of the recipe. After using different feature combinations for testing, we found that features {F1, F3, F4, F5} performed the best. The variation in average accuracy indicates that the number of instructions in the recipe and the instruction times found within the recipe were the most informative to the SVM model. However, we expect that using different features, such as imperative-ingredient bigrams or collecting temperature information, may lead to even lower average error.

With SVM providing a mean error of 48.5% and being within 10% of the true time for 19.84% of our test set, we are confident that we have begun to capture relevant information in our recipe language model based on recipe structure and that we can continue to refine this model with more informative future improvements.

Works Cited

[1] Chloe Kiddon et al. 2015. Mise en Place: Unsupervised Interpretation of Instructional Recipes. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 982-992.
[2] Rahul Agarwal and Kevin Miller. *Information Extraction from Recipes. Tech. Rep. Cs224N-Ling284-AgarwalMiller, Department of Computer Science, Stanford University, Stanford, CA, 2008. 1.*
[3] Jermsak Jermsurawong and Nizar Habash. 2015. Predicting the Structure of Cooking Recipes. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 781-786.