Cuisine and Flavor Classification Using Recipes and Food Images

Yinghui Xia

February 2, 2016

1 Introduction

This project combines recipes, cuisine category, course category, customer ratings, and constructs a system that interprets variation in food habits, flavor and ingredients in different cuisine. Businesses could take advantage of this information to make their recipe and cooking services smarter. Recipe aggregation apps such as Evernote and Paprika could use the result to organize their customers personal recipe boxes into more useful categories. People who care about their food may get some insight of choice of food and eating habits, sort recipes by flavor, salt use, level of spiciness that aren't spelled out in the recipes themselves.
Supervised learning techniques such CART, Naive Bayes, SVM, and deep learning methods for classification are implemented for cuisine classification using recipes as well as food images.

2 Related Work

Japanese researchers Kagaya[3] and Joutou[7] discussed several approaches for food recognition and classification. Kagaya et al. concluded that CNN improves the 10-class recognition performance significantly than using hand-crafted features. Joutou et al. designed hand-crafted food features and implemented a kernel method (multiple kernel learning) and claimed an improvement on a 50-class food classification problem. These two researches focused on Japanese food and had a limited variety of food images. Bossard[4] Food-101 is a comprehensive food image data set that includes 101 popular food categories and 101,000 images in total. Bossard et al. compared a random forest method that mines discriminative components with CNN and SVM. Our image data set is comparable to Food-101 in food variety and data size. My experiment focuses on flavor prediction, which is related to color, shape, cuisine, etc.

One of the earlier works is Kalas[9]. Kalas et al. proposed a personalized recipe recommendation method based on user's food preferences in terms of ingredients, which were derived from user's recipe browsing activities. Su et al.[8] presented a SVM method; Xie et al.[12] proposed a semantic item model for recipe search by example, representing different kinds of features of recipes. Forbes[13] proposed an approach for recipe recommendation to incorporate recipe content into matrix factorization method. Since there are fewer researches conducted on analyzing recipe cuisine and the ingredients, and motivation for swapping ingredients and create new recipes, I investigate if the recipe cuisines can be identified by exploiting the ingredients of recipes.

3 Building Food Images Dataset

3.1 Data collection

I collected 132,938 food images and their corresponding recipe[1]. See detailed information in Appendix.
Another data set that I used contains 40,000 recipes, including the type of cuisine, and the list of ingredients of each recipe (of variable length). By querying the ingredients in the first data set from the latter one, we could label many of the food images into 20 different cuisines. Both data sets are stored in JSON format.

[1] Yummly API, https://developer.yummly.com/documentation

Figure 1: Sample food images

The cuisine type in the recipe data set is not balanced. The distribution of the number of recipes across different cuisine varies. Italian and Mexican food dominate the data set.

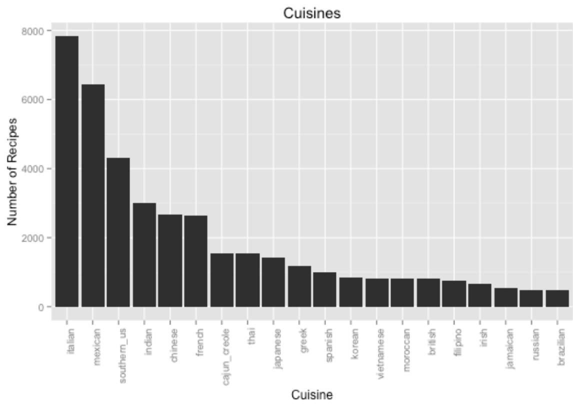

Figure 2: Cuisine Type Distribution

3.2 Bag of Ingredients

There are around 5,000 distinct ingredients in the training set. The frequency of the most common words are showed in Table 1

The next step is add bigrams and trigrams for better representation of gradients.

345 ingredients appear more than 1,000 times and 1,161 ingredients appear more than 100 times in recipes.

	Raw	Count	Processed	Count
1	salt	87760	pepper	34027
2	sugar	68697	salt	30793
3	pepper	64448	oil	29076
4	oil	63085	onion	24065
5	ground	55712	garlic	23624
6	butter	52043	fresh	23482
7	fresh	45728	ground	22826
8	flour	41685	sauce	16490
9	garlic	39473	sugar	15650
10	olive	38539	olive	15276

Table 1: Ingredients Frequency Table

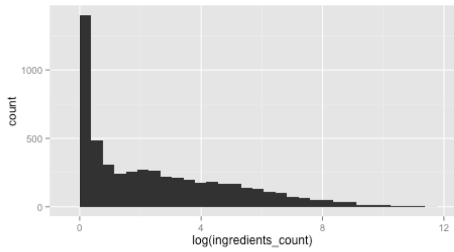

Figure 3: The distribution of ingredients frequency

3.3 Feature Engineering

First, create recipe-ingredients frequency matrix of the data set. Calculate tf-idf score of the matrix. Add the total number of ingredients in each recipe to the tf-idf score matrix. The score matrix of the test set is constructed on top of the vector space model built on training set.

3.4 Word2Vec

Aside from Bag-of-Words model, I also used Word2vec, a neural network implementation that learns distributed representations for words, for comparison. It produces word vectors, where words with similar meanings appear in clusters, and clusters are spaced such that some word relationships, such as analogies, can be reproduced using vector math. Word2vec learns quickly relative to other models and captures intrinsic characteristics that are ignored by Bag-of-Words model, therefore, would be a good experiment in recipe classification.

4 Cuisine Classification

4.1 CART

This is a baseline for my experiments. Tree method is the most straightforward way for cuisine classification. Simply determine the cuisine category based on one or several ingredients. The accuracy on test data is 60.821%. The top words selected by CART based on variable importance are showed in Table 2.

Rank	ingredients	cuisine
1	tortilla	Mexican
2	parmesan	Italian
3	soy sauce	Chinese
4	olive	Italian
5	masala	Indian
7	cilantro	Mecixan
9	fish	Thai
10	cumin	Indian
15	freq	NA
17	mirin	Japanese

Table 2: Cuisine Nodes

Total word count in each recipe is selected as feature, which means feature engineering gives more information than the raw tf-idf matrix.
I choose the size of the tree based on validation error. When cp is smaller than 1e-4, the model starts to overfit. 500 is a reasonable estimate.

4.2 Naive Bayes Classifier

In the Bernoulli model a recipe is represented by a binary vector, which represents a point in the space of ingredients. We have a vocabulary V containing a set of 5,000 ingredients, the t^{th} dimension of a recipe vector corresponds to ingredient w^t in the vocabulary. Let b_i be the feature vector for the i^{th} recipe R_i; then the t^{th} element of b^i, is either 0 or 1 representing the absence or presence of ingredient w_t in the i^{th} recipe.

$$\begin{aligned} &P(R_i|C)P(bi|C) \\ &= \Pi_{t=1}^{|V|} b_{it} P(w_t|C) + (1 - b_{it})(1 - P(w_t|C)) \end{aligned} \tag{1}$$

To classify a recipe R_j, estimate the posterior probability for each class combining equations (1) and Bayes Rule:

$$\begin{aligned} P(C|R_j) &\propto P(b_j|C)P(C) \\ &\propto P(C)\Pi_{t=1}^{|V|} b_{jt} P(w_t|C) + (1 - b_{jt})(1 - P(w_t|C)) \end{aligned} \tag{2}$$

The accuracy on training and test data using a Naive Bayes Classifier is 45.050% and 42.152%.

4.3 SVM

I tested SVM for several motivation. The recipe data set resides in high dimensional input space; document vectors are sparse; text categorization problems are linearly separable. The accuracy is 85.337% on training set and 78.862% on test set. Penalty on error term C is 0.80. The multi-class support is handled according to a one-vs-one scheme.

4.4 Deep Learning&Word2vec

Neural network (2 layers, 256 neurons in hidden layer, objective function softmax, learning rate 0.01) trains for 200 epoch and converges with loss 1e-03. This setting gives some interesting results, e.g. the most significant ingredients for each cuisine, for example, soy sauce, green onions, starch, ginger, sesame oil, rice vinegar, five-spice powder, etc. are the most important features to classify a recipe. The accuracy on test data is 78.912%.

Word2vec gives insights into the data set and finds some useful clusters of ingredients, for example, the model returns a list of ingredients that are similar to 'guacamole', namely, tostada, baked, colby, mexican, salsa, picante, verde, fajita. When performing k-means, Word2vec clearly clusters ingredients together, e.g. Japanese ingredients, mirin, miso, natto, wakame, etc. are clustered in one group.

4.5 Other Experiment: Food Recognition using Deep Learning

The model is is fine-tuned using Alexnet as in Table 3. 80% food images are labeled as training images. An image is claimed to be classified correctly if the highest flavor score associated with the 6 flavors (Spicy, Sweet, Sour, Salty, Savory, Bitter) is identified. The hyperparameters are learning rate

= 1e-07, momentum = 0.9, weight decay = 0.0005.

Layer	Type	Channel	Filter
1	conv+max+norm	96	11*11
2	conv+max+norm	256	5*5
3	conv	384	3*3
4	conv	384	3*3
5	conv+max	256	3*3
6	full	4096	-
7	full	4096	-
8	full	6	-

Table 3: CNN Structure

The model is trained on G2 Instances GRID K520 on AWS. The training time is 7.5 hours and the test accuracy is 63.054%.

Method	Accuracy
Train	72.119
Test	63.054

Table 4: Classification Errors

4.6 Remedy

Overall, the number of cuisine type is not balanced in the training set, which may result in a bias towards cuisine with larger sample in the training data. I approached this problem by bootstrapping from the 14 lowest cuisine types such that all cuisine types have greater than 2000 recipes. Since this technique allows estimation of the sampling distribution of almost any statistic using random sampling methods, it augments cuisine types with fewer recipes and makes it more representative of the ground truth.

4.7 Result

Tree method may misclassify recipes as several cuisines share similar ingredients. A naive Bayes classier is trained using only those features ranked and is highly-biased. It's performance is no better than classifying everything as 'Italian' cuisine. SVM uses most features and improves very much. l_2 regularization is added to SVM to avoid overfitting. NN method learns features that capture the information well. 6-fold cross-validation is performed in SVM and NN model. The CV results of these two models are reported in Train accuracy. NN and SVM gives comparable results, here I only present the confusion matrix for SVM.

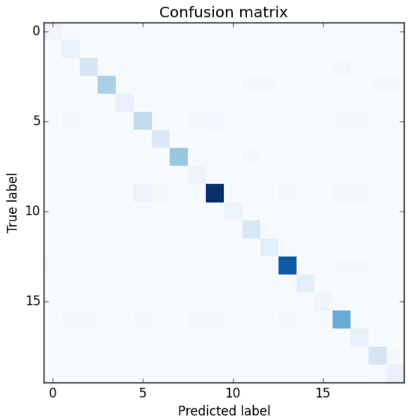

Figure 4: Confusion Matrix for Different Cuisines

Repeat SVM model after bootstrapping, the training accuracy increases but the test accuracy not very much. This could be caused by a bad bootstrapping design. Word2vec gives a slightly better estimates on both training and test sets.

Method	Baseline1	Baseline2	SVM(L2)	NN
Train	67.167	45.050	85.337	82.403
Test	60.821	42.152	78.862	78.912
hline Method	SVM(bs)	Word2vec		
Train	88.320	86.231		
Test	78.911	81.034		

Table 5: Classification Errors

5 Conclusion & Future Work

We compared tree methods, Naive Bayes, Neural Network, SVM based on Bag-of-Words model, and a deep learning based Word2vec model on cuisine classification. SVM with l_2 penalty and Word2vec model give the best results. The accuracy is around 80% across these two models. By treating ingredients as features, we construct classifiers that associate cuisine type and ingredients and make reasonable cuisine type predictions. For flavor prediction problem, I did an experiment using transfer learning from Alexnet. This is a rather tricky question because different user may rate the same food with different scores. Also, training using deep learning method take a huge amount of time and GPU resources, which makes it hard to cross-validate and tune parameters. To combine different data sources and to generate more reliable flavor data set would have a significant impact on the result.

For future work, how to better deal with unbalanced data worth more efforts. One possible strategy to deal with this problem might be classify Italian and Mexican vs all-others; out of all-others, do classification on the rest classes.

Deep learning on the food image data set would have a better performance after more delicate tuning of hyperparameters. Image augmentation, i.e. cropping, flipping, rotation, etc. would generate more variability in the training data.

6 Appendix & Reference

1. Recipe and nutrition information table.

Field	sub1	sub2	sub3	sub4	sub5	sub6
nutrition	calFat	totFat	cal	satFat		
ingredients						
imUrl						
tastes	Spicy	Sweet	Sour	Salty	Savory	Bitter
recipeTitle						
recipeUrl						

Table 6: Recipe and Nutrition Information

6.1 Reference

0. Lamberto Ballan, PhD generously provided Yummly account for image scraping and some of the food images in comparison of his research result. 1. Libraries used: Word2vec library, Caffe.
2. Data source: Yummly API.
3. Kagaya et al. Food Detection and Recognition Using Convolutional Neural Network, 2014.
4. Bossard et al. Food-101 – Mining Discriminative Components with Random Forests, 2014.
5. Farinella et al. A Benchmark Dataset to Study the Representation of Food Images, 2014.
6. Yang et al. Food Recognition Using Statistics of Pairwise Local Features, 2010.
7. Joutou et al. A Food Image Recognition System with Multiple Kernel Learning, 2009.
8. Su et al. Automatic Recipe Cuisine Classification by Ingredients, 2014
9. M. Svensson, K. HooK, and R. Coster, Designing and Evaluating Kalas: A Social Navigation system for food recipes. 2005.
10. M. Uede, M. Takahata, and S. Nakajima, User's Food Preference Extraction for Personalized Cooking Recipe Recommendation, 2011.
11. M. Ueda, M. Takahata, and S. Nakajima, Recipe Recommendation Method Based on User's Food Preferences, 2011.
12. H. Xie, L. Yu, and Q. Li, A hybrid Semantic Item Model for Recipe Search by Example, 2010.