540 iFood Competition Project

Author:Yawen Guo & Negar Erfanian
Rice University

ABSTRACT

In this project, we try to accurately classify the food images.We apply three models: Logistic Regression, DenseNet and Hog feature and Gabor filtering combined SVM models. All models can accurately identify food image. However, all the models are limited to input data size. So we haven't gotten sufficient results. We illustrate our thoughts and methods here, hope there is more time to run our algorithm.

INTRODUCTION

Our data mainly contains 3 sets: train set, validation set, and test set. Each set contains lots of food images from different classes, and except for the test set, the other two sets are labeled. The train set is used for the learning process and the validation set is used for evaluating our learning algorithm to tune the parameters for feature selection. In the meanwhile, we also have to divide a subset to help us with choosing the best classifier among all the classifiers (or stacking).

We first divided the images in the train set in different classes using their labels to analyze the images better. Thus, we have 251 folders (classes) and each folder only contains the images associated with that class. A general analysis of the whole images in the train and validation sets informed us about different image sizes in the first place. This fact tells us that before starting any further analysis we have to resize all the images in different classes and sets to have the same image size for further processes.

METHODS

For our final model we used hog feature and Gabor filtering to extract the textures of images. We tuned the hyperparameters for gabor filters using 10-fold cross validation on the training set. Our final choice for the frequency of Gabor filter was 0.5.
We therefore used PCA with 500 components to only focus on the most pronounced parts of the texture feature for each image. For our final image classifier we used SVM with the rbf kernel.

Challenge

Automatic food identification can be difficult since there are large amounts of food and thousands of different kinds of food in the world. Besides, food doesn't have specific discriminative criteria, and it is hard to tell every ingredient from a cooked dish.
For machines, it is really tricky to tell the class of a dish with only one picture. And pictures sometimes will be influenced by the angle of view, the lighting conditions, but also the very realization of a recipe are among the sources of high intra-class variations. Since the light condition will change the color of the food, the angle to take this picture may influence the shape or size of the food. All these uncertain factors will add complexity for machines to correctly identify the food.

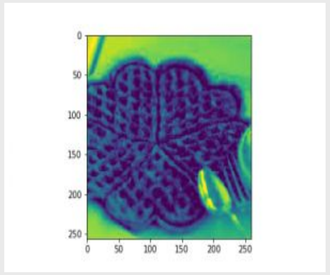

Figure 1. Image Segementation

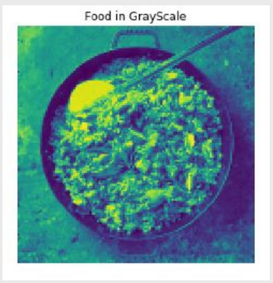

Figure 2. Grey style

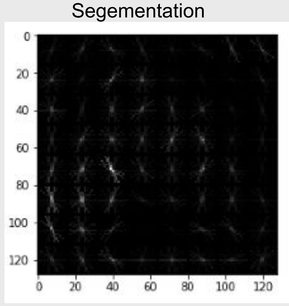

Figure 3 The hog feature

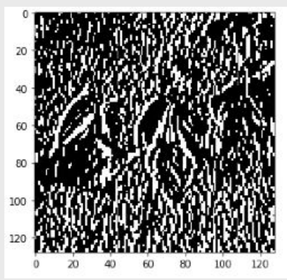

Figure 4 Texture feature.

The comparison of models

We first tried to use densenet as the perfect established model for classifying large classes of images. Due to the large size of data sets and not being able to use cpu for this project, and the lack of sufficient memory while using cuda, we could not work with deep nets. Therefore, we decided to extract features from the data set and use a multiclass classifier to classify our images based on the extracted feature. To do so, we first resized every image to have a 128 by 128 pixels and then made a grayscale of all images. The reason for this was the variety of image size and food colors of images in different and the same classes. Therefore, we realized that color is not a good feature to rely on. We extracted the hog feature of the resized gray-scale images using the hog built-in function in python. We also used the gabor built-in function to extract the texture of each image. Finally we made a vector of combinations of these features for each image. The gabor filter provided a vector of size 32768 for each image and the hog feature provided a vector of size 2916 for each image.

We used the SVC built-in function using python to use SVM as our final classifier. In order to tune the parameters of SVC we used the built-in grid search using python to choose the final classifier with parameters C = 1 and gamma = 0.1. Using this method, we could also control overfitting.
Finally we used the tuned model to train using both train and validation set to finally predict on the unseen test set.