# Yelp Restaurant Attributes Prediction

Group #: 4, Name: DataDogs

Yu Wen, Huijing Zhang, Qian Shen, Shiyu Zhang

# Abstract

Yelp is running a huge platform which many people rely on to find great local restaurants. It hosts tons of millions of restaurant photos uploaded by Yelp users. From the perspective of Yelp, if it would like to enhance the quality of service, it needs to mine these photos thoroughly. The more detailed and accurate restaurant information it is able to extract the better. To address this challenge, we would like to build a restaurant photo classifier which takes a restaurant photo as input, and automatically tags each restaurant photo with multiple business attributes. After building the classifier using training data, the classifier is able to predict business attributes for each restaurant photo in an efficient and accurate way.

# 1. Introduction of the overall goal and background

Most people use Yelp to locate desired restaurants, write reviews, and upload restaurant photos to Yelp via their mobile devices. Knowing the business attributes or labels such as whether this restaurant is expensive can be a great convenience for customer to find appropriate restaurants. To Yelp users, when looking for certain category restaurants, these restaurant labels will help them quickly find out the ones that satisfy their requirement. There is no doubt that, classifying restaurants into different categories can better serve users’ request in a more efficient way.

Currently, the restaurant labels can only be manually selected by Yelp users while submitting a review, however, the accuracy of manually selected features is not reliable. Since this selection is optional, some users may not be willing to select at all or forget to select the attribute. It is also possible that, users select a feature at will without careful consideration because of limited time or other issues. Or, in a worse case, some customers have strong subjective bias. According to Yelp researchers, there are only a small number of users who would like to category restaurant photos uploaded. As a result, many restaurant with a lot of uploaded photos are not or partially classified.

However, appropriate classification of these restaurant photos plays a significant role from the website’s side perspective. Lacking thorough analysis of attributes for each restaurant photos will lead to poor website performance. In a worse case, users may abandon Yelp if all the restaurants are listed without providing detailed and accurate category information, or their categories are presented in a mass.

Luckily, Yelp has stored tens of millions of photos shared by Yelpers all over the world. If we can mining accurate information from these reliable objective data, we can obtain the goal of knowing accurate business attributes for restaurants.

For now, we are focusing on 9 “important” business attribute, 1) good for lunch, 2) good for dinner, 3) takes reservations, 4) outdoor seating, 5) restaurant is expensive, 6) has alcohol, 7) ambience is classy, 8) has table service, 9) good for kids.

In this project, we would like to get rid of manually labeling from Yelp users. Instead, develop a data mining technique to build a restaurant photo classifier. This classifier is responsible for taking a restaurant photo as input, and automatically attaching some of the above-mentioned 9 business attributes to it based on the algorithm developed. Our expectation is that, after building the classifier using training data, the classifier is able to predict business attribute for each restaurant photo in an accurate way. With the help of this classifier, Yelp can gain a more detailed and precise analysis over these uploaded restaurant photos and enhance the quality of service.

# 2. Problem definition and formalization

In this project, the model we would like to build takes a bunch of images for one specific restaurant and make predictions of the 9 business attributes mentioned above.

For example, there are 233 images of restaurant *MexicanRolls.* Then we input these 233 images into our model and get the set of business attributes of *MexicanRolls* as {1 3 5 8}, which indicates that has the attributes 1) good for lunch, 3) takes reservations, 5) restaurant is expensive 8) has table service.

Note that, for one specific restaurant, the output may contain 0 to 9 attributes. So, this problem is a multi-label classification problem.

# 3. Data description and preprocessing

**3.1 Data description**

In this project, we use the data given by yelp which contains 234,843 photos taken from 2,000 restaurants. Details are as followings,

1. train.csv,

continas a field of “business\_id”, which indicates a restaurant, and a field of “labels”. There are 2000 examples in this file. Here are two examples

|  |  |
| --- | --- |
| business\_id | labels |
| 1000 | 1 2 3 4 5 6 7 |
| 1031 | 6 8 |

1. train\_photo,

an folder contains photos of the training set. There are 234,843 photos of restaurants, each photo has a photo\_id as their file name.

1. train\_photo\_to\_biz\_ids.csv,

this file maps the train\_photo id to business id, which gave us information about for an business\_id in train.csv, which images in train\_photo are taken from this restaurant.

1. test\_photo,

an folder contains photos of the test set. There are 7GB photos of restaurants, each photo has a photo\_id

1. test\_photo\_to\_biz\_ids.csv,

maps the test\_photo id to business id, similar to train\_photo\_to\_biz\_ids.csv

Note that this projects derived from a Kaggle competition, so there are no true label of restaurants in test\_photo\_to\_biz\_ids.csv. If we want to see the performance of our model on test\_photo\_to\_biz\_ids.csv, we have to hand in the prediction results on Kaggle, then , we can see the evaluation given by F1 score.

**3.2 data preprocessing**

Here are the data preprocessing we have used

**3.2.1 Resizing**

 Training images have different resolutions, as a result, we need to resize the images to 256\*256 in advance to ensure each image can have same number of pixels. In this way, this resizing make sure we have the unified input dimension if we use SVM, feed-forward ANN or CNN to build our model and treat each pixel as a feature of training example. We may benefit from resizing images in a parallel fashion using mapreduce. But after we judge and weight the method, we hope to make things simple, so we choose to use shell commands, something like:

---------------------------------------------------------------------------------------------------------------------------------

for name in /path/to/image/\*.JPEG; do

    convert -resize 256\*256\! $name $name

 done

**3.2.1 Computing Image Mean**

 Then what we need to do is to compute the image mean, we think the model requires to subtract the image mean from each image. Meanwhile In practice, subtracting the mean image from a dataset significantly improves classification accuracies. So we have to compute the mean. In the caffeNet, there are tools can help us achieve this.

# 5. Methods description

  After the preprocessing of the image, we need to decide how to train images, this step is very tricky.

Due to our project is a multi-label project, we have to find a way to set a single business id with multiple labels. So we have concluded two solutions, first is to preprocess every image as a single vector and then set labels of every business id by union labels of its sub-images. The second is to confine all images under a single business id into one matrix and train this matrix to get the final label.

 Then we started to do some data preparation.

We want to train our data with two different methods.

I. Model Definition

The first but very difficult way to train our data is to set up a neural network.  As we’re trying to use neural network to solve this project, so we need to get the definition of the model. To go to describe a reference implementation for the approach, we’ve searched several papers. One of the famous paper is proposed by Krizhevsky, Sutskever, and Hinton in their [NIPS 2012 paper] (<http://books.nips.cc/papers/files/nips25/NIPS2012_0534.pdf>).

The network definition follow the protocol that Krizhevsky made. In the protocol, there are several ‘include’ sections specifying either ‘phase:TRAIN’ or ‘phase:TEST’. There sections allow us to define two closely related networks in one file: the network used for training and the network used for testing. There two networks are almost identical sharing all layers except for those marked with ‘include{ phase:TRAIN}’ or ‘include { phase:TEST}’. In this case, only the input layers and the one output layer are different. Hence, we need to lay out a protocol buffer for running the solver. So at this period we have maken a few plans:

1.We will run in batches of 256, and run a total of about 90 epochs(about 450,000 iterations).

2.For every 1,000 iterations, we test the learned net on the validation data.

3.We set the initial learning rate to 0.01, and decrease it every 100,000 iterations.

4.Information will be displayed every 20 iterations.

5.The network will be trained with momentum 0.9 and a weight decay of 0.0005.

6.For every 10,000 iterations, we will take a snapshot of the current status.

II. SVM Classification

Two of the common method to enable the SVM is the 1A1 and 1AA techniques. The 1AA approach represents the earliest and most common SVM multi class approach and involves the division of an N class dataset into N two-class cases. If say the classes of interest in a dessert image include ice cream, water and table, classification would be affected by classifying table against other non-table areas, or water against non-water areas etc. The 1A1 approach on the other hand involves constructing a machine for each pair of classes resulting in N(N-1)/2 machines. When applied to a test point, each classification gives one vote to the winning class and the point is labeled with the class having most votes. This approach can be further modified to give weighting to the voting process.

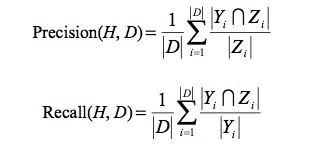
# 6. Experiments design and Evaluation

In our project, first we will extract features from the training set and the testing set by using pre-trained CaffeNet model. We will represent each feature as vector with multi-dimensions. In the training set and the testing set, because each business has multiple photos, then we will compute the mean feature vector among photos that belong to that business. Therefore, each business is correspondent to a single feature.  We will use the training set with the true labels and feature vector to train our model, and predict labels on the test set. Finally, we will use cross-validation method to get average mean F1-Score in order to exam the accuracy of our model.

Here is an explanation of our evaluation method. Mean F1-Score, also known as example-based F-measure in the multi-label learning literature. It considers both the [precision](https://en.wikipedia.org/wiki/Precision_(information_retrieval)) *p* and the [recall](https://en.wikipedia.org/wiki/Recall_(information_retrieval)) *r* of the test to compute the score: *p* is the number of correct positive results divided by the number of all positive results, and *r* is the number of correct positive results divided by the number of positive results that should have been returned.

For this project, we would like to build a restaurant photo classifier which is the multi-label classification problem. Therefore, Let D be a multi-label evaluation data set, L be a label, consisting of |D| multi-label examples (*xi*, *Yi*), i=1…|D|, *Yi* ⊆ *L .* Let H be a multi-label classifier and *Zi* = *H(xi)* be the set of labels predicted by H for example *xi* .

The following metrics are used for the evaluation of H on D:



The F1 score can be interpreted as a weighted average of the [precision and recall](https://en.wikipedia.org/wiki/Precision_and_recall), where an F1 score reaches its best value at 1 and worst at 0. The traditional F-measure or balanced F-score (**F1 score**) is the [harmonic mean](https://en.wikipedia.org/wiki/Harmonic_mean#Harmonic_mean_of_two_numbers) of precision and recall:

F1=2∗precision∗recall-precision+recall

The F1 metric weights recall and precision equally, and a good retrieval algorithm will maximize both precision and recall simultaneously. Thus, moderately good performance on both will be favored over extremely good performance on one and poor performance on the other.

# 7. Schedule

# 8. Conclusion

# 9. Reference

1) Yelp Restaurant Photo Classification in Kaggle:

<https://www.kaggle.com/c/yelp-restaurant-photo-classification>

2) Yelp Restaurant Photo Classification Data Files:

<https://www.kaggle.com/c/yelp-restaurant-photo-classification/data>

3) CaffeNet: Deep Learning Framework:

<http://caffe.berkeleyvision.org>

4) Multi-Label Classification:

<http://dml.cs.byu.edu/~cgc/docs/atdm/Readings/MLM-Overview.pdf>

5) F1 score:

<https://en.wikipedia.org/wiki/F1_score>