

Image Classification Using Convolution Neural Network

Edison LIM Jun Hao

Edisonjl@andrew.cmu.edu

Carnegie Mellon University

ABSTRACT

Image classification is tested solution to classifying big data into meaningful insights. Using data from Yelp, a popular food and service crowd-sourcing platform, I aim to investigate two approaches to classification. First, I will use a computer vision method - Histogram Oriented Gradient, to classify images. Using this computer vision method, I compared the results to that of a deep-learning approach. Using Convolution Neural Network with Support Vector Machine (SVM) multi-label classification, I obtained a score of 0.8001, which is significantly higher than computer vision approach of 0.7165.

INTRODUCTION

In this project, I am classifying the images of restaurants based on the images that other users have posted on Yelp. Given a set of trained data with labels, my objective is to correctly classify the test photos into their respective labels.

Background

Yelp's website, Yelp.com, is a crowd-sourced local business review and social networking site. Active users on the platform submits reviews on local businesses such as salons, local business services and eatery. Though the company is meant to be a local business review site, most users use the service for food reviews. On top of leaving reviews of the restaurant and a star rating (out of 5), users often upload photos of food and information on its pricing.

Yelp is available through yelp.com. Yelp is also available on iOS and Android.

Labels

Currently, Yelp has an option for users to add a label to the review to help with classification. These labels are useful for filtering and optimizing search results.

There are nine attributes of image classification. A business can have more than one picture tagged to it. Each business can have zero to nine labels.

- 1 good_for_lunch
- 2 good_for_dinner
- 3 takes_reservation
- 4 outdoor_seating
- 5 restaurant_is_expensive
- 6 has_alcohol
- 7 has_table_service
- 8 ambiance_is_classy
- 9 good_for_kids

Dataset

The dataset comprises of test and train datasets that is approximately takes approximately 18gb of storage. There are 234,842 train photos matched to 1996 businesses¹, and 237,842 test photos matched to 10,000 businesses. Both dataset has a comma-separated value (.csv) file that maps the photo ID to the business ID. The train dataset also includes the labels for each photo.

Evaluation

The method of evaluation to test the correctness of prediction is through the means of *Mean F score*. The F1 score measures accuracy using the statistics precision p and recall r . Precision is the ratio of true positives to all predicted positives.




$$F1 = 2 \frac{pr}{p+r} \text{ where } p = \frac{tp}{tp+fp}, r = \frac{tp}{tp+fn}$$

An observation of the *F1 score* reveals that it measures both the recall and precision in equal weightage. Thus, a good score must maximize both precision and recall simultaneous. Good performance on both is favored over extremely good performance on one but poor performance on another.

DATA EXPLORATION

The photos given is cleaned, and the Yelp team took special efforts to make the photos relevant to the challenge. However, the data science challenge would not be realistic if data is too clean.

Within the dataset, there are some photos that are considered “irrelevant”. These “irrelevant” photos do not directly translate into meaningful features for analysis. Below are some examples of irrelevant photos:

		
Photos of people Many users post pictures of themselves with the food.	Text photos Some users put photos of text such as menu.	Business Logos Store signage and logo is common on Yelp reviews.

Though some of these photos - like the menu and the prices - is very useful to a normal human user, it is a nightmare for machine learning. Since machines do not have the ability to decipher what is relevant and what is not, it will end up learning everything that we parse into the framework. Accuracy will hence be affected with the presence of these photos.

¹ 4 business IDs is dropped from the original 2000 due to incomplete data

EXPERIMENTS

I performed two experiments to explore how deep-learning can solve the image classification problem.

Feature extraction without Convolution Neural Networks

To extract the features of the image, I used the histogram of oriented gradients (HOG) method that was available through *scikit-image* python package².

Extracting features through HOG takes a few steps. First, we must resize it into a computational feasible format (256 x 256) and convert images to black and white representation. Then, scikit-image will compute the gradient image and normalize across the blocks. Finally, after collecting the HOG from all blocks, the image is flattened into a feature vector for use in classifier.

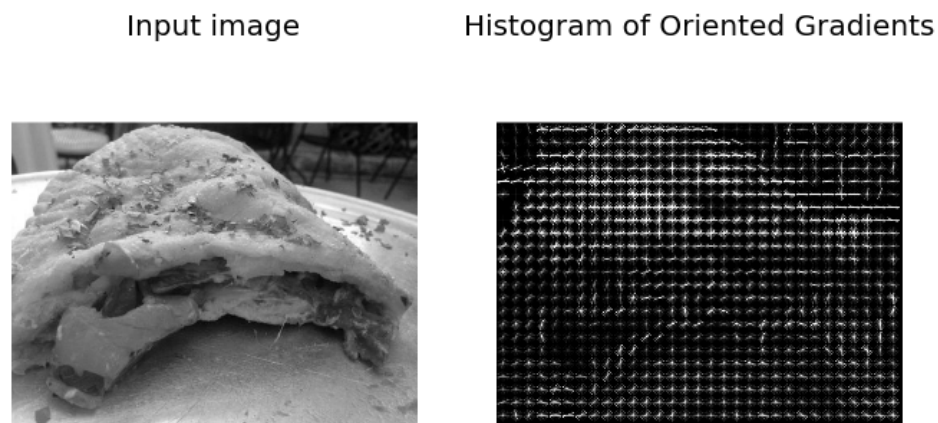


Figure 1: Applying HOG to a sample train image

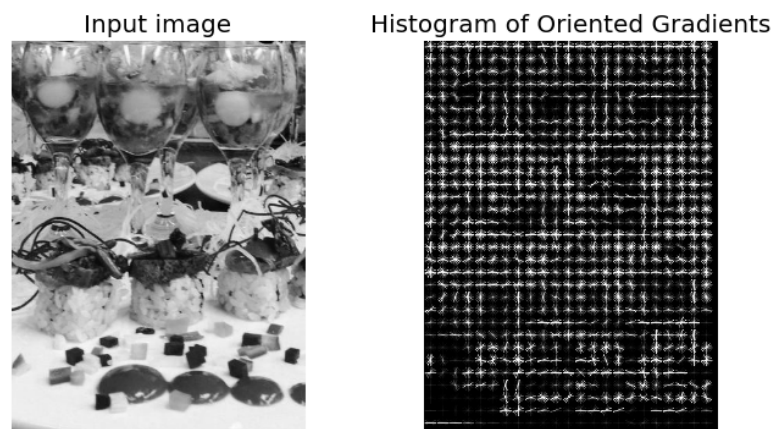


Figure 2: HOG on a bright image

Observations

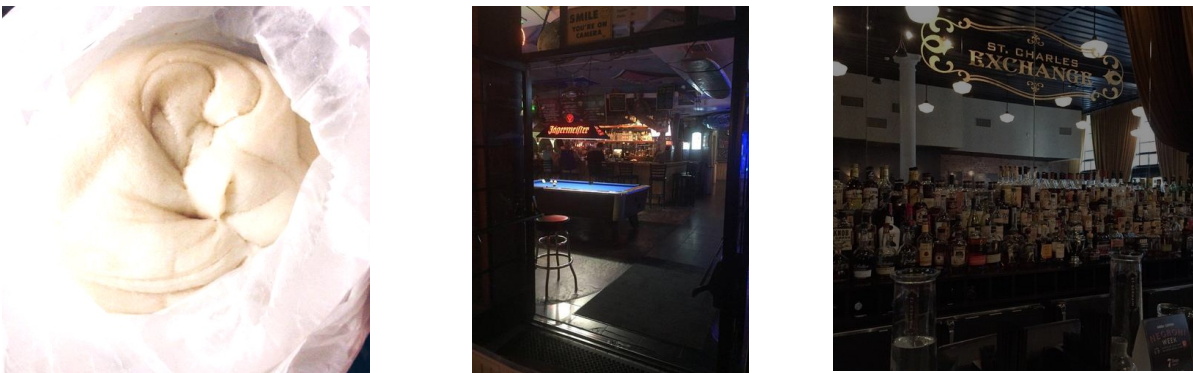
Since HOG extract vectors based on the values from a black-white rendition of image only, the effectiveness of the technique is limited when it comes to extremely bright and dark images. Also,

² HOG via Scikit-image: http://scikit-image.org/docs/dev/auto_examples/plot_hog.html

since my implementation does not account for transposition or angle-correction, the existence of duplicates will skew the results of the prediction.

For instance, if the photo of a certain food F is taken from different angles, the current implementation will consider them as different images. Although recognizing a visual concept such as whether two photos are of the same food is trivial to humans, it is an immense challenge for computers. The main challenges for computer vision problems are view point variation (orientation distortion), scale variation, deformation, illumination and occlusion.

To correct this characteristics of user-generated content, where we do not have the means to ensure a controlled image extraction, a computer vision algorithm must be implemented to correct these invariants. A good image classification model must be invariant to the cross product of all these variation, while simultaneously retaining sensitivity to the inter-class variations³.



Examples of images that are not effective with HOG method due to illumination

Feature Extraction with Convolution Neural Networks

One of the important thing to consider when using deep neural network is to decide if you wish to use pre-trained dataset or to train from scratch. Aside from being limited on resources like time and money, the dataset is also “small” and it cannot produce a good model. Indeed, it might be possible to train with ImageNet database, but the process is time-consuming and does not guarantee better results than Caffe. Thus, I decided not to train a network from scratch and download the pre-trained caffe network instead.

Caffe is a deep learning framework that is developed by Berkeley Vision and Learning Centre (BVLC)⁴ from UC Berkeley. I chose Caffe for its extensive support, documentation and suitability for the kind of image that I am classifying (i.e. photos of real objects).

In addition, since Caffe is implemented in *layers*, it is possible to adjust and fine-tune by layer and classify it using SVM. The fact that Caffe has python support also made it a compelling choice over other deep-learning frameworks.

³ Reference on computer vision challenges: <http://cs231n.github.io/classification/>

⁴ Caffe Deep-learning framework: <http://caffe.berkeleyvision.org/>

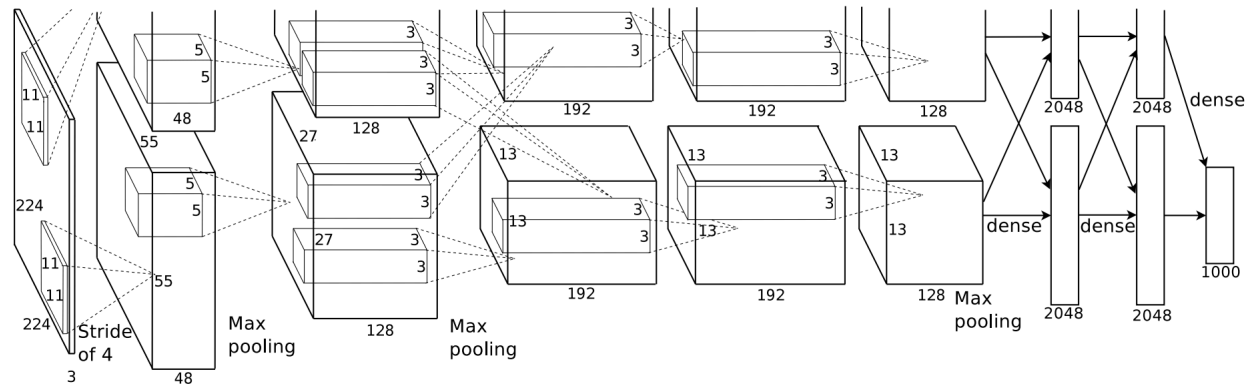


Figure 3: Architecture of ConvNet (Krizhevsky et al⁵)

The basic idea of using deep convolution neural network (Deep ConvNet) is that we feed an image on the left side. The depth of 3 refers to the channel of image, specifically RGB. Then the framework will process it through a series of convolution filter and generate a 1000-dimension on the output.

An observation about the ConvNet is that the last fully-connected layer are often overfitted to the ImageNet classes. Thus, the best features are obtained from the second layer from the last – penultimate layer. Using the features from this layer which is in the form of 4096 dimensions, I processed it through a SVM.

Classifier

Both my models are classified with SVM linear model. I used the `OneVsRestClassifier` from *scikit-learn*⁶, which fits one classifier per class. For each classifier, the class is fitted against all other classes. Each label is represented by only one classifier, hence my model builds 9 binary classifiers for the classification stage.

⁵ <http://papers.nips.cc/book/advances-in-neural-information-processing-systems-25-2012>

⁶ <http://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html>

RESULTS

F1 Score

The two approaches outlined above performed much better than randomly assigning labels. It is also observed that the transfer-learning using CNN performs much better than simple image analysis. As discussed, on top of transfer learning which helps the model identify objects,

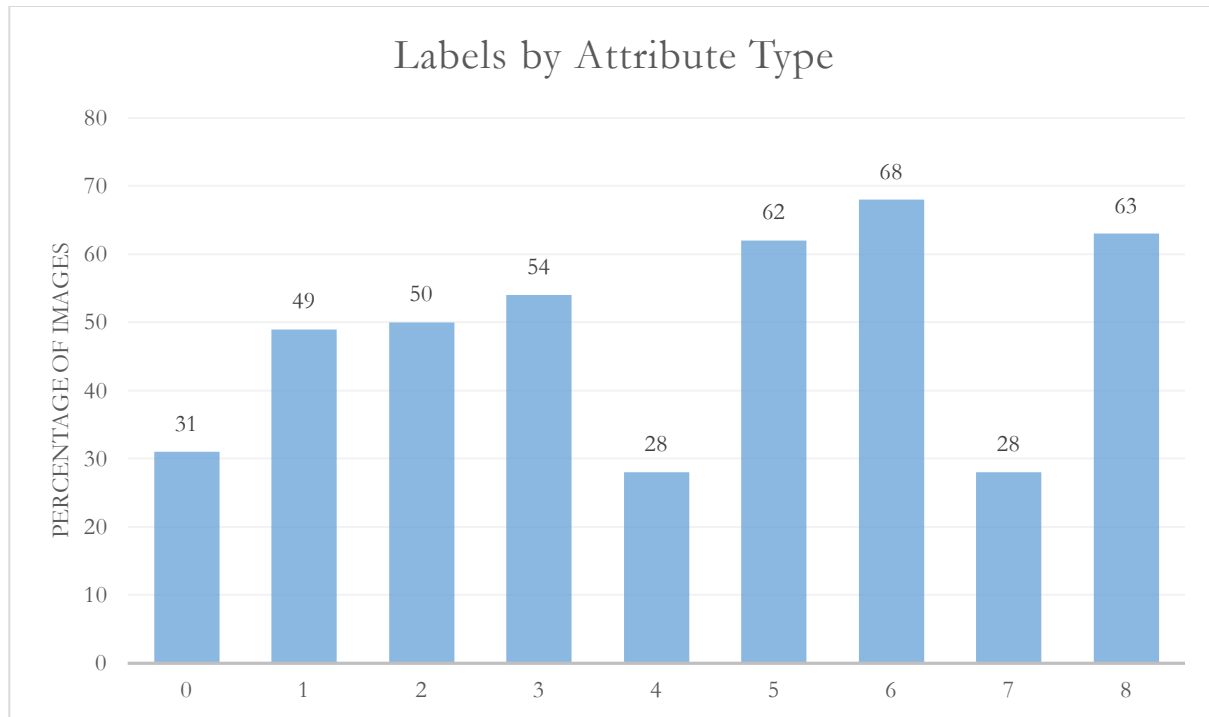
Method	F1-Score
Randomly Assigning Labels	0.4347
Color Distribution Distribution	0.6459
Histogram of Oriented Gradients	0.7165
Deep-learning with CNN	0.8001
Top 10 Submissions on Kaggle	0.81 to 0.84
Yelp's production pipeline	0.94

Photo Output

IDs	Photos	Labels
<u>00er5</u>		<p>1 good_for_lunch</p> <p>5 restaurant_is_expensive</p> <p>6 has_alcohol</p>
<u>98317</u>		<p>2 good_for_dinner</p> <p>3 takes_reservation</p> <p>4 outdoor_seating</p> <p>5 restaurant_is_expensive</p> <p>6 has_alcohol</p> <p>7 has_table_service</p> <p>8 ambiance_is_classy</p>

Distribution of Labels

The distribution of labels on the test data is as follows:



Attribute 6 (has_table_service) appears to be the most common label among the test photos. Attribute 4 and 7 (restaurant_is_expensive and ambiance_is_classy) appears to be the least common labels.

FUTURE WORKS

This project is meant to be an exploratory study on using deep-learning to classify photos. By implementing the pre-trained network on my data, I have only scraped the tip of the iceberg.

To further improve the accuracy of prediction, more work needs to be done.

- Exploring other deep-learning frameworks
 - Different deep-learning framework (e.g. mxnet inception network) might yield different results
- Fine-tuning weights to obtain better results
 - Fine-tune the weights of a pre-trained network by continue backpropagation because the later layers of the Convolution Net become progressively more specific (but might not contain results that we want).
- Use of different classifiers to obtain weighted averages instead of using only one classifier
 - Due to overfitting concerns

WHAT DID NOT WORK?

I spent a large part of the paper explaining what worked, but I felt equally important to document what did not work so that people of the same level as me (beginner) can avoid.

I started off the project believing that I need to remove duplicates from the database. It is a computationally intensive work since the model must still iterate through all the photos and identify which are the duplicates. This did not work out well due to the technical complexities, and also it is hard to justify what is considered a duplicate. If a user takes photos from different angles, is it considered a duplicate?

I tried to use the kNN method for multi-label classification but it did not yield good results.

I still believe that stacking is not an ideal solution for a small dataset as it might over fit the data.

USE-CASE OF IMAGE CLASSIFICATION IN REAL LIFE

The above-mentioned strategy produced a score of 0.800, which is a fair accurate model. The prediction score of the model can be further increased by the strategies outlined under the Future Works section. With this accuracy, the following use-case can be made possible for Yelp:

- 1 Assigning attributes automatically for those images that are not tagged by other users
- 2 Perform better filtering results and increases relevance of reviews for users
- 3 Help business owners identify its current reputation and business value from defined business attributes
 - a. E.g. “Is my restaurant considered upscale?”

ACKNOWLEDGEMENTS

I would like to extend my gratitude to these organizations or people for contributing to this project in one way or another:

- Yelp Engineering Blog
- Stanford CS231n (<http://cs231n.github.io/>)
- Kaggle Forums
- Amazon EC2