Yelp Restaurant Photo Classification

Abhinav Jain Aditi Nayak Shreyas Bhatia

111495982 111482252 111432576

abjain, adnayak, shrbhatia@cs.stonybrook.edu

***Abstract*— The project aims to build a model that automatically tags restaurants with multiple labels using a dataset of user-submitted photos.The problem presented is that the user has to select labels while uploading the images. This may give rise to inconsistencies such as leaving partial or unlabeled images.This hampers the Yelp's ability to recommend the most helpful and reliable reviews for the Yelp customers.We discuss a few technical approaches along with challenges which we faced and provide an analysis of the different models by reporting the mean F1-scores.**

I. INTRODUCTION

The Yelp site apart from providing restaurant reviews help to review the businesses.The photos posted by the user provide a overview about the restaurant about which the reviews are written. Also the images are indicative about the look, feel and amenities available at the restaurants. For example, a photo of bar area implies that restaurant serves alcohol and may not be suitable for types of users like small children. Currently the Yelp platform requires the user to manually fill the labels for the photos which is optional. In this project, we are tackling the problem of automatic labeling of images uploaded by the users depending on the features mapped to restaurants. This project will further benefit Yelp by being able to give better label suggestions to the users and categorize them accordingly.

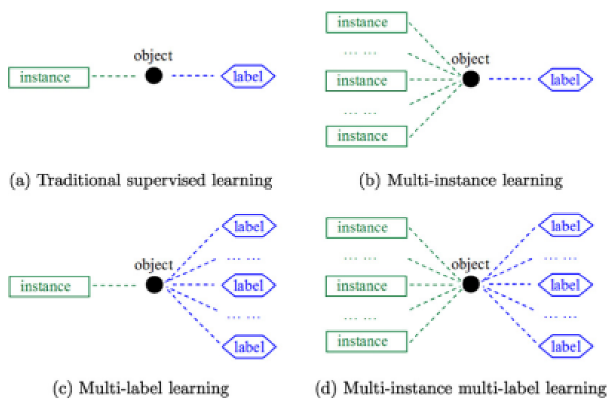

instance
object
label
(a) Traditional supervised learning
(b) Multi-instance learning
(c) Multi-label learning
(d) Multi-instance multi-label learning

Fig. 1: Types of Learning Models[3]

The problem requires us to generate multiple labels for a restaurant given multiple images. A trivial machine learning problem would be either regression based or binary or multi-class classification based. This problem requires us to generate multiple labels i.e. to predict multiple classes for a single instance. Furthermore, there are multiple instances as well (restaurants) and we have to predict the classes that are there for all the restaurants. This makes the problem harder as we now have to address multiple instances. Such types of problem are Multiple Instances and Multiple Labeling learning problems (MIML). We will discuss how to solve such problems below.

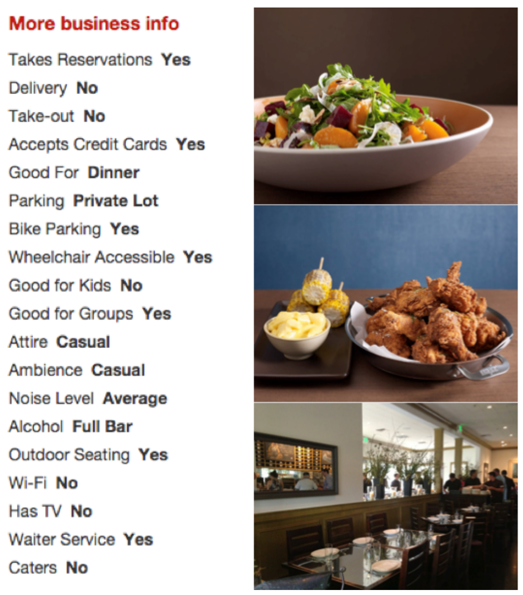
II. RELATED WORK

The Winner[1] of the Kaggle competition implemented a lot of concepts apart from pre-trained Convolutional Neural Networks(CNN). He used concepts of Fisher Vectors,VLAD descriptor and applied PCA.To deal with multilabel instance of the problem he used multi-output neural network.He also applied PCA on the penultimate layer of Full ImageNet Inception-BN.

The Runner Up[2] contestant of the Kaggle competition used pre-trained convolutional networks to extract image features and used resnet-152 provided by Facebook.To handle multilabel instance of the problem he used multilayer perceptron since it gave the flexibility to handle both the multiple label and multiple instance at the same time.

III. DATA

The dataset used for this project is provided by Yelp on Kaggle. The full dataset is comprises of approximately 234, 000 untagged RGB images corresponding to 2,000 restaurants.The labels which are to be predicted are:
0: good_for_lunch
1: good_for_dinner
2: takes_reservations
3: outdoor_seating
4: restaurant_is_expensive
5: has_alcohol
6: has_table_service
7: ambience_is_classy
8: good_for_kids
There is also train_photo_to_biz_ids.csv and test_photo_to_biz.csv which maps the business ids to photo ids. Also there is train.csv which maps the business ids to its corresponding labels. Our challenge is to provide prediction for each business id as follows :



More business info

Takes Reservations	**Yes**
Delivery	**No**
Take-out	**No**
Accepts Credit Cards	**Yes**
Good For	**Dinner**
Parking	**Private Lot**
Bike Parking	**Yes**
Wheelchair Accessible	**Yes**
Good for Kids	**No**
Good for Groups	**Yes**
Attire	**Casual**
Ambience	**Casual**
Noise Level	**Average**
Alcohol	**Full Bar**
Outdoor Seating	**Yes**
Wi-Fi	**No**
Has TV	**No**
Waiter Service	**Yes**
Caters	**No**

Fig. 2: Labels values for a business id[4]

IV. PROCEDURE

The Procedure of this project involves 3 main steps:

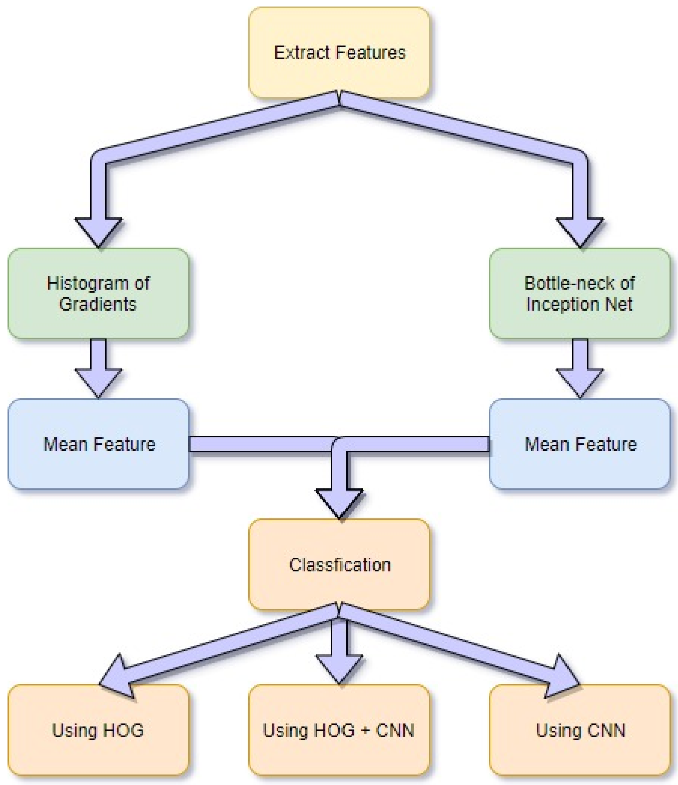


Fig. 3: Work Flow Diagram of Project

A. *Step 1:Feature Extraction*

Feature Extraction involves representing the image in such a way that the it extracts the useful information and discards the non important information.The technique used for extraction of features converts an image of width x height x 3 channels size into a feature vector of size n.This is one of the most important steps in the entire work flow. In this step we are extracting the features using following 2 approaches:

*1) **HOG**:* In the HOG feature descriptor, the distribution (histograms) of directions of gradients (oriented gradients) are used as features. Gradients (x and y derivatives) of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and we know that edges and corners pack in a lot more information about object shape than flat regions.The images in the data set are of varying sizes. Initially we resize the images to a uniform shape of (256X256) and convert it to gray scale. By doing this, it is easier to extract the feature vector from the images using HOG. We extract the image feature vector of 2048 size by passing parameters to HOG such as $orientations = 8, pixelspercell = (16, 16), cellsperblock = (1, 1)$.Then we create a .csv file of around 10GB where we map the image id to its feature vector. This took around 8 to 10 hours to extract and was the most time consuming task of our problem.

*2) **Bottle neck features of a CNN**:* A convolutional neural network has two parts

- Part 1: A chain of Convolutional Layers
- Part 2: A network of fully-connected

The convolutional layers are used for feature learning. These features are the input for the fully connected layer. We will use these features (The output of the convolutional layer) as our feature vector. In this project, we have used InceptionNet V3 as our pre-trained CNN. The output of the convolutional layer are taken in as our feature vector of size 2048. Initially, we were extracting images one-by-one (get_feature() function). This turned out to be extremely slow (taking around 8 hours for 15,000 rows) because of repeated call to TensorFlow.session(). To optimize this, we processed the images in batches. We first read an image and resize it accordingly.We then created batches of images and passed them into the network. We then merged the outputs to create our final feature vector file. Even after this optimization, we still had to spend around 10 hours on the training data to extract the images. This file was also roughly 10gb.

B. *Step 2:Business Feature Extraction*

We tried to convert our problem to a machine learning labeling problem. For this, we can take either median or mean for the image feature vectors extracted in the previous step. The mean feature vector works better in our case .For the taking the mean vector of each business id We add the features of images mapped to the business id column wise and then average them out by dividing the column wise summation of images feature by the total numbers of images mapped to a given business id.This gives us a a file of mean business features mapped to a business id. The main and most difficult part of this step was to handle the large input file of feature vectors from previous step.Since the file from the previous step was huge we decided to divide the file in chunks,process it and write the output in chunks to the file.

We did this step for the two files generated in our Feature Extraction Step giving us two files, one for HOG and one for CNN.

C. Step 3: Multi label Classification

In this step we train our model to generate labels for our restaurant using the feature generated in the previous step. The end result will be labels predicted for each restaurant. We tried three different approaches in this step to create our feature vectors for labels:

- a) Using HOG features: We directly used the feature vector generated after taking the mean features per business.
- b) Using bottle-neck features from our Inception Net: We directly used the mean feature vector generated in the previous step.
- c) Using a combination of HOG and bottleneck features: We stacked the mean feature vector generated from HOG and bottle-neck feature generated in the previous step. After this, we had a feature vector of size 4096 (2048 HOG + 2048 CNN). We applied Principal component analysis (PCA) on this vector to reduce the dimensions.

After generating the features, we need to deal with the labeling aspect of the problem. One way, is to convert it into a multi-class problem. We will convert each label into a set of binary labels. Then, we can use the one vs all strategy to classify the multi-class problem. The base classifier we have used for the strategy is a Support Vector Machine. We also tried different kernels for it like linear, rfg and sigmoid.

We are randomly splitting our train data set into 80:20 as training data and training-test data. Using this we are calculating our measure of test accuracy. We cannot calculate this for the given test data because labels for the test are not given.

V. RESULTS

We are using F1-score to measure our test accuracy. Due to the nature of our problem, we cannot use this directly. Rather, we are using mean F1 score. We define mean F1 score as the mean of all the F1 score for all our labels.

Yelp prepared several baseline models to facilitate benchmarking and comparisons. [4]

- The first model is just a naive random guesser: it makes a random assignment for each attribute with equal probability. Random guessing results in a score of 0.4347.
- The second model plays with color: it calculates the color distribution of all the images of a test business, compares that to the average color distribution of businesses with positive attribute values and negative attribute values respectively, and assigns the value with a more similar color distribution to the test business. This is harder to beat - our color feature benchmark results in a score of 0.6459.

Out of all the techniques, the combined feature vector of Histogram of Gradient and Inception Net bottleneck performed the best with a mean F1 score of 0.77 on our train-test data.

The linear kernel of SVM out-performed the rest. With a rbf kernel, we were getting a score of 0.73 and for a sigmoid kernel, we got 0.71 when used with the combined features.

A. Tables

Sr No.	Method	F1 Score
1.	HOG + OneVsRest Classifier	0.71
2.	CNN + OneVsRest	0.74
3.	HOG + CNN + PCA + OneVsRest	0.77

TABLE I: Results

B. Graphs

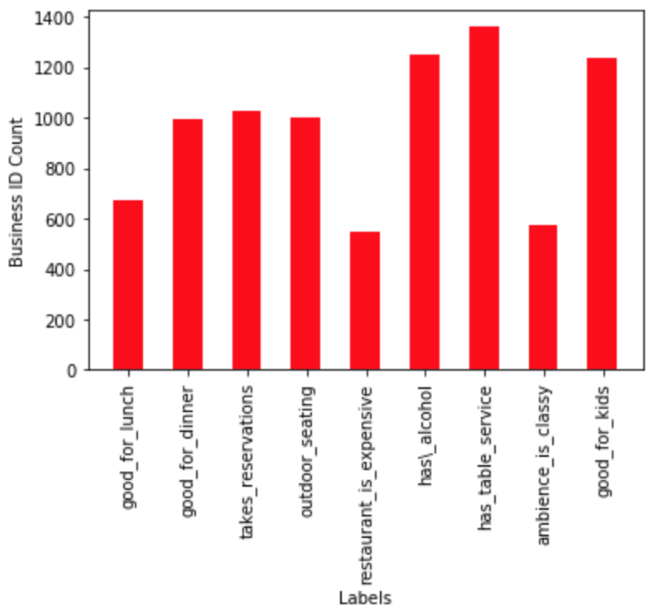


Fig. 4: Labels vs Business id's of Train Images

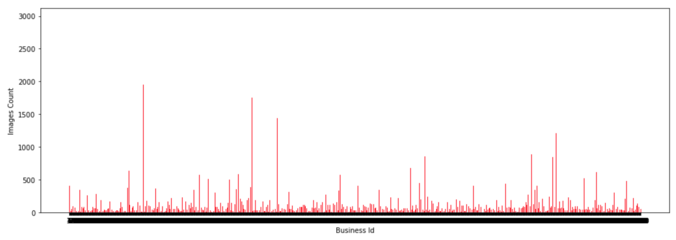


Fig. 5: No of Images per Business id of Train images

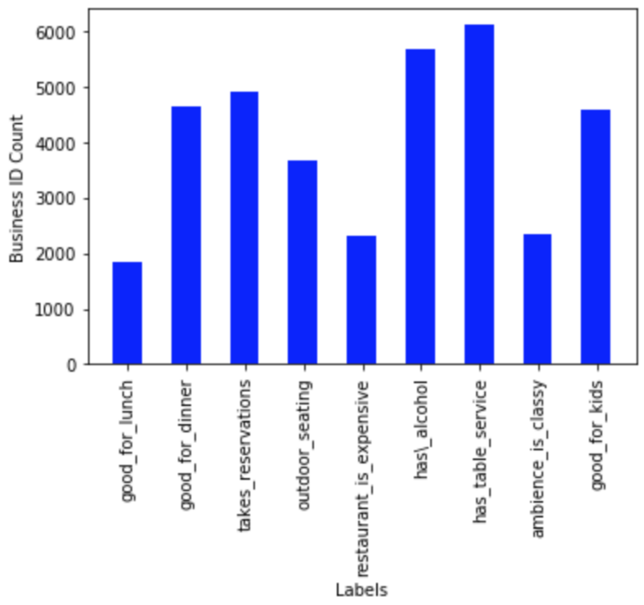


Fig. 6: Labels vs Business id's of Test Images

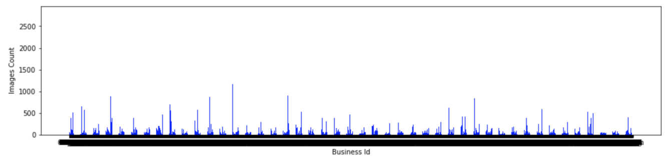


Fig. 7: No of Images per Business id of Test images

VI. CONCLUSION

Through this project we discovered the various aspects of MIML type of classification. We presented a few different solutions to the problem and have beat the baselines set by Yelp.The future scope of the project includes using multi-layer perceptron or XGBoost to increase the F1-score. We can also try to implement the algorithm given in [8].

REFERENCES

[1] https://engineeringblog.yelp.com/2016/04/yelp-kaggle-photo-challenge-interview-1.html
[2] http://blog.kaggle.com/2016/05/04/yelp-restaurant-photo-classification-winners-interview-2rd-place-thuyen-ngo/
[3] http://www.sciencedirect.com/science/article/pii/S0004370211001123
[4] https://engineeringblog.yelp.com/2015/12/yelp-restaurant-photo-classification-kaggle.html
[5] https://arxiv.org/abs/0808.3231
[6] $http : //mlwiki.org/index.php/One - vs - All_C lassification$
[7] https://mlr-org.github.io/mlr-tutorial/devel/html/multilabel/index.html
[8] http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=5693992