Yelp Restaurant Photo Classification

Haomin Peng
haomin@stanford.edu

Yiqing Ding
yiqingd@stanford.edu

Boning Huang
bnhuang@stanford.edu

1. Introduction

Everyday, millions of users upload numerous photos of different businesses to Yelp. Though most of them are tagged with simple information such as business names, the deep-level information typically are not revealed until looked by humans. Such deep-level information contains things like whether the restaurant is good for lunch or dinner, whether it has outdoor seating, etc. This information can be crucial in helping potential customers to determine whether its the ideal restaurant they want to dine in. Hand labeling these restaurants can be not only costly but also tedious since photos are uploaded almost every second to Yelp.

In this project, we used machine learning techniques, specifically image classification methods, to identify the context and rich information of different photos on Yelp. We used three different models to train and test the given dataset.

The dataset is provided by Yelp on Kaggle.com. In the training set, 234842 arbitrary-sized images submitted by Yelp users are correlated to one of 2000 Yelp businesses. The number of corresponding photos for each business ranges from 2 to 2974, with mean of 117. The image count distribution is shown by Fig1. Every business is tagged with multiple descriptive labels in 9 aspects. The labels includes:

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

As an example, Fig2 illustrates the structure of the dataset. Our goal is to predict the labels of a business in these 9 aspects given its photos. A notable challenge is that the labels are not directly correlated to each image. Individual photos can possibly be indicative in one aspect, yet it is tagged with all the labels of the business. As a result, the dataset might exhibit high noise level. The fraction of each label across the training set is shown in Fig3. Though assumed independent, some label pairs exhibit high covariance, such as "has_alcohol" and "has_table_service" (Fig4).

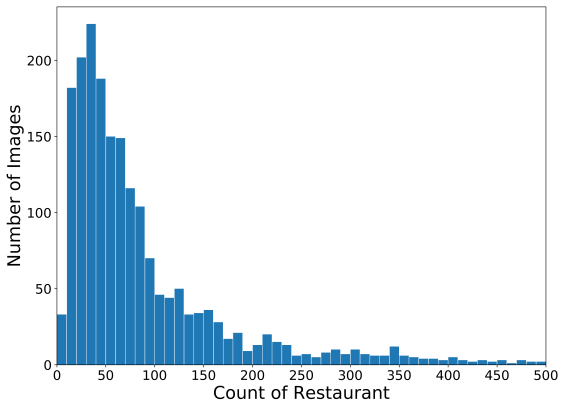

Figure 1. Image count distribution of the dataset.

Figure 2. Sample images corresponding to two businesses in the dataset. (a) A business with labels 1, 2, 5 and 6; (b) A business with labels 2, 3, 4, 5, 6 and 7.

2. Related Work

The main problem we are tackling in this project is essentially an image classification problem. Because of the three-dimensional nature (width, height and color) of images, traditional neural networks usually cannot scale well to the entire image and have the problem of overfitting when learning parameters of images. To tackle this issue, Convo-

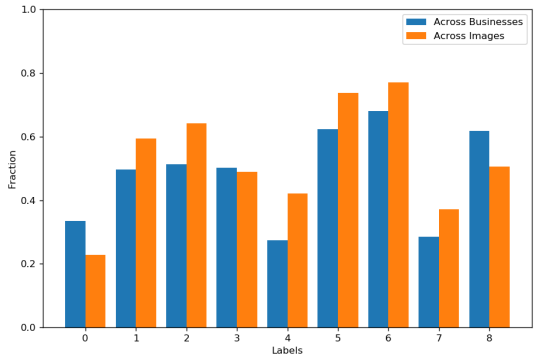

Figure 3. Fraction of each label across all businesses and images in the dataset.

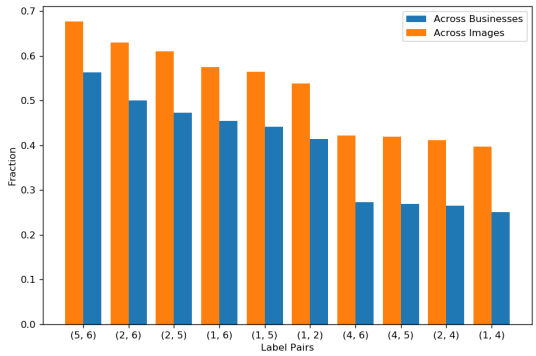

Figure 4. 10 most frequently-appeared label pairs in the dateset.

lutional Neural Network (CNN) was introduced. CNN differs from traditional neural network by using a 3D volume of neurons and neurons in each layer will only be connected to a small region of the layer before it. This allows CNN to process image data much faster than traditional neural network while avoiding the problem of overfitting.

Assessed with the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012, AlexNet of CNN was able to achieve a top-5 error of 15.3% [5]. Later on, the evolution of CNN was evident in ILSVRC with GoogLeNet[6] achieving a top-5 error of 6.7% in 2014 and ResNet[7] achieving a top-5 error of 3.6% in 2015. It is apparent that CNN has achieved an impressive result in learning images, therefore we widely utilized the ResNet in this project to classify photos of different restaurants.

ResNet is an variant of CNN with residual neural networks to tackle the issue of vanishing gradient in neural networks. Rather than the usual nonlinear activation function $H(x)$ used in traditional neural network, ResNet uses a nonlinear function defined as $F(x) := H(x) - x$. At the output of the stacked weight layer, x is added to $F(x)$ so

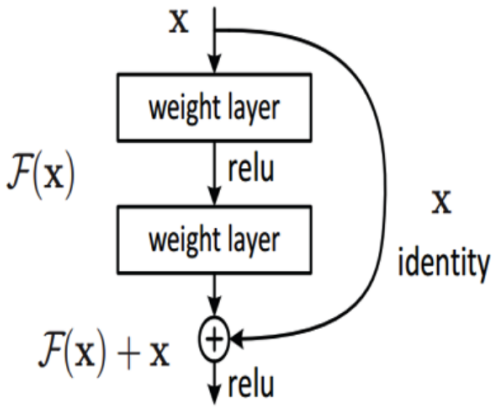

Figure 5. Residual Learning in ResNet [7]

that the original mapping is recast and passed through the Rectified Linear Unit (ReLU). This enables us to transfer important information from the previous layer to the next layers.

3. Evaluation Metric

Since this is a multilabel classification task, we use mean F1 score as evaluation metric to evaluate performance of our model on our train_dev set and the test set provided by kaggle (which is also the official evaluation metric for this competition) . F1 score measures accuracy based on precision p and recall r, $p = \frac{tp}{tp+fp}$ where tp is true positive and fp is false positive, $r = \frac{tp}{tp+fn}$ where fn is false negative.

$$F1 = \frac{2pr}{p+r}$$

4. Method&Experiment

4.1. High Level Pipeline

Considering the nature of this challenge is a combination of image processing and multi-instance-multi-label classification on processed features, our instinct is firstly try out several feature extraction method and then elaborate on the classification task. Another method is to first assign the labels of a restaurant to all its photos (remember that the labels are given for restaurants in training set, not for photos) and then do transfer learning on these photos with CNN. For the evaluation, we could compute the F1 scores on photo level, we could also get the mean or other measurement of the outcome of all photos belong to a restaurant as the result probabilities for this restaurant and evaluate on restaurant level.

4.2. Model 1:PCA + classification

The sizes of images in training set varies a lot, to unify the formats for following analysis, we resized all the pictures to 224*224 and transformed them into numpy arrays of size$(224, 224, 3)$.

For a naive baseline, we created a dataset with photos directly tagged with labels and evenly partitioned all the photos into 2 parts with 80% for training and validation and 20% for test. This step of treating each photo independently will lose information of how certain pictures are grouped into a same business and share the same labels, this information, however,is valuable for prediction, because photos of a same business may have similar background features, and the meaning of 9 labels are all concerned with not only the food itself but also the environment and vibe of restaurants. Therefore for the next steps we would try to treat photos for each business separately to take maximum use of the group features. For the baseline, we simply removed the background information of each restaurant, we normalized photos by calculating the mean image for each restaurant and subtracted from each image the mean image of the restaurant it belongs to.

For the second step of pipeline, we applied Principal Component Analysis (PCA) on preprocessed images. PCA is known to be used for dimensionality reduction as well as used decorrelating the data. Clearly, pixel values are highly correlated in raw pictures. The idea is that if we can decrease the number of highly correlated pixels, the small objects can contribute more to the decision. We perform the incremental PCA algorithm to the resized image in training set by a batch size of 10000 since the number of samples is big. We then chose the first 600 principal components and applied the same transform to the test set to make predictions. The number of principal components is determined by maintaining the 95% of variance of the original data, As is shown in fig6.

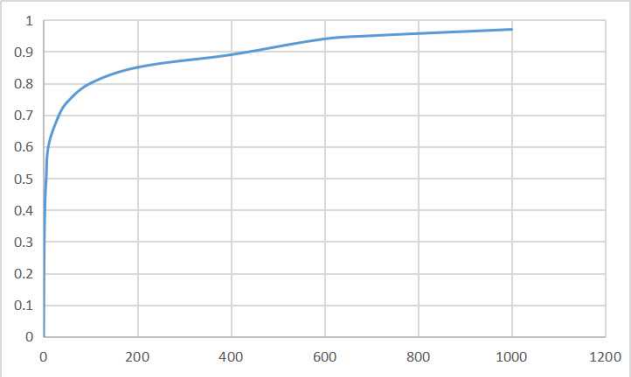

Figure 6. Illustration of top features analysis, x axis represents number of top features and y axis represents ratio of variance these features account for

After performing PCA, our problem has transformed into a multilabel classification task, an instinctive approach is to assume that each label is independent and use one vs rest (OvR) strategy on 9 binary classifiers with each classifier predicts the probability of a single label. For each binary classifier, we simply employed support vector machine (SVM) and generated a likelihood for each class/label where whenever the likelihood is greater than 0.5, we tag the sample with the label. For each sample picture, all 9 classifiers are needed to be executed to identify the potential multiple labels.

For the fine tune of SVM, we performed 5-fold cross validation(CV) on training data (80% of all data) to test the performance of several choice of parameters, which are $Kernel \in \{linear, rbf\}$ and $gamma \in \{0.1, 0.01, 0.001\}$, the best result comes with kernel = rbf and gamma = 0.01.

To compare the results of svm with other common classifiers, we performed ensemble of Logistic Regression and Random Forest with the same OvR strategy. For the parameters of Random Forest, we used default coefficients provided by sklearn toolkit and set the minimum number of samples required to be at a leaf node to be from $\{10, 20, 50, 100, 200, 500, 1000\}$, the best result is for 20.

Table 1. Results of model 1

Model	Train_val F1	Test F1
PCA+Logistic	0.647	0.639
PCA+RandomForest	0.635	0.622
PCA+SVM	0.644	0.629

By applying the models mentioned above on test set(the 20% data), we got F1 scores for both train-validation set and test set as in table1.The scores for train_val set and test set does not show much gaps, which proves the robustness of models and the evenness of data. It is interesting that svm didn't not yield a significant better result than simple random forest and logistic regression, though it is tens of times of time consuming than other two paradigms.

Yelp also provided two baselines. The first is a random guess which makes a random assignment for each attribute with equal probability, we implemented this on our test set and got a score of 0.414. The second one computes the color distribution of a image and compare it with a mean distribution of each business and assigned it the labels of the closest business, this method reports a score of 0.610 and 0.619 on our test set, our baseline have improved the performance compared to these benchmarks.

4.3. Model 2:Transfer Learning

As mentioned in introduction, CNN is very powerful in computer vision task. We want to take advantage of CNN, however, training from scratch would be time-consuming,

and may not converge to desired result due to bad choice of parameters, therefore we did transfer learning on pretrained Resnet model.

As can be seen from fig7 , we replaced the last 1000 class fully connect layer of original resnet with a 9 class fully connect layer followed by a sigmoid layer, and define the loss as cross entropy of the 9 dimension logistic output and the label. We implemented the network in Tensorflow and got the pretrained weights for resnet from the Tensorflow model zoo.

Since ResNet-101 has much more parameters to train, we first trained on ResNet-50. For the choice of paramters, learning rate for transfer learning is typically much smaller since we are fine-tuning the model, So we use 10-4 as base learning rate for the whole network, such that we can update all parameters a bit, and wont affect too much to the pretrained parameters. We chose Adam optimizer with momentum 0.9 and 0.999, and feed into network batch size of 64.

The training and validation loss can be seen in fig9, it is obvious that ResNet-50 shows an overfitting after several thousands of iterations, since the loss for validation set barely decrease, and the difference of loss between training set and validation set becomes bigger, so we freeze the updating at 4000 iterations and did all the following steps on this 4000 iteration model.

We first evaluated the model on each photos as shown in table2. To evaluate performance on restaurant, we used the average output of all the photos of a restaurant as the probabilities for that restaurant, we assigned a label to be 1 when the probabilities for that label is bigger than 0.5. This gives us a result as shown in table2.

We repeat the above steps for ResNet-100, however, we could see from fig8 that for the first 3000 iterations the ResNet-101 shows a higher loss on validation set and converges much more slowly and viberate more.Therefore we think ResNet-50 is more suitable for this task.

Table 2. Results of ResNet models

Model	Train_val F1	Test F1
ResNet50-photo level	0.762	0.738
ResNet50-restaurant level	0.721	0.704

4.4. Model 3: Feature extraction with Resnet

Despite of being extremely powerful on Imagenet, resnet did not get a desired result here, and moreover, theres a big gap between photo level result and restaurant level result, we think part of this comes from the internal flaw of model 2. Recall that the first step for model 2 is to assign the label of a restaurant to all its photos ,however, it should be noted

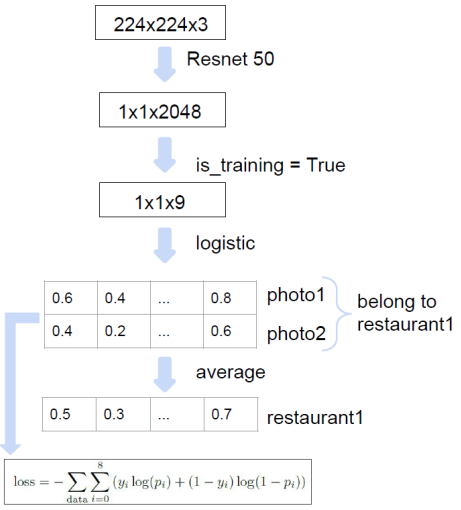

Figure 7. Illustration of top pipeline for Model 2

Figure 8. comparision between ResNet50 and ResNet101

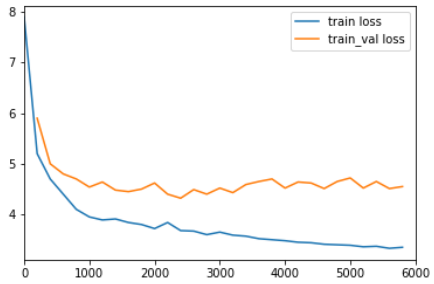

Figure 9. Learning curve of ResNet-50

that in training set not all the photos of a restaurant could contribute to all the labels of this restaurant. for example, for a restaurant with label alcohol , there is only one photo which shows a glass of beer, and the rest photos without any sigh of alcohol, if we simply propagate the labels alcohol to all of this restaurants photos, we are forcing our network to learn features that dont even exist. This is against the original intention of CNN and there is no wonder such would not work well.

So we came to the strategy of only using pretrained CNN as feature extractor without training and then for each restaurant,group the features of all the photo belongs to this restaurant then directly train on restaurants as show in fig10.

Now we are considering how to combine the 2048 features for each photo in a restaurant to generate feature for this restaurant. Simply Mean and std is not a good representative of how this feature behaves among all the photos of a restaurant, therefore for each restaurant, we compute mean, std, and the percentile of 60th, 80th, 90th, 95th, 100th for each of the 2048 features among all the photos belong to this restaurant,and this gives us 2048*7 features for each of restaurant. When inspecting all this features, we discovered that not all of them, some are almost all zeros across all , these features certainly could not improve the performance of classifier, therefore we further remove features with low std and have 10341 features left for each restaurant.

For the building of classifier for restaurant, we simply repeat the binary classifier plus OvR strategy proposed in model 1, this generate results in3.

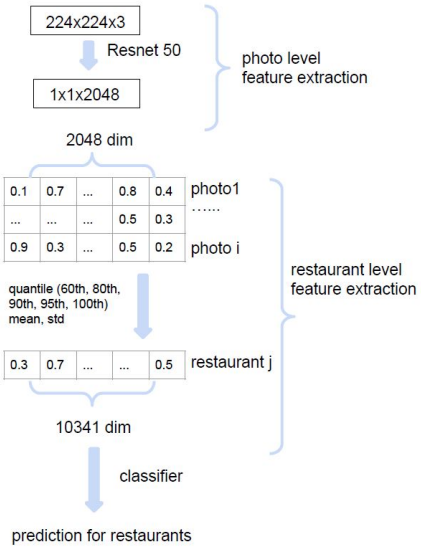

Figure 10. Illustration of pipelines for Model 3

Table 3. Results of Model 3

Model	Train_val F1	Test F1
feature+boost	0.829	0.802

5. Conclusion& Future Work

It is interesting that feature extraction with ResNet has much better result than directly training on ResNet for this task, the reason for this, as mentioned in part of model 3, could be that the activity of simply assigning the labels of a restaurant to all its restaurant bring two much wrong information to ResNet.

We mainly have two plans for future:

- Add more features from the previous layer of ResNet not just the last pooling layer
- If we want to get better result when directly trainning the network, we should design more clever loss rather than simply do sigmoid.

6. contribution

All team memeber contribute equally to this project.

References

[1] Pearson, K. (1901). LIII.On lines and planes of closest fit to systems of points in space. Philosophical Magazine Series 6, 2(11), 559-572. doi:10.1080/14786440109462720

[2] Gorban, A. N., & Zinovyev, A. Y. (2009). Principal graphs and manifolds. In E. S. Olivas et al. (Eds.), Handbook of research on machine learning applications and trends: algorithms, methods, and techniques (pp. 28 59).

[3] Rifkin, R. (2008, February 25). Multiclass Classification. Retrieved November 20, 2017, from http://www.mit.edu/ 9.520/spring09/Classes/multiclass.pdf

[4] Li, F. (2017, January 5). Convolutional Neural Networks (CNNs / ConvNets). Retrieved November 21, 2017, from http://cs231n.github.io/convolutional-networks/

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12), F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 1. Curran Associates Inc., USA, 1097-1105

[6] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).