DSCI6005 Final Project

Kaggle Competition: Yelp Images 7/22/17

**Data Scientist**: Chase Renick

**Introduction**:

In 2016, Yelp hosted a Kaggle competition involved classifying random food images uploaded to Yelp’s server. Participants were asked to develop a model that would accurately predict whether or not a given image could be classified as: was good for lunch, was good for dinner, took reservations, had outdoor seating, looked expensive, had Alcohol, was good for kids, had table service, had a classy ambiance. This multi-class classification problem garnered many competitors, especially given the prize: a ‘fast track’ hiring process at Yelp to their machine learning team. The following paragraphs will outline my approach towards solving their image classification problem using multiple machine learning techniques.

**Data**:

The data related to this challenge consisted of four components: the training photos, the test photos, the business id and associated photo ids, and the labels associated with a given business id.

**Challenges**:

There were many challenges associated with developing a robust image classifier ranging from preprocessing to model preparation to how labels were originally selected. Having worked with these files for over a month, it is clear that one of the major challenges from the competition was to create a mapping of photos, to business ids, to labels. This ‘architecture’ was an essential prerequisite to formatting the data in a way that it could be fed into a baseline model as well as a neural net. Among the many preprocessing challenges, some of the issues included:

1. A non-direct mapping of training photos, to business ids, to associated labels
   1. Resolved by using pandas, and carefully mapping image to label
2. Labels given as a string as opposed to ‘one hot encoded’ or organized in a Dataframe
   1. Resolved by splitting up strings, organizing it, and constructing a one hot encoded Dataframe
3. Duplicate photos
   1. Resolved by identifying duplicates and eliminating them
4. Non homogenous photo dimensions
   1. Resolved using Keras preprocessing and resizing dimensions to (281,281)

Other issues related to model preparation included but were not limited to:

1. Class Imbalance
   1. Resolved by calculating class weights as a dictionary and feeding these values into Keras’s fit function
2. Limited training data to extract relevant features to predict 9 classes
   1. Not resolved due to constraints relating to limited images provided by Yelp
3. Necessity for regularization
   1. Resolved by incorporating L2 regularization into Keras models

There were other challenges beyond preprocessing and model preparation that will be discussed in later sections. These challenges relate to how Yelp originally chose to label their photos, and whether or not features existed in the photos to even predict on such labels.

**Methodology**:

The basic methodology towards solving this multi-class classification problem was to compare various models and select the highest performing one among the group. However, as it will be demonstrated in later paragraphs, there are advantages and disadvantages toward each approach. These strengths and weaknesses will be discussed in the section called model comparison. The four approaches towards predicting the nine classes involved using a KNN classifier predicting individual classes, Neural Net predicting individual classes, Convolutional Neural Net predicting multiple classes (2 layers), and Convolutional Neural Net predicting multiple classes (Mock VGG Net).

**Baseline** **Model**:

First, establishing a baseline model would prove useful when comparing later iterations or improvements to that model using various techniques. Even though my labels were one hot encoded, I chose KNN as my baseline model. This has to do with the fact that measuring distances among pixel values in order to predict whether or not an image was in a certain neighborhood makes sense, but also because KNN has been proven to be a successful classifier for images in previous research. Once the KNN predictions were generated for each class, I used their associated F1 scores as benchmarks for future models. F1 score was chosen as the accuracy metric because, as it will be illustrated in later paragraphs, my preferred model predicted class probabilities for each of the nine classes, and there were many false positive and false negatives. F1 scores are calculated by knowing the precision and recall of a classifier, so this proved to be a far better representation of how ‘accurate’ a model was than simply calculating the accuracy.

**Model Comparison**:

The four different approaches yielded varying results, which may or may not prove beneficial to Yelp as they implement their image classification machine learning model in production.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | MC-F1 | TS | HA | GFK | TR | OS | GFD | GFL | AC | RE |
| *KNN (Baseline)* | - | .85 | .73 | .54 | .75 | .44 | .68 | .04 | .49 | .55 |
| *NN Single Class* | - | .87 | .85 | - | .78 | .65 | - | - | - | - |
| *NN Multi-Class* | .51 | - | - | - | - | - | - | - | - | - |
| *CNN (2 layers)* | .68 | - | - | - | - | - | - | - | - | - |
| *CNN (Mock VGG16)* | .69 | - | - | - | - | - | - | - | - | - |
| *VGG 16* | NA | - | - | - | - | - | - | - | - | - |

*MF1=Multi-Class F1 Score, TS=Table Service, HA=Has Alcohol, GFK=Good For Kids, TR=Takes Reservations, OS=Outdoor Seating, GFD=Good For Dinner, GFL=Good for Lunch, AC=Ambiance Classy, RE=Restaurant Expensive*

Comparing our baseline KNN model to various Neural Nets, we can see that the baseline model outperformed our best performing neural net on certain classes such as 'has table service', 'has alcohol', and 'takes reservations.' However, our baseline model performed even worse than 50% on three classes ‘outdoor seating’, ‘good for lunch’, and ‘ambiance is classy’. Thus, our baseline is not very reliable when it comes to correctly classifying whether or not each of the nine classes is present in a given image. Perhaps with more training examples in certain classes, this problem would be alleviated, but for the moment I did not pursue a further solution to optimize the baseline.

Among our neural net models, the mock VGG16 that used 8 convolution and pooling layers possessed the highest F1 score. The intuition behind why this model out performed other methods relates to applying a filter repeatedly and pooling relevant features again and again so that the model could extract relevant features critical towards predicting a particular class. Taking this into account, it is likely the VGG16 Neural Net would outperform our current model, but more time is needed than available to train this model despite using 2 GPUs and 8 CPU’s on Google Cloud.

The neural net that predicted on individual classes also seemed unreliable because it predicted probability scores too close to zero or one to merit confidence it was actually predicting a probability. After trying multiple options for the L2 regularization, learning rate, and activation function, I decided to scrap this model entirely because of its inconsistent results. This is why there are certain classes that do not have scores in this model’s class categories.

**Results**:

At this time, the ‘Mock VGG Net’ that incorporated eight convolutional and pooling layers produced the best F1 score of approximately 69%. After manipulating various hyper-parameters, I found the following hyperparameters to be optimal:

* Filters = 32
* Kernel size = 3
* Stride = 2
* Padding = ‘Same’
* Kernel Regularization L2 = .01
* Activation = Sigmoid
* Loss = Binary Cross Entropy
* Optimizer = Adam
* Epochs = 10
* Metrics = F1 Score

Recognizing that my models generally improved with more convolutional and pooling layers, with more time, I suspect the VGG16 Neural Net would outperform my current best performing model. However, after trying multiple times to speed up the training process using more GPUs and CPUs, there was still not enough time to train the model to make predictions.

**Generalizability**:

One question worth considering is how well a human would complete the same task of the CNN? It is also worth noting that some of the classes are not simply looking for an object such as a beer bottle, rather they are more conceptual like 'takes reservations', 'good for kids', and 'has table service.’ How exactly would a human know from an image (such as a plate of food or picture of the interior of a restaurant) whether or not it took reservations? How well it suited children may be even harder to predict? The chance that a human could achieve an F1 score of even 50% on all nine classes for each of the 7000+ images seems unlikely. Thus, if such a task is extremely challenging for a human to accurately classify images in such categories how well can we expect a neural net to complete a similar task? Are there even enough features within the images for a human (let alone a model) to actually predict whether 'takes reservations is present? In this context, our model that predicts nine classes correctly nearly 69% percent of the time seems quite impressive. In fact, these results are so high it raises questions as to what exactly the neural net is extracting from each image in order to make this classification. With more time, I would like to further explore this issue using tools for Keras that will visually demonstrate what the neural net is picking up during the classification process by highlighting certain areas on the image. From such visually highlighted areas, I could make a better determination as to whether or not this is a reasonable feature for the neural net to be extracting based on a particular photo.

**Conclusions:**

The convolutional neural network that mimicked the basic architecture of a VGG16 Neural Net produced a realistic F1 score that seems slightly better than a human given a similar task. The baseline model proved too erratic in its predictions, and thus not a consistent technique Yelp could deploy into real world production. Understanding what features a neural net is extracting from a picture to predict certain classes is worth exploring in future iterations of the project, mainly in order to gain insights as to whether or not such features are relevant and realistic. Also exploring why the baseline model incorrectly predicted on multiple classes as opposed to others, and optimizing this model may prove to be a competitive alternative to the current CNN architecture.

For this particular task of classifying food, I believe humans do a better job than a CNN across the nine categories for two reasons: first because some of the classes Yelp chose to predict are more abstract than simply identifying an object (such as ‘takes reservations’), but second because most humans would hesitate to make a prediction with insufficient information, perhaps stating “I don't know. How could I know that based on this photo?” A neural net, on the other hand, will simply calculate a probability and classify the image. This “not knowing” is perhaps more valuable than a prediction in some cases because not knowing is information that can help researchers gather better, more informative data on that class, to improve someone’s ability to make an accurate prediction.