Checkin 2:

Introduction

The current problem we have is that in a highly competitive and non-differentiable marketplace, where there are millions of customers and restaurants, we want to figure out how a restaurant's image online affects their rating on apps like Yelp. We want to build a classifier to understand this problem, so that we can help restaurants build their image and online presence with the newest image advertisement techniques in our highly technological world today.

The paper's objective is to figure out the quality of a restaurant's image in relation to its Yelp rating. The first part of the paper focuses on creating a classifier that works to assess the quality of the restaurant's photographic representation online, by giving a restaurant's image as an input, and a Yelp rating of 1 to 5 stars as the output. The second part then focuses on determining which features of a restaurant's images can lead to a higher rating.

We chose this paper because in such a highly competitive market place, even the slightest changes in a restaurant's image can give it a marginally higher rating and lead to growth in revenues. Our goal is to determine first, how well an image can determine a restaurant's rating, and our second goal is to determine what and how the different features of an image can impact a rating. In this way, we can see how we can perhaps make businesses more effective online, and how as consumers who are picky eaters, may have to become more wary as businesses learn to become more effective in their online presence in a way that might be more unrepresentative of their restaurant as a whole.

Our project is a classification problem, as we are building a classifier that classifies images (input) based on their Yelp restaurant ratings (output).

Challenges

One of the challenges we encountered was combining our preprocessing and model codes. Since we are all working separately in different time zones (which is also one of our challenges), one group was working on preprocessing, while the other was working on the model. Thus, we had to make sure that the two were compatible and worked well together, which was definitely a difficult step in our project.

Another big challenge we faced was that our data file was too large, which made the running process too long during the preprocessing stage. Thus, we resolved this by doing a brief "preprocessing" stage for the preprocessing of our data, and separated our data into multiple datasets "drink.json", "food.json", "inside.json", "menu.json", "outside.json" which we can test on individually.

Another challenge was deciding on the number of layers we were going to use, and deciding on the overall structure of our model. We are still trying out different numbers for hyperparameters, and changing the structure within the call() function of our model.py.

Finally, one of the biggest challenges we faced was implementing the ResNet model. Although we expected this process to be relatively simple, it was very difficult to get it to work and are still addressing the issues. Therefore, we started out by writing our own model (using what we learned from our previous assignments), which ended up actually running and producing pretty good results (accuracy and loss). This ultimately made us change our base, target, and stretch goals, which are outlined in the last response (regarding the things changed in our project).

Insights

Our model currently has 3 convolution layers and 3 dense layers. We used a batch size of 100 and defined 5 classes each for 1-5 stars. We trained and tested our model for just one epoch on each of the different datasets so far and saw the following results:

Image type	**Number of images***	**Average loss**	**Average accuracy**
food	60559	1.0022095	0.65438014
drink	6052	1.0745429	0.6508333
menu	848	1.2926049	0.71
inside	27148	0.8805003	0.695
outside	7027	1.0501764	0.6714285

** The images with a star rating in between whole stars (e.g. 1.5) were omitted from training and testing.*

The model's accuracy is close to 70, and performs better than we had anticipated given that we have written our own model, but the accuracy is still not as high as what we would anticipate from a ResNet model for example. Since we have one model working so far, we expect to implement different models like ResNet and GAN and we expect the accuracies across these models to be a little higher.

Plan

We need to dedicate more time into researching more about implementing ResNet and GAN so that we are able to use different models to train and test. We also feel like we need to dedicate more time on adjusting our hyperparameters and the overall structure of our model (tweaking the number of convolution layers and linear layers) for the best outcome.

After more research, we found out that ResNet18 may not be the best model for our image classification project.
https://github.com/tensorpack/tensorpack/tree/master/examples/ResNet
The link above shows that Image Classification Training on ResNet18 usually outputs the highest validation error. Thus, we're thinking about using ResNet-101 instead.

Because we experienced difficulty with using the ResNet Model, our base, target, and stretch goal have changed as well. Our base goal is now implementing our own model and seeing decent results. Our target goal is now successfully implementing the Tensorflow ResNet model. Finally, our stretch goal would be to reach a good accuracy (close to Resnet) on our own model, as well as on a new model such as GAN.