Lab 4 Report

Ben Fu
EE 385T Intro to Machine Learning

Code

https://github.com/bennycooly/INF_385T_Intro_To_ML/blob/master/labs/lab4

Predictions

https://github.com/bennycooly/INF_385T_Intro_To_ML/blob/master/labs/lab4/vizwiz/results/predictions.json

Questions

1. Classification Using Hand-Crafted Features
    e. Method

I downloaded the full VizWiz dataset from the site as specified in the instructions. Then, I selected 1000 random annotations from the training set to train, 200 random annotations to measure accuracy on the validation set, and the first 30 annotations (non-random) from the testing set to make predictions.

Regarding feature extraction, I used a consistent method to convert the Azure API responses to continuous and discrete feature representations. The general idea was to make the features represent the actual important features from the image without overfitting. For the Vision API, I considered two options for the categories: one-hot encoding and taking the average score of the detected categories. However, since there are over 80 different categories, I thought that this would be overfitting the model and decided to take the average of the categories. I used this same average confidence method for the tags and description captions as well. For color, I was also considering one-hot encoding but decided ultimately to use the black and white indicator. For the description tags, I used the average of the captions as one feature and then the number of tags detected as another feature. For faces, I simply used the number of detected faces. Finally, I ignored adult because I found that it did not help the accuracy of classification models since none of the images contain adult content.

The text API feature extraction was much more straightforward. I used the same method of averaging the score for the languages feature. For the sentiment feature, I simply passed on the given sentiment score, which was a continuous value. For the key phrases, I simply used the number of detected key phrases.

To train the models, I chose to use three different classifiers: SVM, Decision Tree, and

Logistic Regression. Then, I used a majority voting ensemble to find the classifier that was most accurate. After training based on 1000 random annotations from the training set, I was able to achieve an accuracy of about 72% on the 200 validation annotations. I did not have the time to run a grid search to optimize all of the model's hyperparameters.

Ultimately, I was glad and almost surprised that the accuracy almost reached 75%. This seems to be a very difficult task for machines to solve, and being able to correctly detect if 3 out of 4 images are answerable is not bad. It would be interesting to see how the accuracy would have gone up if I used additional features and/or different classifiers.

2. Classification Using Neural Networks

Dataset: Breast Cancer (from Sklearn)

c. Optimal Hyperparmeters

I tested between 1-10 hidden layers and from 20-200 (increments of 20) hidden nodes in each layer. This was a total of 100 different hyperparameter values.

The optimal hyperparameters when using the hyperbolic tangent activation function and 100 epochs was 4 hidden layers of 160 nodes each, which resulted in an accuracy of 93.4%.

With the optimal hyperparameters, the total number of weights is 784 * 160 * 160 * 160 * 160 * 1 = 513,802,240,000 weights.

Full output:

```
[(20,), (40,), (60,), (80,), (100,), (120,),
{'hidden_layer_sizes': (160, 160, 160, 160)}
Accuracy: 0.934
```

d. Discussion

The MLP results seemed to align with what we learned in class. Similar to other hyperparameters we tuned in the last lab, the performance first rises between 1-4 layers and then dips a little, showing signs of overfitting.

With only one hidden layer, the model would often fail to converge (reported by sklearn). This may just mean that it did not reach sklearn's default threshold for convergence, because when checking the accuracies reported by the gridsearch, those models still achieved about 90% accuracy (even with one hidden layer of size 20 nodes), which is pretty good.

The accuracies for between 2-10 layers were roughly the same (90% or higher). Overfitting seemed to occur after 4 hidden layers, since the accuracy dropped from the highest mean accuracy of 93.4%.