The University of Toronto, CSC411

Assignment 3: Facial Expression Prediction

Haohan Jiang
999268750
Omri Myers
1001902177

December 11, 2015

1 Introduction

In approaching this assignment, we considered a number of different classification models. Since we did not know if the data set was linearly separable or not, we decided to consider both linear and nonlinear classification techniques.

First, we considered using a linear model, but we decided against using such a simple model to try and categorize 7 different classes. We did not believe that the linear model was capable of such a task considering the large variations in the photographs caused by different poses and different facial expressions.

Second, we considered using multinomial logistic regression. We believed that this would be a powerful method to use because multinomial logistic regression is used when the dependent variable is nominal, meaning that it falls into any one of a set of categories of which there are more than 2. [2]

Third, we considered using an artificial neural network. We considered this approach because of how powerful a neural network can be in classifying data that is non-linearly separable.

Forth, we considered using a support vector machine. We considered this approach because of the properties of a SVM. Specifically, the idea that the SVM constructs a hyper-plane to maximize the margin of separation between different classes.[1] We believe that maximizing the separation will result in a high classification rate.

Lastly, we considered using a ensemble method, specifically a bagging ensemble method because of the ability to combine simple classifiers to perform a powerful task such as image classification.

Constructing our final model: We chose to use the voting ensemble method to get the best accuracy and to keep our model as general as possible. After training each classifier using the bagging method, we combine the soft prediction with different weight for every classifier according to the accuracy we got on the validation set when testing it separately. We gave 0.1 for the K-NN, 0.3 for the Neural Network, 0.26 for the logistic regression and 0.34 for the SVM. The result of that process is for every data of the test set we have the distribution of possible values (1-7) then we simply chose the argmax of every test data and made it our final prediction. The weights were decided after looking at the observed classification rate for each individual classifier. Classifiers that had higher classification rates on the validation set were weighted more than classifiers that had lower classification rate. The reason we put bigger weight to the neural net than to the Logistic regression classifier is that we found out that SVM and the Logistic regression fail at the same kind of data and combining the weight of the neural net will give us better generalization.

2 Description

2.1 util.py

To extract the data from *labeled_images.mat*, we used scipy to convert the matlab file into a numpy matrix. After extraction, the data was in the form of a 32x32x2925 matrix. We then used the numpy library to reshape the data into a 1024x2925 matrix. We further processed the data by extracting the

identities of each data point to ensure that the same faces do not appear in both the training and validation set by using sklearn. cross_validation library function *train_test_split*. It also contains a function called *standard_data*. This function standardizes our data by subtracting each point with the mean, and dividing by the standard deviation.*fix_pixels* that activate equalize histogram on the data

train_test_split split the data into two sets, training and validation. We used 70% of the data set as training and 30% of the data set as validation.

2.2 Processing the Data:

During our experiment we dealt with the problem of having relatively small data set thus the classification rate for every classifier we tested was poor. So we had to come up with a way to increase our accuracy.

We found out that processing the images before activating the classifiers on them increased the accuracy. We compared three different data processing methods: data standardization, Equalized-Histogram, and Gabor-Filter. We also tried combine them. Unfortunately, we did not have enough computational power to use Gabor-Filter with the right frequency (it takes hours to run and created a huge dump file for the processed data), so we decided not to use it.

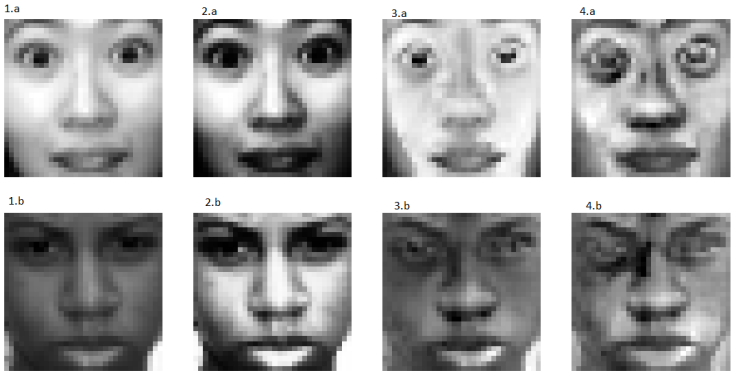


Figure 1: Processed faces

In the above picture, we can see the difference between three kinds of inputs used in our work. From left to right: 1.a,1.b Are unprocessed images 2.a,2.b Are equalized-histogram images 3.a,3.b Are standardized images 4.a,4.b Are both standardized and equalized images

2.3 run_classifier.py

This function contains the implementation of *K*-NN, multinomial logistic regression, SVM, and bagging ensemble method. All input data is standardized before fitting classification models to them.

2.3.1 *K*-NN

A *K*-NN classifier is a method for classify objects based on the closest neighbours in the training set. Training for *K*-NN consists of saving the training points and labels. A point is classified based on the class of the closest training point neighbours. The value of *k* determines how many neighbouring points to consider before returning the class the point belongs to.

The *K*-NN classifier was implemented using the sklearn library. We trained our model on 70% of the total data set and validated our model on 30% of the total data set.

2.3.2 Multinomial Logistic Regression

Multinomial logistic regression is used to model the relationship between independent observational data and a class that the data belongs to. This is a supervised learning model as each training point is marked with a training label.

This classifier was implemented in sklearn with hyper-parameters as followed:
1. L2 penalization of the weights
2. 50 training iterations
3. Newton-cg optimization algorithm
4. tol = .001 which is the tolerance for stopping

2.3.3 Support Vector Machine

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of k categories. SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. Our program uses SVC which is a similar method that also builds on kernel functions but is appropriate for unsupervised learning and data-mining. It is considered a fundamental method in Data-Science.

This classifier was implemented in sklearn with hyper-parameters as followed:
1. kernel='rbf'
2. C=50 ; Penalty parameter C of the error term.
3. We trained our model until convergence.

2.3.4 Bagging Ensemble Method

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it. We use sklearn ensemble library which implement BaggingClassifier. After trying different base classifiers (Logistic regression, and SVM) and find the best hyper parameters for them we trained two different bagging classifiers one for each of the base classifiers that increased the accuracy of our model.

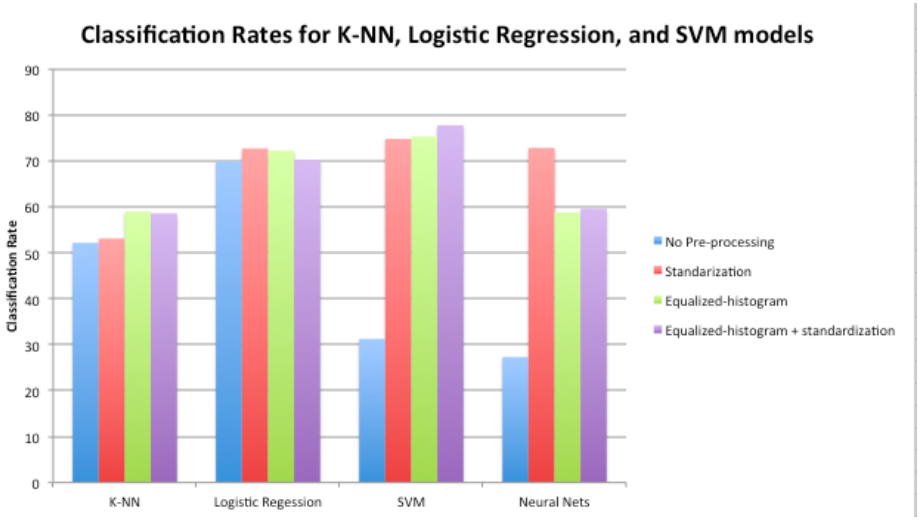
2.4 neural_nets.py

Neural nets are non-linear discriminative classifiers that utilize functions of input variables. Neurons are organized into layers: input, hidden and output. The input layer is composed not of full neurons, but rather consists simply of the record's values that are inputs to the next layer of neurons. The next layer is the hidden layer. Several hidden layers can exist in one neural network. The final layer is the output layer, where there is one node for each class. A single sweep forward through the network results in the assignment of a value to each output node, and the record is assigned to the class node with the highest value.

Our model based on a neural net with 1024 nodes in the input layer, 320 hidden layers with full connection, and 7 nodes on the output layer. We used softmax layer for the output and sigmoid function for the hidden layers.

3 Results

Using the above classifiers, after processing our data and training our classifiers, we observed the following rates of classification.



(a) Classification rate of different models

Figure 2: Classification vs data-preprocessing

From the graph above, you can observe that the SVM model with training data that is preprocessed using standardization and histogram equalization achieved the highest classification rate of 77.73%.

3.1 *K*-NN

Using *K*-NN as our baseline classifier, we observed the highest classification rate of 58.96% on our validation set with a k-value of 11. This rate was observed with training data that is preprocessed using standardization and histogram equalization. This rate is slightly off the baseline 60% classification rate. We believe that we observed a lower classification rate because of how the

training data was structured.

We believe there might have been noise in the data that had a large influence on the classification of the validation set. The noise could be caused by the color of the skin as that has no bearing on the class the image belongs to, but is part of the data.

3.2 Multinomial logistic regression

With our implementation, the multinomial logistic regression was able to converge in less than 50 training iterations. We observed a final classification rate of 72.68%. This is much higher than our *K*-NN classification rate of 58.96% and performs much better than the baseline 60%.

The above classification rate was observed with respect to standardized data. The standardization process is described above in section 2. If we did not standardize the data, 50 iterations was not enough for the model converge. Instead, we observed convergence at 100 iterations and a classification rate of 69.9%. Historiographical equalizing the data also did not improve the classification rate and resulted in a slightly lower classification rate of 71.26%.

The observed classification result from standardized data confirmed our earlier beliefs that a multinomial logistic regression model would do well in classifying different faces.

3.3 Support Vector Machine

Using a support vector machine, we observed the highest classification rate when data was preprocessed using historiographical equalization and standardization. This rate was 77.73%. We were surprised that a support vector machine returned the highest classification rate because

3.4 Neural Nets

Using a neural net, we observed the highest classification rate of 72.78% when data was only standardized.We found out that the configuration that change the most was the number of hidden layers we ran our model with 100 hidden layers but achieved only 67% on the validation set. After trying both sigmoid and than function for the hidden layers we chose the sigmoid layer which perform only slightly better. We were surprised that the neural net model only performed slightly better than the multinomial logistic regression model as we believed that the neural net model was going to perform the best.

3.5 Bagging Ensemble Method

The bagging ensemble method was designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting. Since our data set is relatively small we decided to use this method to make our final prediction.

Using this ensemble method, we observed a classification rate of 74.64%.

4 Conclusion

We believe that considering the limited data set we had and the limited time and computational data we had. We manage to perform well and achieve just over 75% percent on the hidden test. We found out how important is to pick the right image pre-processing tool and apply it on the input data as well as on the test validation data. Every classifier has the pre-processing tool that maximized the result and not just one that fits all.

5 References

[1] $http://shodhganga.inflibnet.ac.in/bitstream/10603/2538/15/15_chapter\%208.pdf$
[2] $https://en.wikipedia.org/wiki/Multinomial_logistic_regression$

6 Instructions

Our code uses the Scikit-learn python module. The Scikit-learn module requires Python 2.6 and above or Python 3.3 and above, NumPy, and SciPy. Scikit-learn must be installed before running any code.

To install Scikit-learn using pip, run the below command in terminal.

Listing 1: Pip install

```
pip install -U scikit-learn
```

We use skimage to process the data. To install skimage please run the below command in terminal.

Listing 2: Pip install

```
pip install -U scikit-image
```

We use pybrain for our neural net. Please install it before trying to run the code.

Listing 3: Clone pyBrain library from GitHub

```
$ git clone https://github.com/pybrain/pybrain.git
```

finally, to run the model: Put all the input data (the labeled set, public and hidden test in the same folder with all the .py files and the .dump file we provide) the type: python2 run_tests.py and follow the instructions on the screen.