CS3244 MACHINE LEARNING

NUS SCHOOL OF COMPUTING

HOMEWORK 3: LABELED FACES IN THE WILD

Team : PETS

Su Xuan, Akankshita Dash

A0119395W, A0132788U

Contents

1 INTRODUCTION

The Labeled Faces in the Wild dataset provided to us contains 7 classes, and hence can be treated as a multi-class classification problem. For this homework, we tried several different approaches - **Support Vector Machines, Decision Trees, XGBoost, K-Nearest Neighbors** and finally, **Convolutional Neural Networks** as well as **Simple Majority Voting**.

2 FILE LISTING

We submit the following files:-

- hw3-lfw.ipynb (our main submission, use for evaluation)
- hw3-lfw-non-neural.ipynb (other classifiers)
- hw3-lfw-xgboost.ipynb (XGBoost method, requiring external dependency)
- common.py (utility functions)
- preprocess.py (dataset preprocessing)

3 DISCUSSION

3.1 Dataset

Since the number of features (1850) greatly exceeds the number of data (966), there is a high chance of overfitting **[1]**. We decided to perform **k-fold cross-validation, normalization, regularization and finally, PCA (Principal Component Analysis)** to reduce the chances of overfitting.

3.2 Data Processing

We standardize the dataset to normally distributed data: Gaussian with zero mean and unit variance. We normalize the data before applying our learning models. We also apply 5-fold cross-validation.

3.3 Non-Neural Learning Models

For all these classifiers, we used sklearn's cross-validation module to compute the weighted F_1 score and accuracy score.

3.3.1 k-Nearest Neighbours

Nearest neighbour classifiers compute classification from a simple majority vote of the nearest neighbors of each point **[4]**. The classifiers have a free parameter to configure, k. We varied this parameter for a few typical values (1, 3, .. 13, 15), and examined the relationship between the F_1 score and the values of k. The F_1 scores fluctuate between 0.4667 and 0.5350,

and the optimal value for is F_1 is obtained at $k = 1$. The optimal value for accuracy is obtained at $k = 7$.

3.3.2 Decision Trees (with Adaboost)

We apply the Classification Tree model **[6]**. F_1 score for the raw training data was 0.4428, and accuracy hovered around 0.44. We performed Principal Component Analysis, hypothesizing that the low score was due to features being collinear/orthogonal. However, the F_1 score did not have any significant gains after dimensionality reduction and Adaboost, reinforcing our initial belief that decision trees are not able to model the data well.

3.3.3 XGBoost

We installed the the well-known gradient boosting XGBoost library to model our data. After regularization, the F_1 score was 0.6826, outperforming the previous two models.

3.3.4 Support Vector Machines

We predicted that non-linear kernels would give us better results as there are features which may not be linearly separable. However, we considered all 3 kernels in our analysis.

1. **Linear Kernel:** We started out with the linear kernel. The F_1 performance of a linear SVM is 0.8069 and the accuracy is 0.8084.
2. **Polynomial Kernel:** We experimented with increasing degrees and found that the polynomial kernel with **degree** = 2 has the best performance. However, it gave us results inferior to a linear kernel, with an F_1 score at 0.23 and an accuracy of 0.40.
3. **RBF Kernel:** The F_1 score produced by a RBF kernel was 0.8166, higher than the above two kernels. The accuracy was reported to be 0.8198. On Kaggle, this SVM gave us the best result so far, at 0.8371. We hypothesized that due to overfitting and a large number of features, we could further improve the classification by finding the optimal hyperparameters. Through **GridSearchCV**, we found the optimal hyperparameters and applied PCA for dimension reduction. However, the results continued to be the same, showing only a minor improvement on our accuracy on Kaggle.

3.4 Neural Learning Models

We considered Convolutional Neural Networks (CNNs) which are particularly suitable for tasks related to image processing **[8]**. The popularity of CNNs in computer vision tasks led us to believe that they will have better performance as compared to non-neural models [10].

We note that in contrast to the aforementioned non-neural models which do not necessarily require any transformation done on the input data, for CNNs, it is necessary to reshape the data into its original form - 2D images - since convolutional operations are originally meant for 2D structures.

3.4.1 Data Augmentation

Considering the small size of the dataset and the nature of CNN models, we decided to experiment with **data augmentation** so as to increase the dataset size for better performance **[9]**. We examined the grey-scale images, and found that an easy way to increase the number of data instances is to horizontally flip the images. This mirroring operation increased our data from 966 instances to 1932. Moreover, we attempted another way to transform the images by rotating them for up to 30 degrees. We did not include this second transform in our final submission due to time constraints.

3.4.2 Models

We experimented with three different CNN architectures which have increasing complexity. The first model consists of one single set of convolutional, maxpooling, dropout, and fully connected layers. The second model includes two such sets, and the third model consists of three. Experimental results show that CNNs present great improvements over the non-neural models. The first model has an overall F_1 score of 0.8784, the second model has an F_1 of 0.9126, and the third model brought the overall performance up to 0.9379 reported F_1.

3.5 Simple Majority Voting

We experimented with an ensemble method, majority voting, in the hope that using the neural and non-neural classifiers together would combine advantages from both worlds, thereby giving further improvements. We experimented with five models: three CNN models together with two best performing SVM models. The idea is to compare the predictions from these five models and take the simple majority predictions. If there is no majority, take the median value. This ensemble classifier gave a result of 0.9378 on Kaggle.

4 CONCLUSIONS

We obtained the highest training data performance at 0.9347 for our ensemble model (submitted) and highest F_1 score on Kaggle - 0.95652 - using our third CNN model, under suitable parameter tuning. Hence, we conclude that Convolutional Neural Networks can generalize a good model for this dataset **[10]**.

5 WORK ALLOCATION

We split our work based on model exploration, implementation, experimentation and documentation. One member focused on the non-neural methods while the other implemented the final model used for evaluation.

6 STATEMENT OF INDIVIDUAL WORK

[S.X and A. D] We, **A0119395W,A0132788U**, certify that we have followed the CS 3244 Machine Learning class guidelines for homework assignments. In particular, we expressly vow that we have followed the Facebook rule in discussing with others in doing the assignment and did not take notes (digital or printed) from the discussions.

REFERENCES

[1] https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/
[2] http://scikit-learn.org/stable/modules/preprocessing.html
[3] http://scikit-learn.org/stable/modules/cross_validation.html
[4] https://nlp.stanford.edu/IR-book/html/htmledition/multiclass-svms-1.html
[5] http://scikit-learn.org/stable/modules/neighbors.html#neighbors
[6] https://citeseer.ist.psu.edu/myciteseer/login
[7] https://nlp.stanford.edu/IR-book/html/htmledition/multiclass-svms-1.html
[8] http://www.sciencedirect.com/science/article/pii/S0031320301001789
[9] https://machinelearningmastery.com/image-augmentation-deep-learning-keras/
[10] https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/