## Machine Learning Report

# Exercise 3 – 3.8 Deep Learning

For this exercise the task 3.8 Deep Learning was chosen, to compare traditional feature extraction with CNN architectures. As datasets the suggested datasets “CarData” and “FIDS30” were chosen.

## Traditional Feature Extraction

For traditional feature extraction the suggested way from TUWEL was chosen, based on color histograms etc. Additionally, a feature extractor based on SIFT and subsequent visual bag of words (VBOW) was implemented. The implementation was done in Jupyter notebooks, one per dataset and one per traditional feature extraction approach.

## 2.1. Cars

For the cars dataset *Figure 1* shows the performance of different classifiers based on different way of constructing the training data. The classifiers were used with the scikit-learn default parameters. *MLP\_OPT* and *RandomForest\_OPT* indicate runs with optimal parameter settings obtained using a exhaustive grid search. The different raining sets are the same, as they are constructed and named in the respective TUWEL example notebook. Every experiment was conducted using 10-fold cross validation.

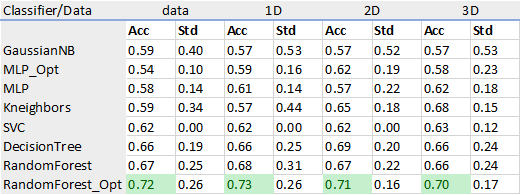


Figure : Performance per Classifier and training set

*Figure 2* therefore shows the first 15 results ranked by mean accuracy achieved with Random Forest Classifier. The optimal parameter settings therefore are default settings (*max-depth=None, max-features=”auto”*) and settings *n\_estimators* to 300. As *Figure 1* shows the performance can be slightly improved with optimal parameter settings (~+5% accuracy).



Figure : Results of grid search for RandomForest (first 15 ranked)

*Figure 3* shows the first 15 results ranked by mean accuracy achieved with MLP. However when using these optimal parameter settings back in the original experiment shown above (*Figure 1*) leads to worse performance, than using default parameters, although the result of using the default settings is not residing in the top 15.



Figure : Results of grid search for MLP (first 15 ranked)

The performance per training set is shown in *Figure 4,* whereas *Figure 5* shows the performance per classifiers, grouped by training set. As already indicated by *Figure 1* the best performance was achieved using Random Forest with optimal parameter settings. Surprisingly not only GaussianNB, but also MLP shows the worst performance. As expected, a higher number of channels improves performance.

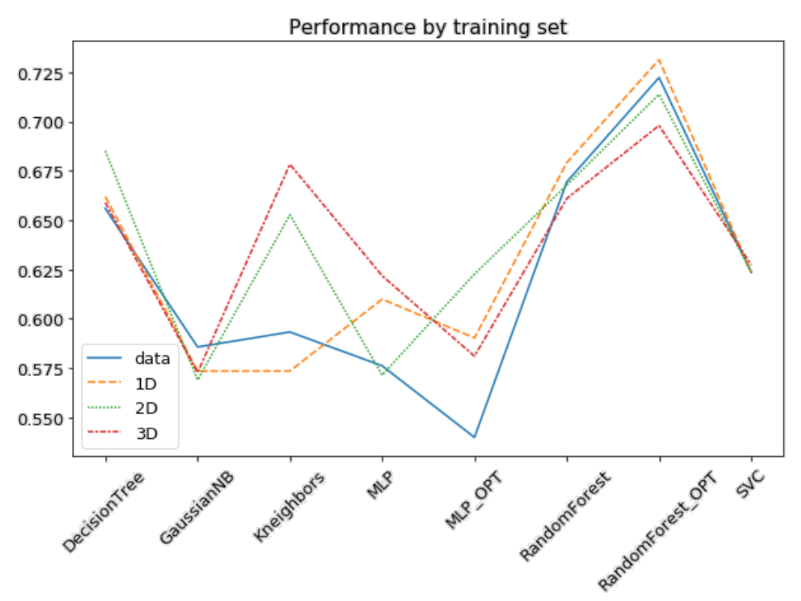


Figure : Performance by training set

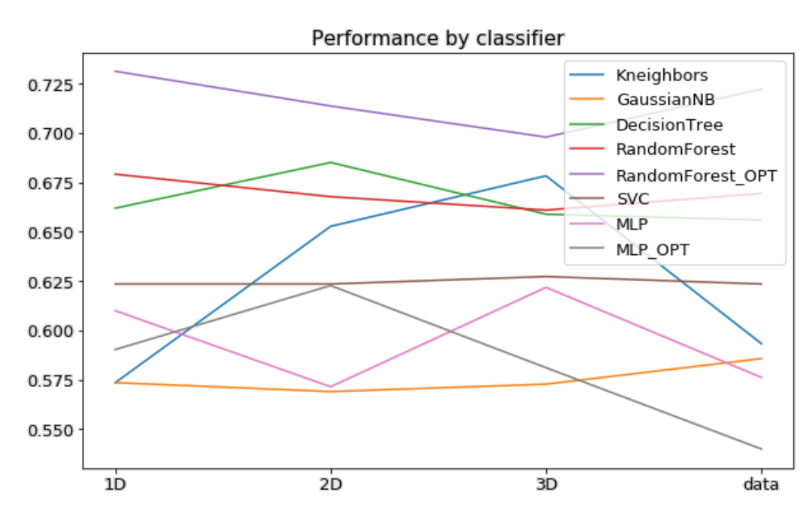


Figure : Performance by classifier

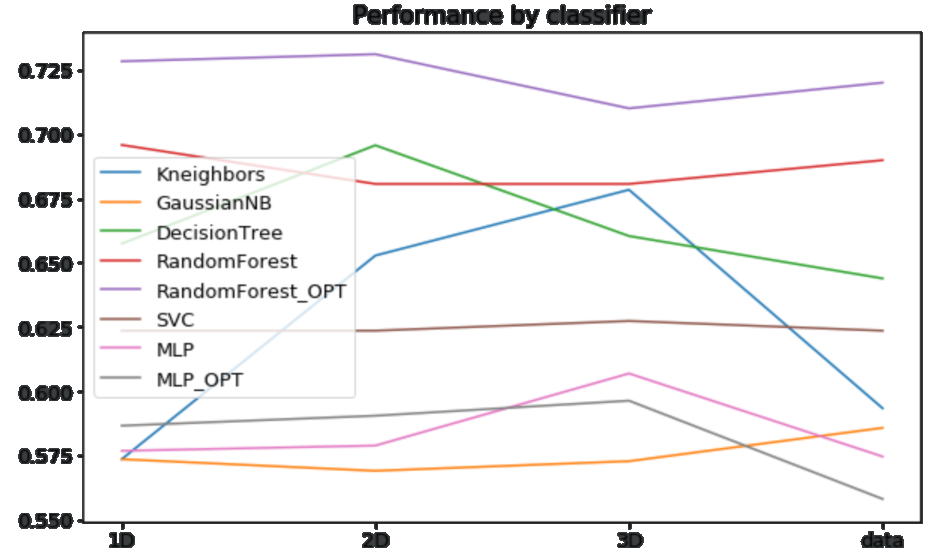
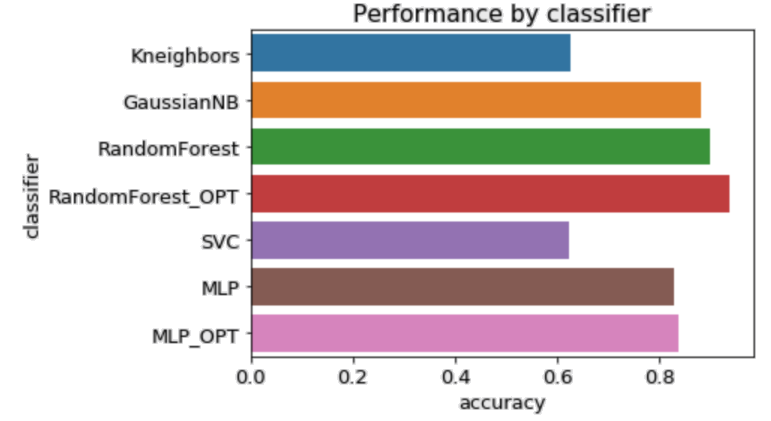
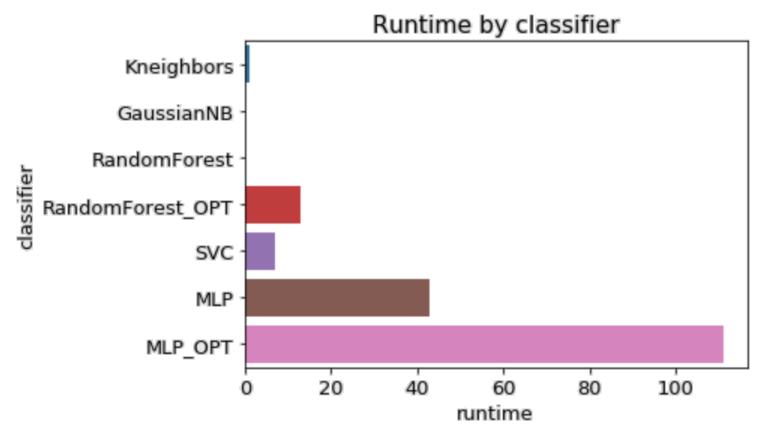


Figure : Runtime by classifier

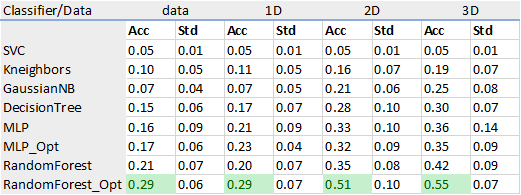


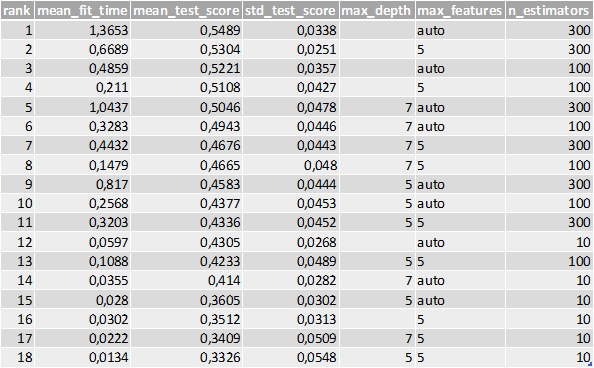


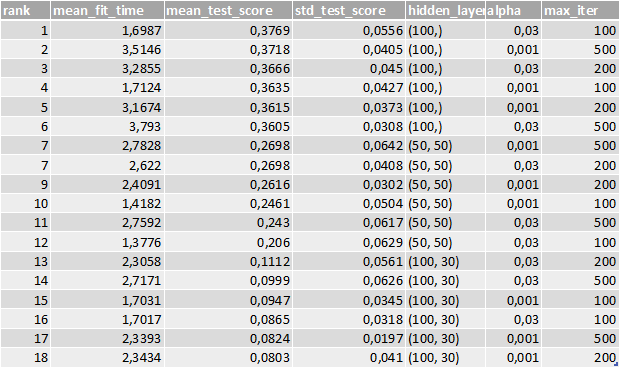


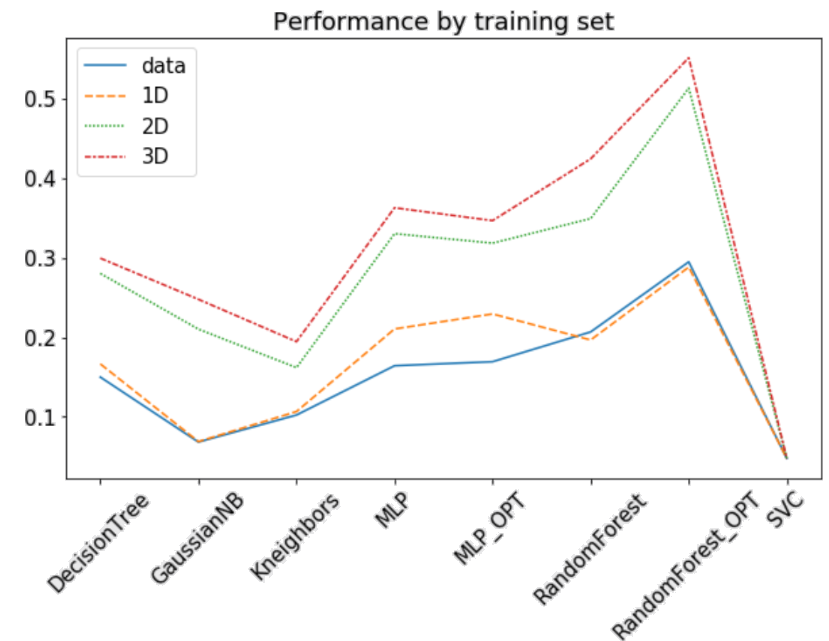


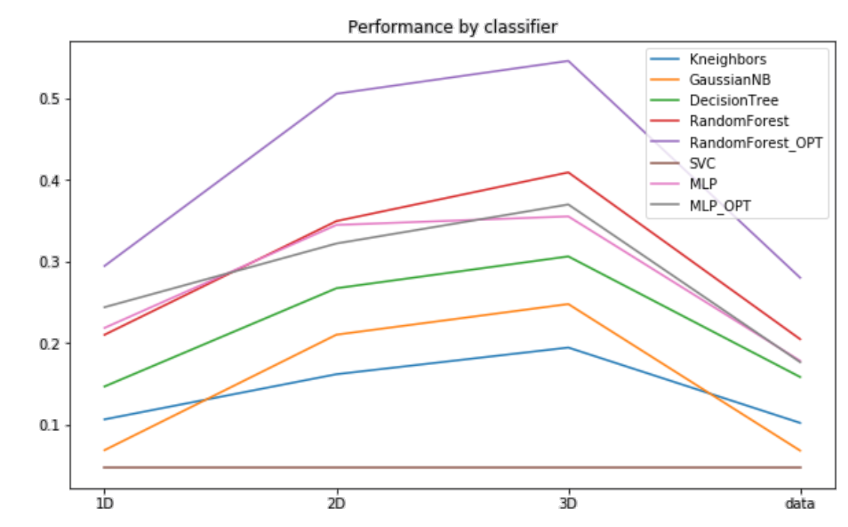
Fruits:

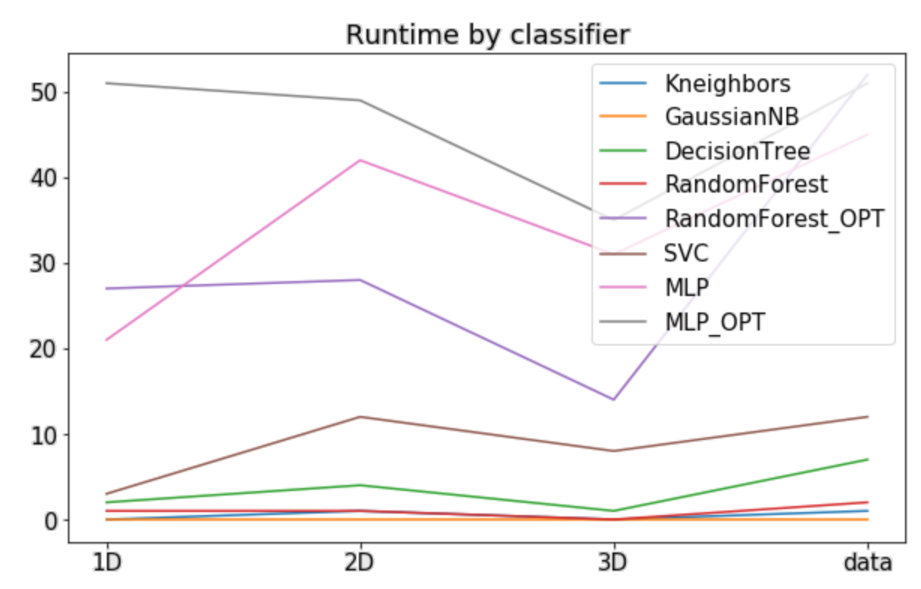


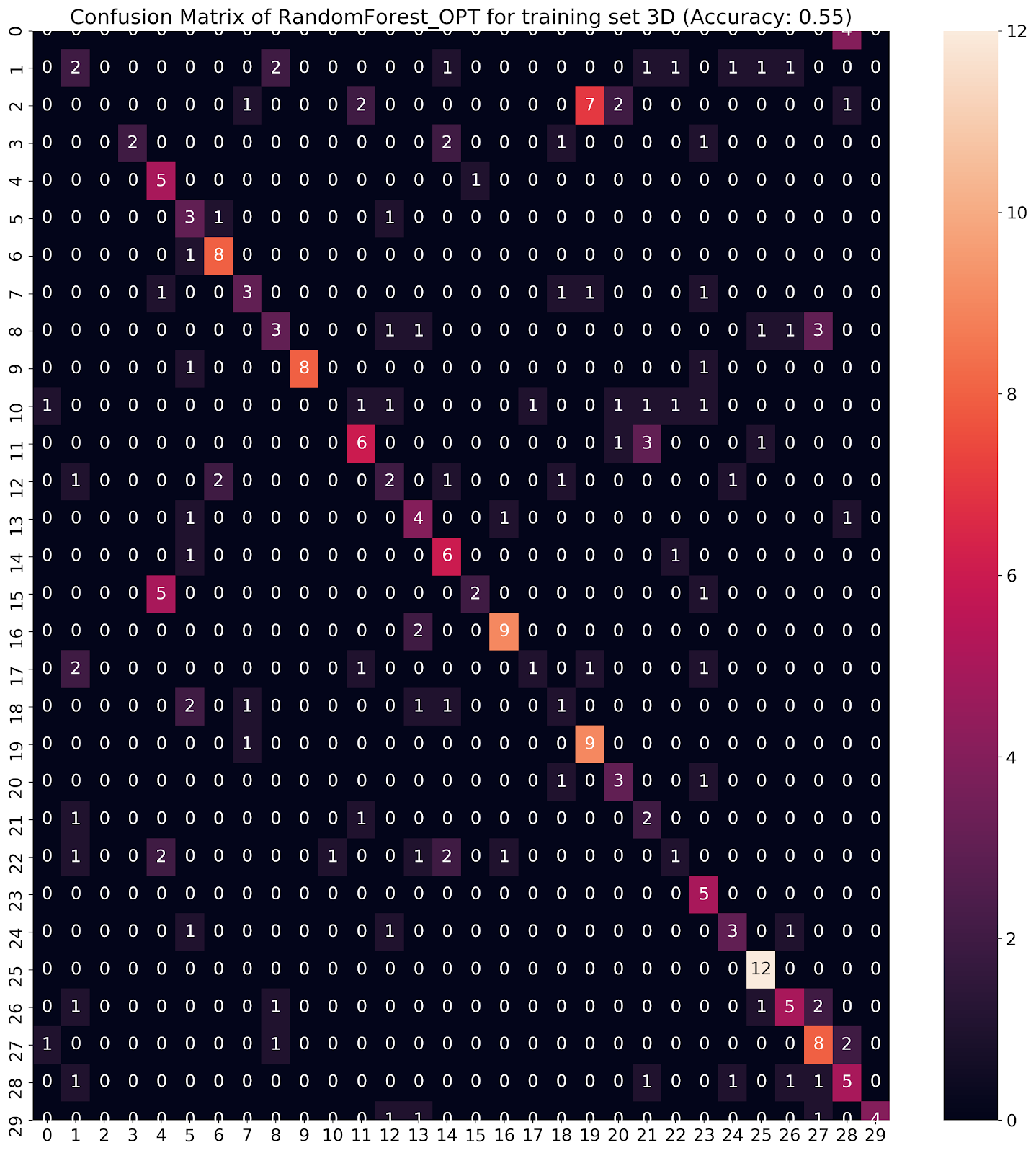


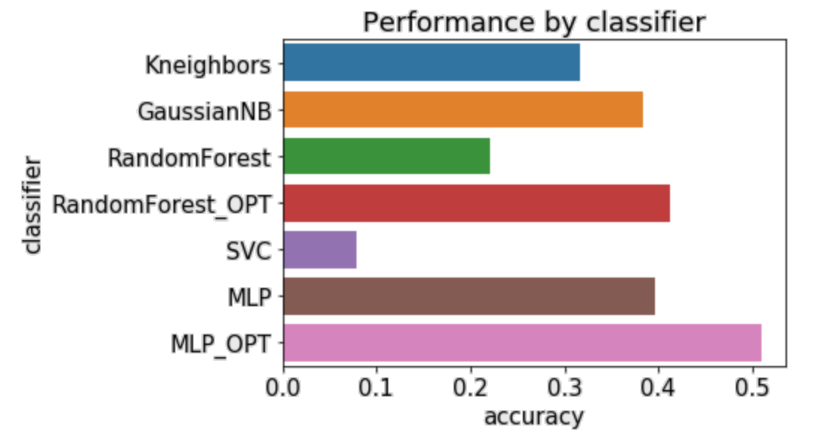


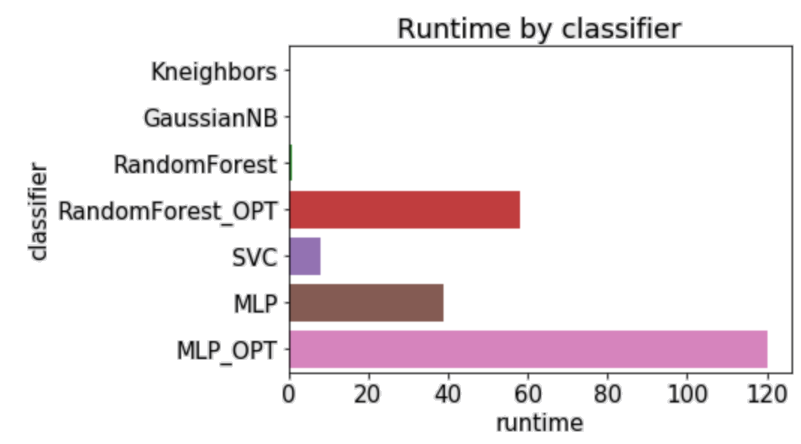


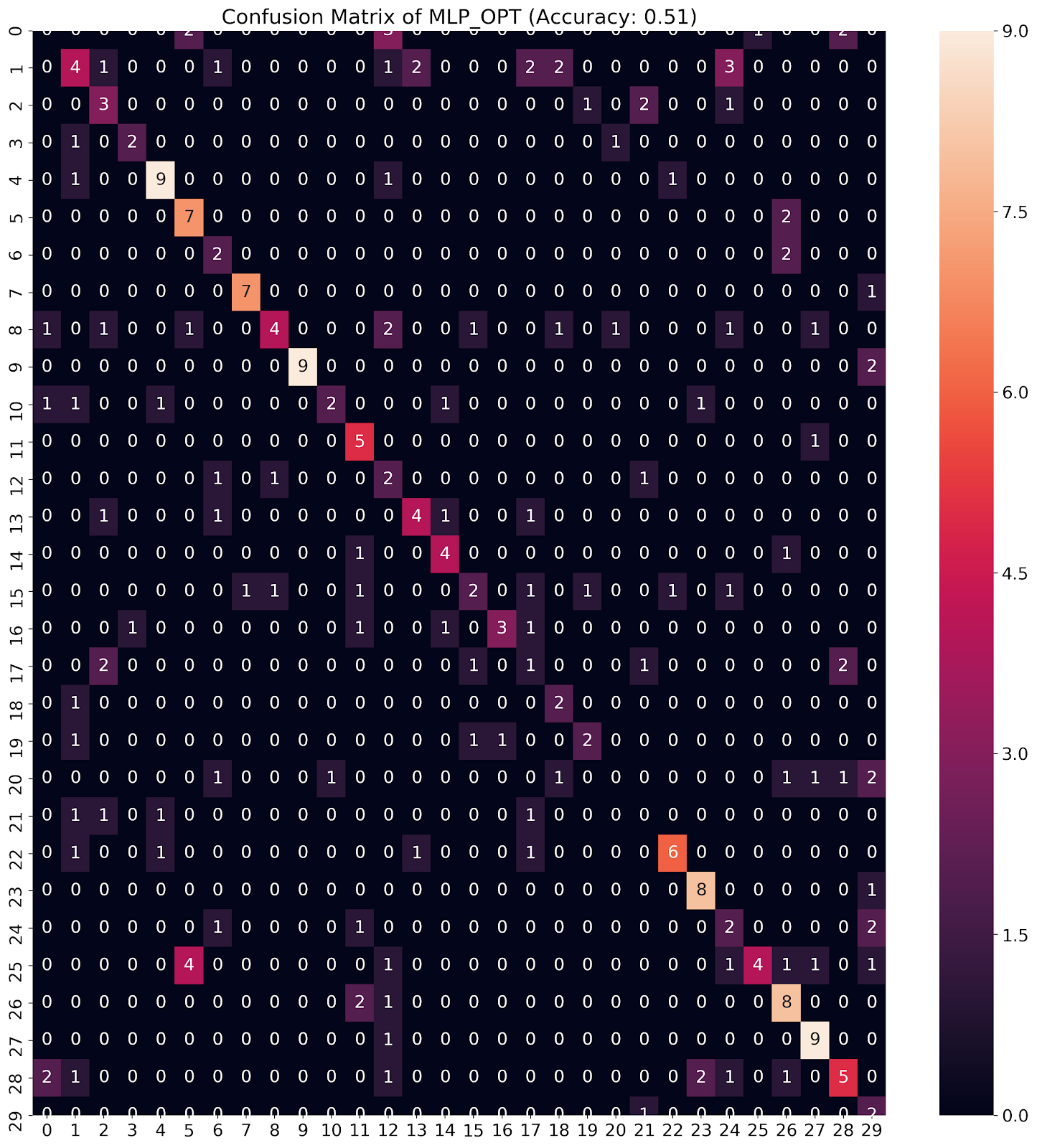












## Convolutional Neural Network