Machine Learning: Image Classification

Tanuj Tanuj: *SNR* - 2038893 / *ANR* - 653883
Shreshtha Sharma: *SNR* - 2037590 / *ANR* - 434770
Max Knegt: *SNR* - 2041458 / *ANR* - 733570

Tilburg University
Machine Learning
Master DSS
Group Challenge

Group 45

June 26, 2020

Data Preparation (Shreshtha)

To start the project, data was loaded and prepared by means of data preprocessing. First, all features and labels were extracted from the "*training-dataset.npz*" dataset which consists of 124,800 unique image arrays of shape 784. These images are paired with 26 unique labels ranging from 1 to 26 where each number represents its respective alphabet index. To correctly identify images, the array dimension of 784 was reshaped into 28 x 28 which in turn allows for displaying recognizable images.

Next, data was splitted across train, validation, and test splits using stratification sampling. Stratification was chosen to ensure that each label would be trained fairly, even when underrepresented. Two baseline values have been tested, first, a 70-15-15 split was used which resulted in a size of 87,360 training features. Although this was perceived as an adequate size for reliable classification, another split of 80-10-10 was used to verify for optimal performance. But, the former one was chosen as the final baseline as it was performing better in terms of accuracy. This approach applies to both the tasks.

Task 1: Neural Network Classification (Tanuj)

As a model, the neural network (CNN) was chosen. Although a decision tree and KNN were trained, results of the CNN returned the highest accuracy on validation data. Therefore this task will only focus on the application of CNN. In all the models that were trained, four convolution layers were used. After each two layers, the results were pooled in a separate layer to reduce computation capacity and to account for overfitting. Within each layer hyperparameters have been tested to find what values result in an optimized accuracy on the validation set.

The hyperparameters that have been tuned within the convolutional network are filters, kernel size and strides. Aside from within the network, additional tuning was involved in the model prediction through setting a variation of epochs. The epochs however were paired with an early stop function to prevent overfitting. As a result, the model often interrupted the prediction as it had already found an optimal value. The activation of all layers was by "*relu*" function and padding was set to "*Same".* For padding, *"Valid"* was used initially but this method proved to make the model less accurate and was computationally expensive. Further, "*Softmax*" was used for the output layer because it provides a confidence weight useful for multiclass classification.

Results Task 1 (Max)

The best performing model was run with ten epochs, filter values of 96, a kernel size of (3,3), and strides of (1,1). These are the default values as set in the attached python file "*ML_Challenge_Team45_Task1*" and can be reproduced by running the code. An early stop will interrupt the model after it has achieved the optimal fit as shown in figure 1.1. An accuracy of 94 percent is achieved for the validation set, and by evaluating the test set an even higher accuracy score of 95 percent is observed. The model as discussed was able to correctly predict 17,792 out of 18,720 test images.

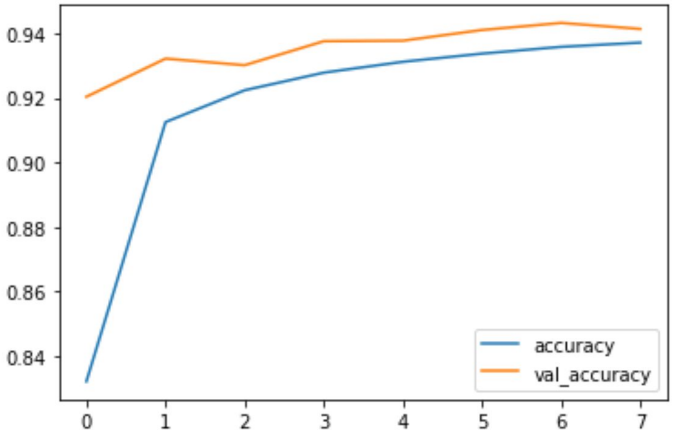

Figure 1.1: Training and Validation Accuracy scores for the best performing Model

Other hyperparameters that were tested included wider convolutions such as a kernel size of (6,6) in the first two layers and (5,5) in the last two layers, paired with varying strides of (2,2) and (1,1) in their respective two layers. Or a slower moving kernel of (5,5) in all layers with a stride of (1,1). However, neither of these models matched the highest validation score as they scored 92 and 93 percent respectively. Other values were tested as well, but none were found that matched the best performing model.

Task 2: 5 Letter Classification

Noise operations (Max)

Task 2 consists of 10,000 features where each instance represents an 30 x 168 image in which five letters can be recognized. Additionally each feature is presented with salt and pepper noise complicating the classification process. To handle the noise, two methods have been attempted. First, salt and pepper noise was added to the training set so that it became representative of the

actual test set. However, for this to work, the validation and test set for task 1 will also require added noise so that the data remains similar. Secondly, noise was removed from the test data using a Median Blur function, using this method all salt and pepper noise was removed while the training set remains untouched. This approach faces some issues where letters are partly covered because of a weaker activation rate after blurring. After comparing both methods on similar models, it was expected that the model trained by adding noise was more accurate in predicting the test images as part of task 2. This hypothesis is based on manual testing which consisted of randomly selecting indexes from the "*test-dataset*" file and comparing its image with the predicted letter.

Data Augmentation (Tanuj)

Further data augmentation technique was also used by creating a subset of each train image into five combinations. Each combination represents one pixel shift to the top, bottom, left, right and the fifth combination is treated as the actual image.This method was used for both the cases of train dataset, one with the noise and one without it. This new subset of 436,800 train images was used to train the classifier. This also resulted in an accuracy of 93 and 94 percent on validation and test set respectively for the trained data without noise, and 94 and 95 percent for the noisy trained data. The trained model was further used to evaluate the test images but the results for both the cases were not satisfactory based on the eyeball test performed on a sample of test images. The code used to perform the above operation has been included in the Task 2 file under Additional Content at the end.

Top 5 Accuracy (Shreshtha)

Before making predictions on the test images, each test image was splitted into a subset of (28,28) pixel images. Then, the trained model predicted each of these subsets and consolidated the results per test image. Since the model allows for looping over each letter now, a top 5 accuracy can be generated for all features in the test set. For each letter that is evaluated, the five most likely labels are returned based on their probability (confidence score). Leading zeros were added to labels ranging from one to nine to match the requested format of the output. With the help of some array operations, the output is flipped and transposed so that the order represents the five most likely letters in a row where the order of letters in the image matches with the index order in columns. This and the above discussed noise operations can be

reproduced by running the code enclosed in submission as given in "*ML_Challenge_Team45_Task2*".

Results Task 2 (Tanuj)

After reducing the number of dimensions and combining each five rows so that one row consists of five classifications of one image, the final outputs are saved to the csv file. Within this file, 10,000 rows can be found where each row represents the top five accuracy from one image. The first ten digits represent the most likely combination and as the column index increases, the less likely the combination is.

In an eyeball check performed by the group, it was observed that the model failed to return a correct classification on all five letters in the most likely prediction. However, there is a good chance that the most likely prediction involves on average three correct letter predictions within an instance and for some images even four predictions were classified correctly. Further, the second most likely prediction often involves one or more of the letters that were classified incorrectly at first which could indicate that the model is only slightly off in these cases. Therefore, the top 5 accuracy can involve all correct letters, but not in the same prediction instance. The enclosed submission "*Top_5_Accuracy_Group45*" includes all 10.000 predictions made by the model. Further tuning is possible to optimize the top 5 accuracy predictions.