**Group Name: Literally a placeholder**

1. Group Organization

   In the beginning we had a kick-off session to identify and collect the first tasks. These were mostly of administrative nature such as Github organization setup, repo setup, ilias group sign up etc. We used the Github Task tracker as our project management tool.

   At the beginning of each assignment, we met in person and walked through the assignment tasks. Then we tried to identify tasks or rather stories that were more or less independent of each other. For example, for the keyword spotting, we created a task for the preprocessing, the algorithm itself and the evaluation part. These large tasks were assigned to subgroups of 2 people, who then took care of any subtasks that might arise. We clearly defined what inputs and outputs were to be expected from each sub-team, so that we could work independently and reduce dependencies. This proved to be a good and efficient system.

2. What worked, what did not work

   In the beginning, the task distribution and the entire planning was a little difficult, because we underestimated how well defined these steps have to be to enable independent development and combining everything to a complete workflow afterwards . After the first assignment, the group organization improved and the workload was more balanced. We also planned some spare days to run the algorithms in case we would encounter such situations again.  Hence, the other assignments worked quite smoothly.

3. Task walkthrough

   **First assignment - MNIST**:

   *Approach:* for the SVM we used a 3-fold cross validation and GridSearch for the parameter tuning. For CNN we completed the DeepDiva implementation. Building a MLP for MNIST was an

exercise in the Advanced Topics in Machine Learning course. So we just had to adapt it to the data format.

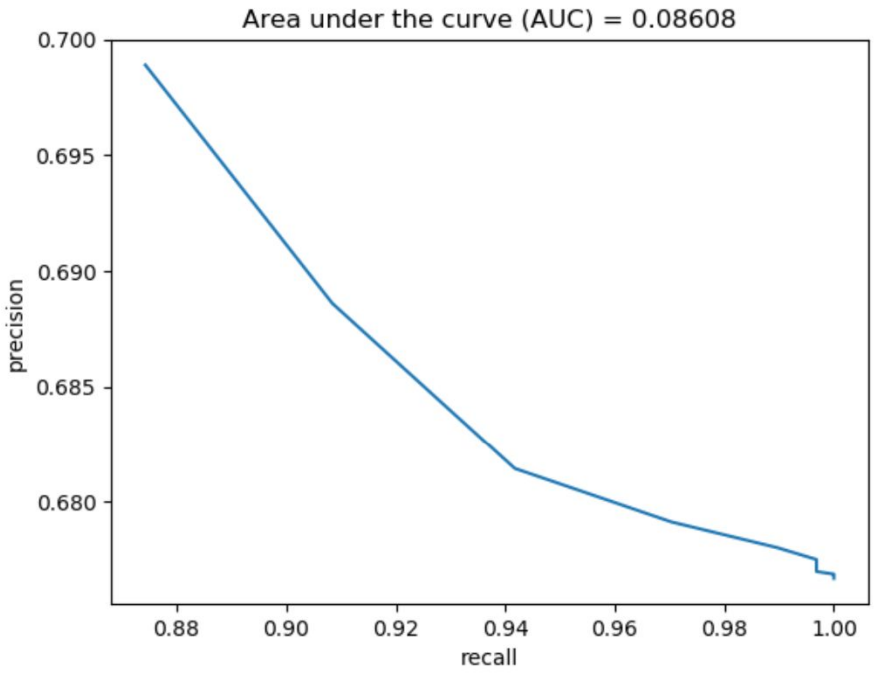
*What worked, what did not:* We were still getting used distributing the workload evenly and we found this rather difficult.. The tasks were strongly independent of each other and it was difficult to estimate how much work each - SVM, MLP and the CNN with deepdiva would generate and especially how long these algorithms would have to run on the full dataset. We split the team into three subgroups (one person for SVM, one for MLP and three for CNN). We used only a small slice of the MNIST dataset during the development. The MLP finished in time. For the SVM it worked on the smaller dataset, but the parameter tuning for the full dataset simply took too long. After about 72h we had to cancel it, for smaller datasets (about 10k) it worked just fine, but seemed to grow exponentially with additional data.
Working with deepdiva, we had some problems in understanding what we should do, i.e. what is expected to be implemented. However, we managed to implement it with some additional group effort.

**Second assignment - Keyword Spotting**:
*Special about solution: We further normalized the dissimilarity measures resulting from the dynamic time warping, to accommodate for variances in the range of distance values, which tend to be generally higher/lower for certain types/structure of words. Further we implemented simple multicore processing for the DTW step, splitting the task by keywords searched, which massively improved runtime.*

*Approach:*
For our second task, we received handwritten letters and corresponding svg files, containing vectors that would assist us in cutting out words and naming them. All the words were cut out, transparent borders were removed and each image was resized to

100x100 pixels. The PIL and  svgpathtools were used to perform these steps. The pictures were then exported as png files named according to the cut vector. These images are then featurized with simple summary statistics of each 1px slice in the x-axis of the image and compared to each other by dynamic time warping to get dissimilarity measures.

*What worked, what did not:*
The calculation of the AUC (which should be ca 0.6) is too low; so, something didn't work with the calculation.




Furthermore, the simple multicore processing lead to slight inconveniences in combining and saving the results, since each subtask by design had to be independent of each other, thus results had to be stored separately and combined after the calculation.

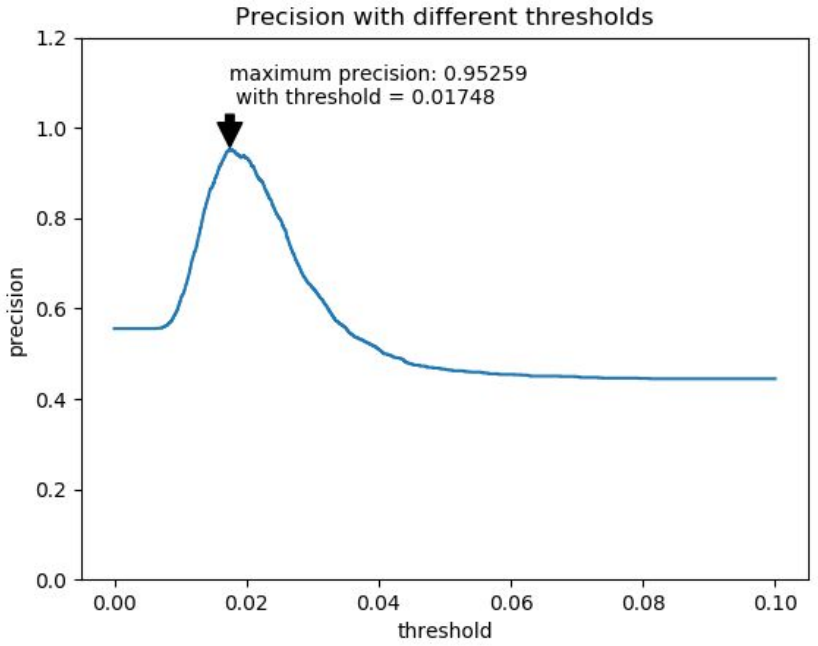
**Third assignment - Signature Verification**:

*Special about solution: Same as in the KWS task, normalizing the resulting distance measures, here to accomodate for the different feature sequence lengths (signature length).*

*Approach: Since the input data are already features, no particular preprocessing was needed, only velocity features calculated based on the x,y-positions and the time were added. Then comparison by dynamic time warping, as in the KWS task, but without multicore processing, since speed was not that big of an issue.*

*What worked, what did not: Everything went rather smoothly, since we did not change the approach much from the approach of the KWS task.*

*Results:*
After Normalizing the data we calculated the precision with thresholds between 0.0 and 0.1 with a step size of 0.00001 and found a maximum precision of 95.3% at threshold 0.01748 (Fig. below)




4. General Thoughts:

All the assignments took up quite some time. We think that it would have encouraged us, if they had an effect on the final grade. .