Lab 3 report

Hidden Markov Model

Authors:
Hussein AWALA
Amine BIROUK
Bonpagna KANN
Assem SADEK

Supervisors:
Prof. Jean-Baptiste DURAND
Prof. Xavier Alameda-Pineda

February 3, 2019

Contents

1 Hidden Markov Model

Hidden Markov Model, is one of the ongoing research topic. It's one of the sequential models that help to model a sequence of data over time, like in the application of genes expression, speech recognition and many more.

In the following report, we will answer the questions on Hidden Markov Models lab using different data, and comparing between models using different emission distributions.

1.1 Lab work

1. After plotting the letters A and L, we show in figures 2 and 1 the plotting of letter A with and without connections respectively. On the other side, we show in 4 and 3 the plotting of letter L with and without connections respectively.

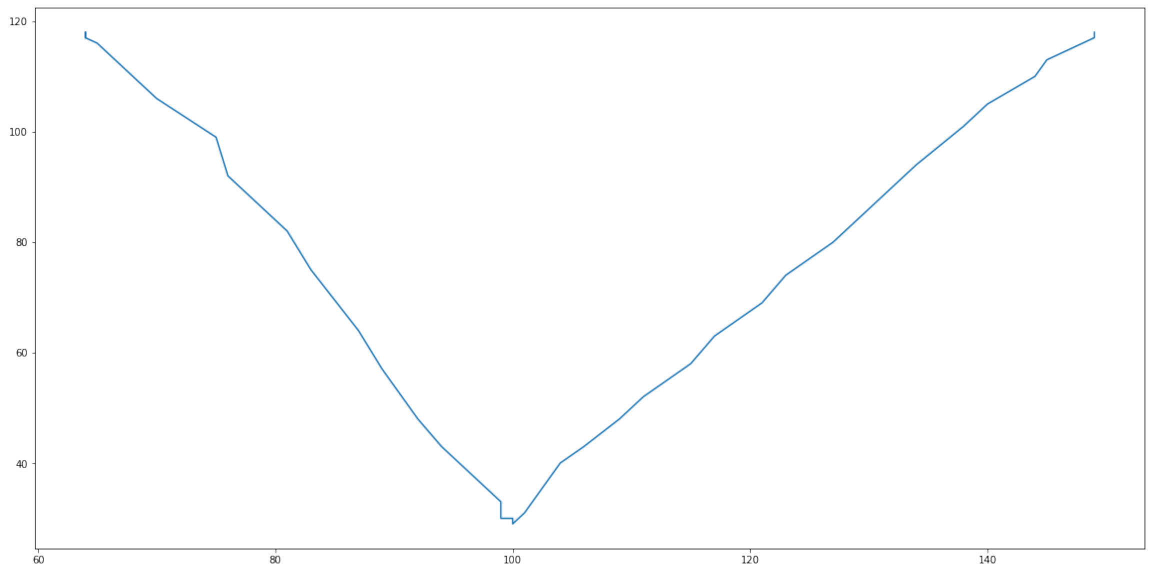

Figure 1: Plot for letter A, with connection

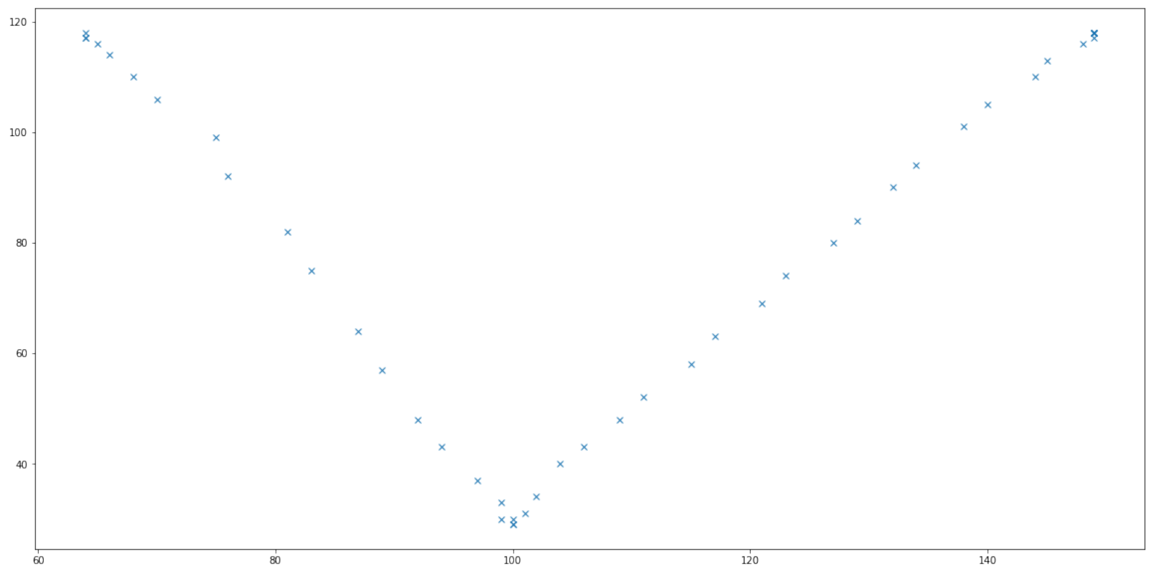

Figure 2: Plot for letter A, without connection

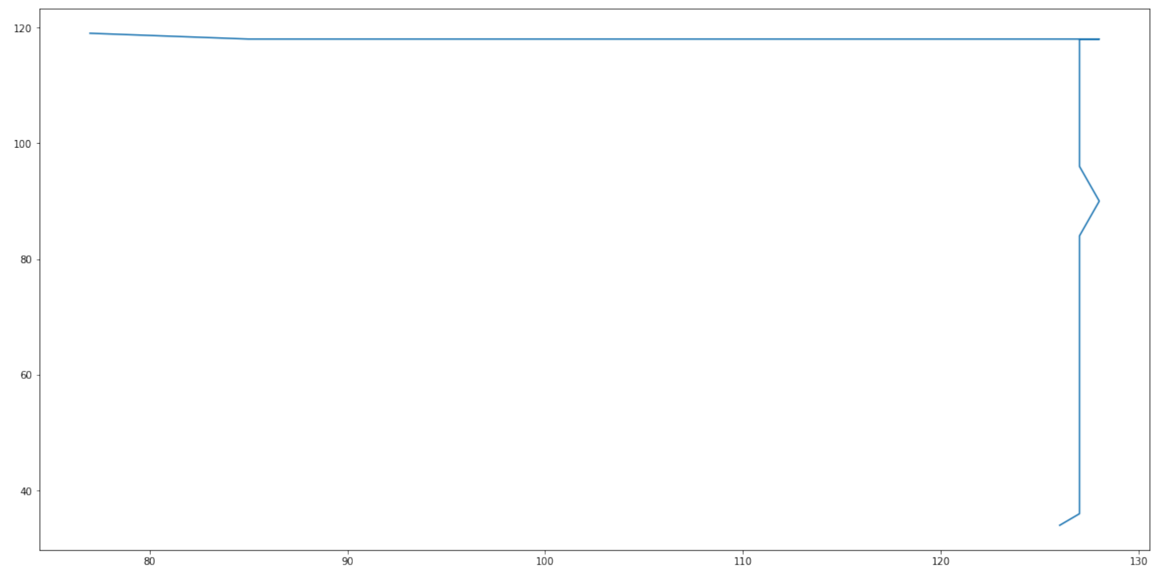

Figure 3: Plot for letter L, with connection

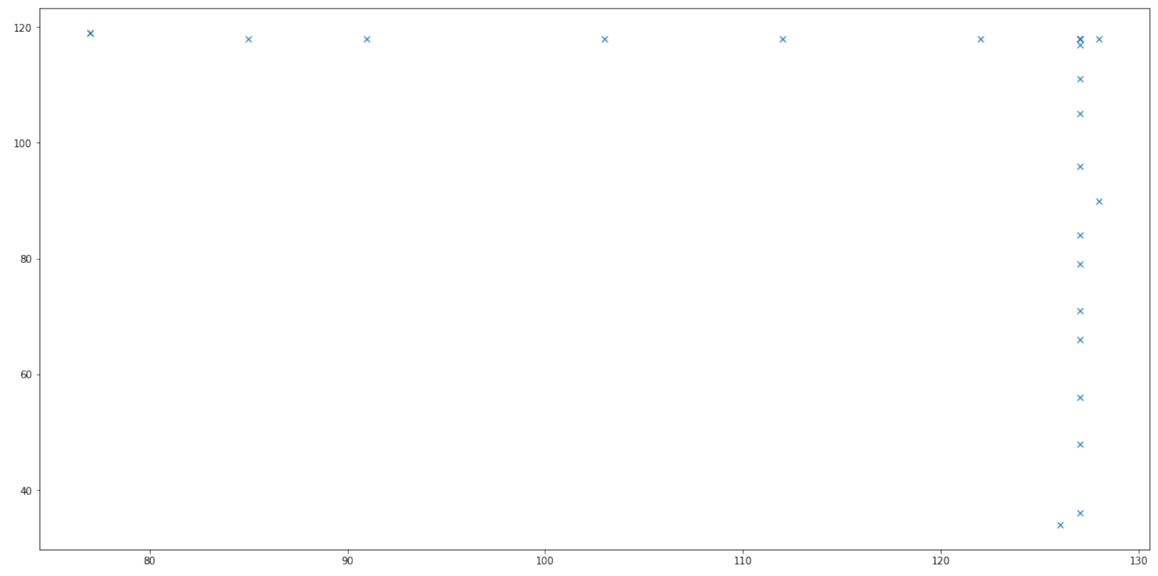

Figure 4: Plot for letter L, without connection

2. We defined the HMC model with 2 states for the latent variables, because we can see that we can classify the strokes of the letter A in two main diagonal directions (from upper right to lower left, and from lower right to upper left). Thus their parameters are:

   $$\pi = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

   $$\mathbf{A} = \begin{bmatrix} 0.95 & 0.05 \\ 0 & 1 \end{bmatrix}$$

   The emission distributions are Gaussian with parameters:

   $$\boldsymbol{\mu_1} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \boldsymbol{\mu_2} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

   $$\boldsymbol{\Sigma_1} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \quad \boldsymbol{\Sigma_2} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

   The reason behind the choice of these values are as follow:

   (a) For the initial latent variable probability, one component of the vector has the full probability because it will force the model to start with a initial state, in our case, it will start from upper right going to lower left (we noticed that the writing of inverted letter is started from this side).

(b) For the transition matrix, we force the transition from the second cluster to the first cluster to be zero and the full probability go always to the same cluster. This means that we are already writing the second diagonal of A, thus no need to go back to the first cluster. On the other side, the first cluster after a certain moment, it will have to make a transition to the second cluster to continue drawing the V-shape.

(c) Concerning the mean of each generated vector, under certain cluster, we find that the most representative vector for the first cluster will be a vector with a direction from upper right to lower left (-1,-1). Using the same concept for the second cluster we will end up with a vector of (-1,1).

(d) To decide the covariance matrices, we can figure out easily that the letter A has a sharpe direction. If we compared it with the letter like B or C, we need some curvature to draw this letters. This curvature or variety could be achieved by having a good variance in generated the arrows within a cluster. But in our case, we need the directions to be sharpe as much as possible. Thus we had choosen a very narrow distributions with small-element in diagonal covariance matrices.

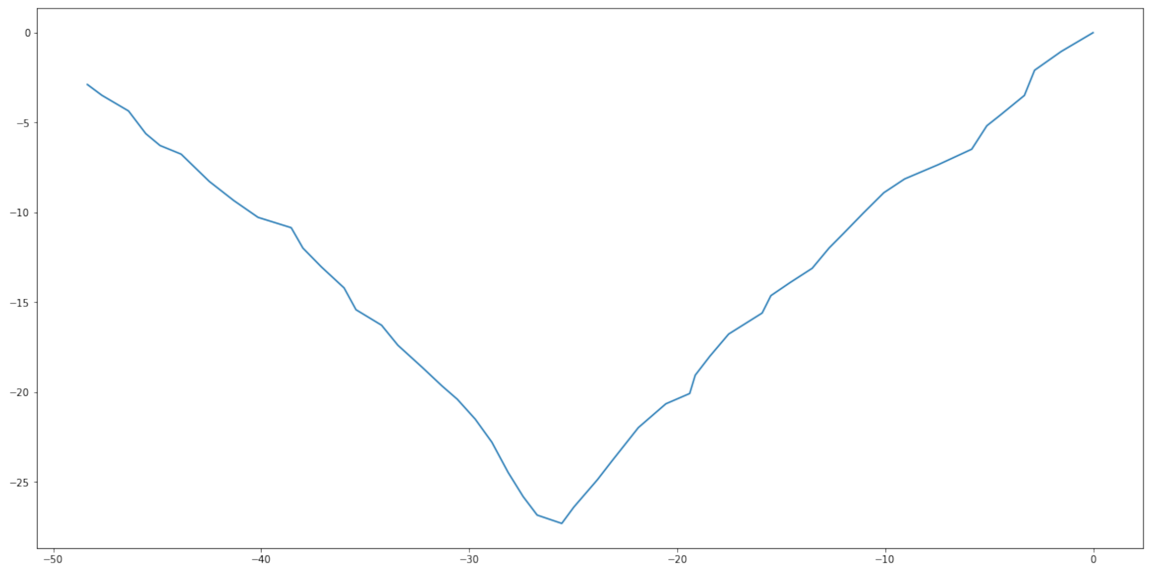

Figure 5: Letter A generated for 50 samples, using defined HMM

3. We have used hmmlearn.py to estimate a HMC model for the letter A, with 2 states in for the latent variables with Gaussian parameters for emission distributions:

$$\pi = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 0.872 & 0.128 \\ 0.001 & 0.999 \end{bmatrix}$$

$$\mu_{\mathbf{1}} = \begin{bmatrix} -0.56 \\ -1.01 \end{bmatrix}, \quad \mu_{\mathbf{2}} = \begin{bmatrix} -0.42 \\ 0.97 \end{bmatrix}$$

$$\mathbf{\Sigma_1} = \begin{bmatrix} 0.289 & 0.409 \\ 0.409 & 0.784 \end{bmatrix}, \quad \mathbf{\Sigma_2} = \begin{bmatrix} 0.269 & -0.407 \\ -0.407 & 1.160 \end{bmatrix}$$

We can notice that the value estimated are very close to the one we defined in question 2. Thus our comments on these value will be the same as our reasons in question 2.

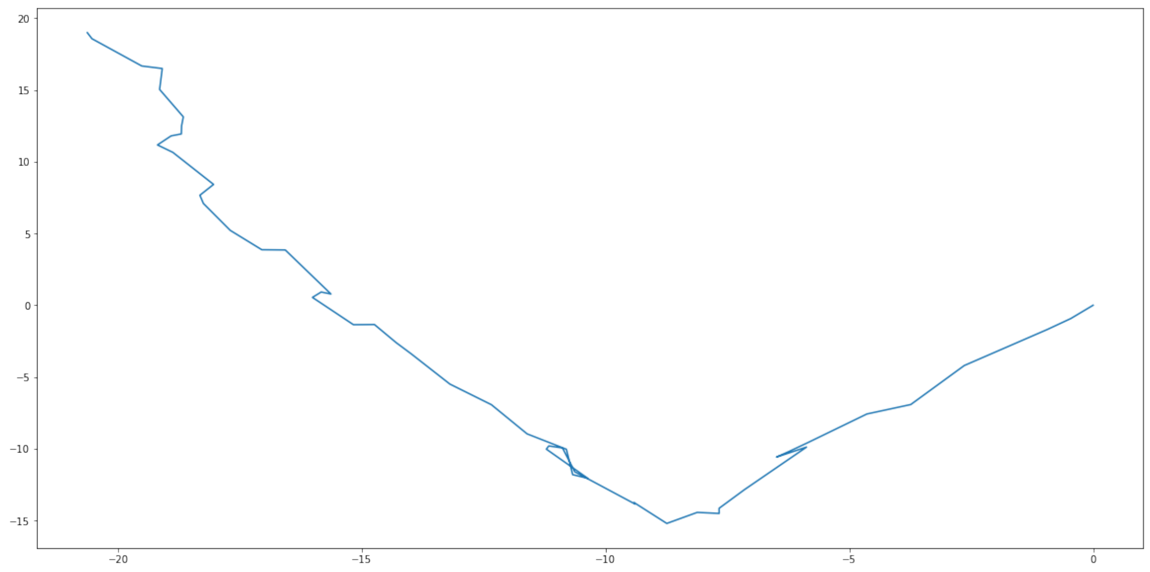

Figure 6: Letter A generated for 50 samples, using estimated HMM, with normalized trajectories

4. Similarly, in case of letter L, we estimated a HMC model with 2 states in for the latent variables (the same reason as letter A, we have to main directions: horizontally and vertically) with Gaussian parameters for emission distributions:

   $$\pi = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

   $$\mathbf{A} = \begin{bmatrix} 0.986 & 0.014 \\ 0.112 & 0.888 \end{bmatrix}$$

   $$\mu_{\mathbf{1}} = \begin{bmatrix} -1.004 \\ 0.032 \end{bmatrix}, \quad \mu_{\mathbf{2}} = \begin{bmatrix} 0.013 \\ 1.121 \end{bmatrix}$$

   $$\mathbf{\Sigma_1} = \begin{bmatrix} 1.046 & -0.035 \\ -0.035 & 0.033 \end{bmatrix}, \quad \mathbf{\Sigma_2} = \begin{bmatrix} 0.028 & 0.004 \\ 0.004 & 1.109 \end{bmatrix}$$

   The reasons behind the value of the parameters estimated could be similar to the one's for letter A:

   (a) We also start with one direction always.This interprets the 1-value vector.

   (b) Since the transition will be done one time from one direction to another, this is explain the nearly zero value of the second cluster line in the transition matrix. Also it explains the weak value in the first cluster to switch to the other direction.

   (c) the two mean vectors approaches to (-1,0) and (0,1) which also are the representative of such two directions (from right to left and from down to up).

   (d) Since we need sharpness also in the later L, like letter A, thus, the value of the covariance matrix are relatively small.

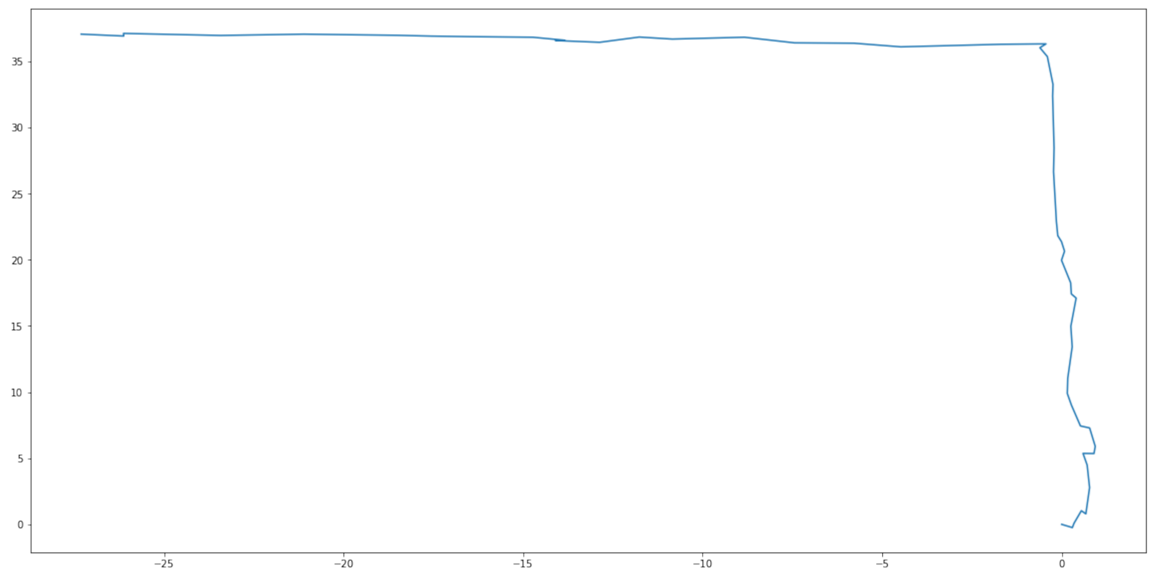

Figure 7: Letter L generated for 50 samples, using estimated HMM, with normalized trajectories

5. Figures 8 and 9 show the result of using Viterbi algorithm on A1.txt and L1.txt. We plot the unnormalized sequence, and the input for prediction was always unnormalized. The prediction remains consistent given that the data is unormalized.

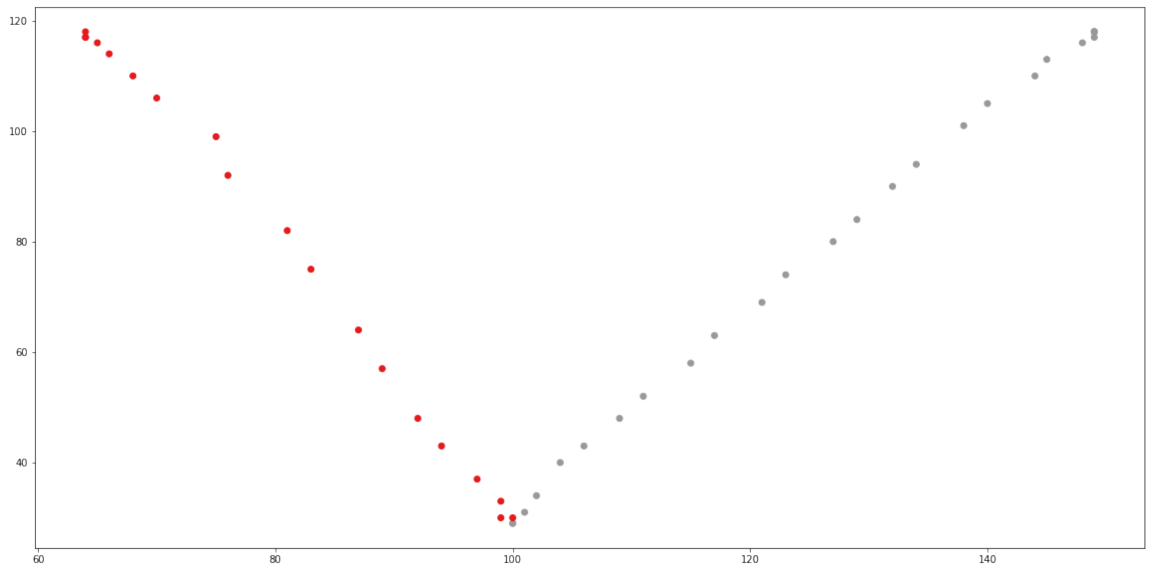

Figure 8: Letter A predicted for 50 samples, using Viterbi algorithm

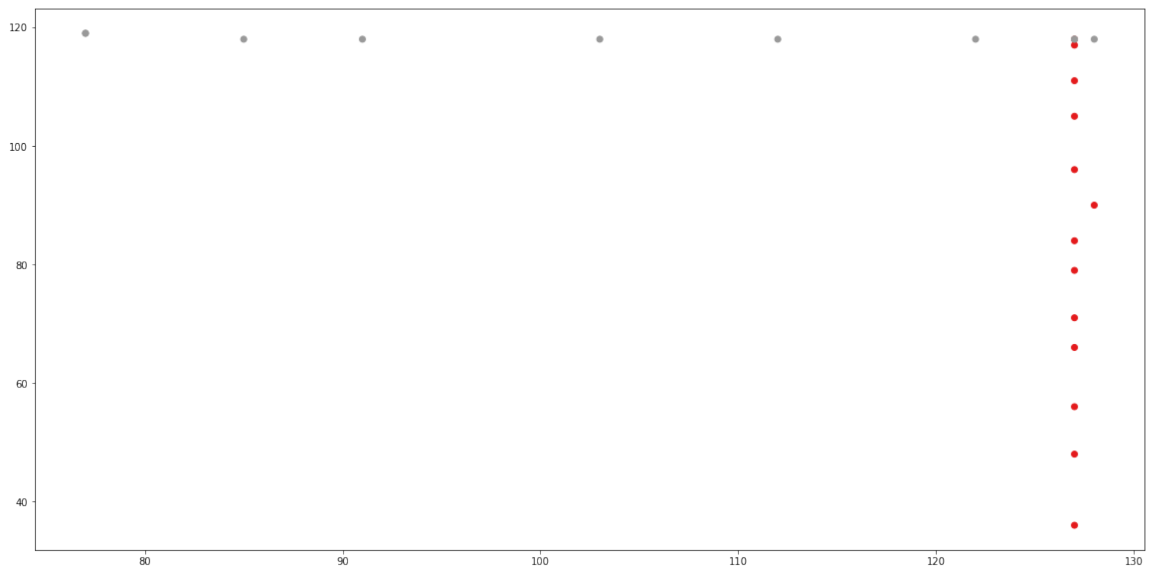

Figure 9: Letter L predicted for 50 samples, using Viterbi algorithm

6. Inspired form the previous lab of mixture models, we can use the same visual representation used in the previous lab to validate this assumption. The idea is simple : after we estimate a HMC model for a corresponding the sequence object, here is a letter A or L, We will predict the latent variable (cluster) for every cluster.

   Instead of visualize it in sequence like in figure 8, we will visualize it in unsequential manner like in previous lab (see figure 10). Depending of the visual results we can comment on the quality of choosing the number of latent variables and the distributions.

   Thus to summarize:

   - Estimate the model.
   - predict the clusters.
   - Plot the point without taking the temporal factor into consideration.
   - check the cluster density, intra-class variance and inter-class variance visually.

   By checking figure 10, we can confirm that for letter A, the assumption of 2 states was reasonable. Also we can note the circular form of the data, which can give us a motivation to claim that the von-mises distribution will be better to capture the clusters form giving the circular form of the data (or it could be the same result, since already the partitioning in the figure is acceptable). Another important note, we can comment similar to the previous lab that a univariate gaussian mixture will also give us a good clustering, since the high impact comes from the y coordinate more than the x coordinate. Or it will be better to take only into consideration the angle of the vectors.

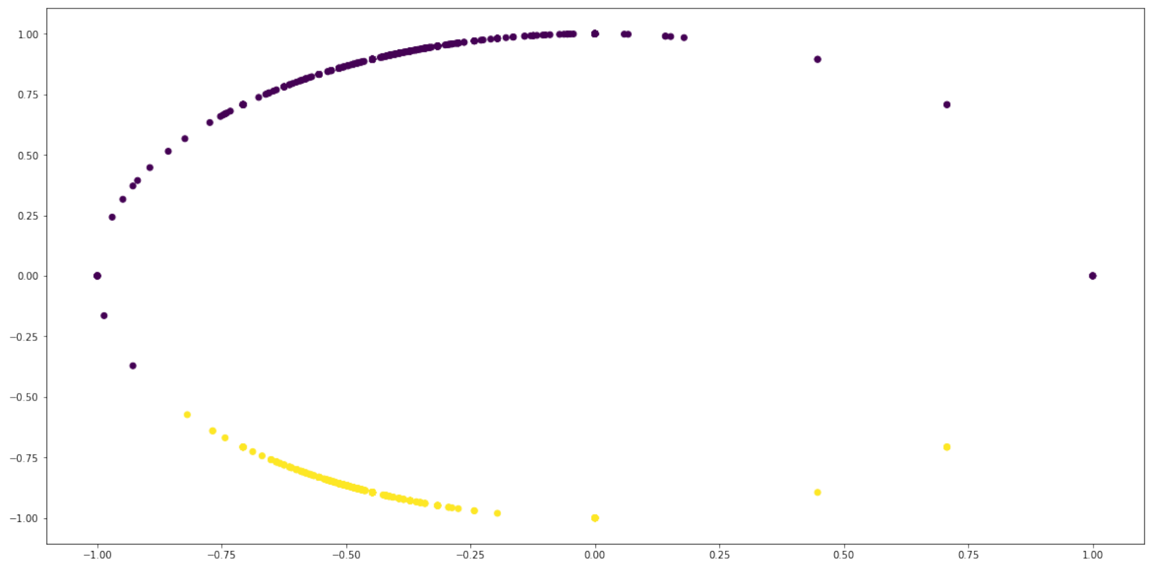

Figure 10: Letter A's strokes assigned to hidden states using Gaussian distribution

1.2 Mandatory additional questions

1. Concerning the Data, we first gather all the unistroke files for the letters A and L, shuffle them all, then divide their data into 5 folds. Iteratively, we choose one fold to be the test data, and the rest is used for training two 2-state HMCs (one for A and one for L).

   To do classification, we label A and L as 0 and 1 respectively. The decision condition is described in equation 1, where the log likelihood $\log p(x|z_l)$, is given by the function score in the HMM model.

$$C = \arg \max_{l \in \{0,1\}} \log p(x|z_l) \tag{1}$$

   For both emission distributions (Gaussian and Von mises), we got an accuracy of 100%.

1.3 Optional additional questions

1. The consistency of Number of states Estimator is verified when the size of the data m used increases, the estimated number of parameter $K^{'}$ approach to true number of states K.

$$\lim_{m \to \infty} K^{'} = K \tag{2}$$

   The number of states in hidden Markov models is a hyperparemeter whose value is set before the learning process begins, and is subject to manual or automatic tuning.

   To estimate this hyperparameter, the general approach is to fix it to a certain setting (value) each time, train a model on the data, then evaluate the performance of the model with a certain model selection criterion, then retain the value that maximizes the chosen criterion.

   The traditional way of performing hyperparameter optimization is grid search, which is simply an exhaustive searching through a manually specified subset of the hyperparameter space of the learning algorithm, which must be guided by a performance critirion, typically, it is set to Bayesian information criterion (BIC) which is defined as follows :

$$BIC = d \log(m) - 2 \log(L) \tag{3}$$

   Where, d is the number of parameters in the models, m is the size of the data and L is the log likelihood.

   The use of BIC in here ensures the consistency of the estimator because it penalizes complex models that are not able to generalize well.

2. In this question, we consider the model selection criterion to be the cross-validation of the log-likelihood.

   To evaluate the consistency of this estimator we suggest the following protocol :

   (a) We fix a Hidden Markov Chain that by fixing a distribution and its parameters, this HMC will serve as the ground truth. In here, we consider the Gaussian distributions

   (b) Generate sequences of increasing lengths from the previously defined HMC.

   (c) Use the estimator to estimate the number of states of the HMM.

   The estimator goes through different number of states k varying from 1 to K, and for each k:

   i. split the generated data to 5 folds

   ii. train k-state model on 4 of the 5 folds.

   iii. compute the sum of the log likelihoods of the test data.

$$\sum_{x \in X_{test}} \log P(x|k)$$

   (d) Retain the number of states $\hat{k}$ that has the highest

   (e) Check if the estimated number of states $\hat{k}$ converges to the ground truth k

   To test the consistency of the estimator in consideration, we used a 4-state HMC with the Gaussian distribution to generate sequences of fixed length, and ran a grid search from 1 to 9, we obtained the result shown in 11.

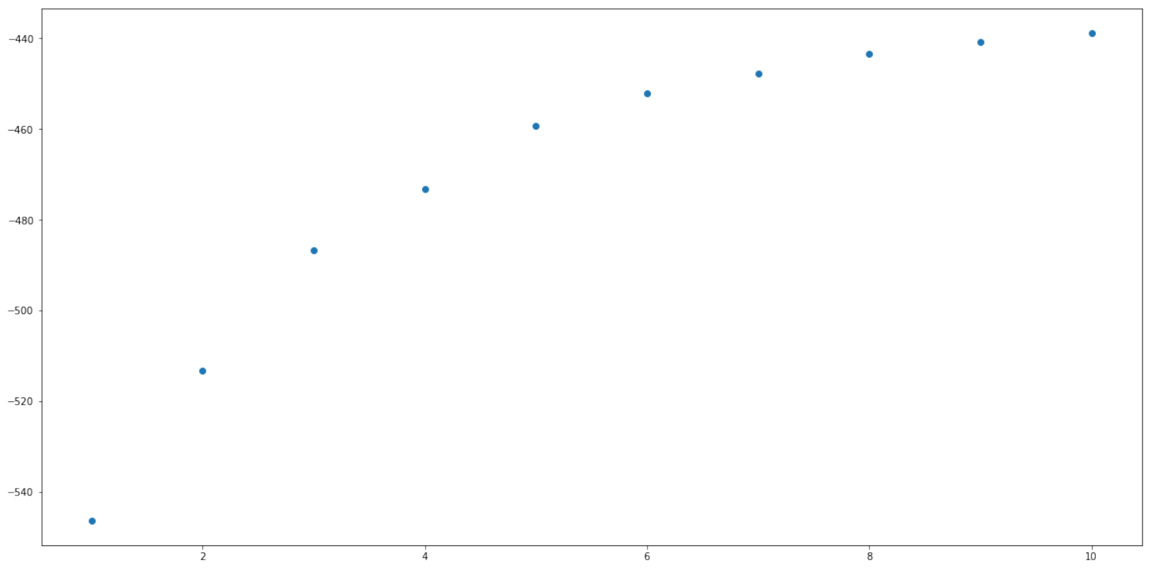

Figure 11: The cross-validation of the log-likelihood with respect to number of states of the HMM

Our results show that the estimator favors complex models with a high number of states. The estimated number of states given by this estimator is 10, different than the ground truth 4, this estimator is therefore not consistent. We suspect that the inconsistency of the estimator in consideration is due to the fact the selection criterion it uses does not penalize complex models that tend to overfit to the sample data.

Now let's compare the results of this estimator with the results of the estimator we proposed in the previous question which penalizes complex models (with a higher number of states), the results of the later are shown in 12

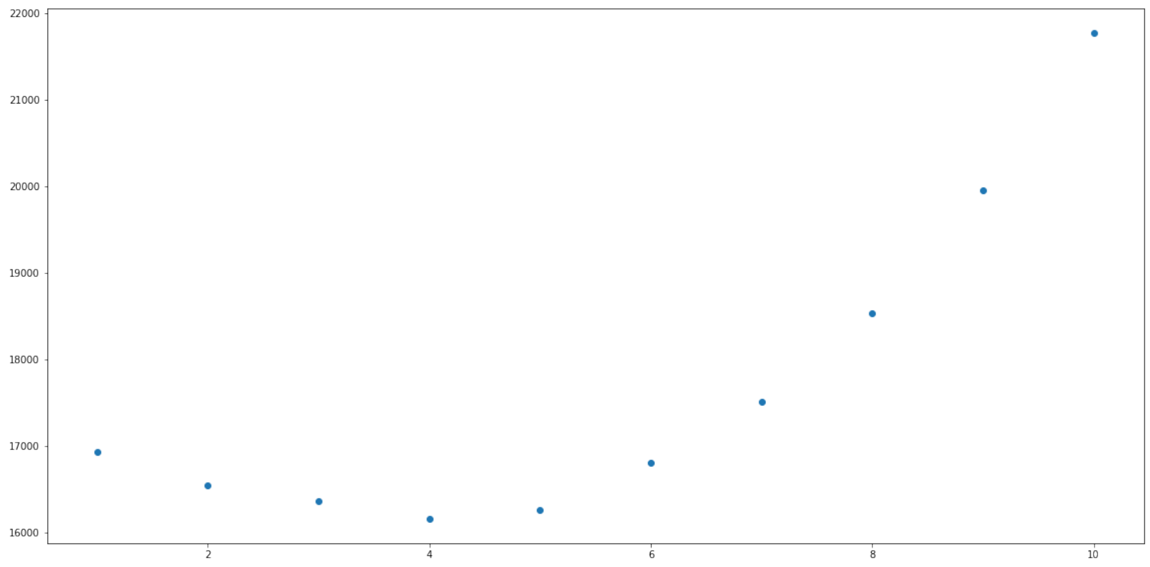

Figure 12: The BIC scores with respect to number of states of the HMM

We see that the estimated number of states (the one with the lowest BIC score) $\hat{k} = 4$ coincides with the ground truth $k = 4$ which proves that it is a consistent estimator.

3. We have used von Mises and by applying the procedures (from library msmBuilder library), we got a result, connected to the figures 13 and 14. The result is disappointed, as it partitioning is not accurate, as we assumed in question 6.

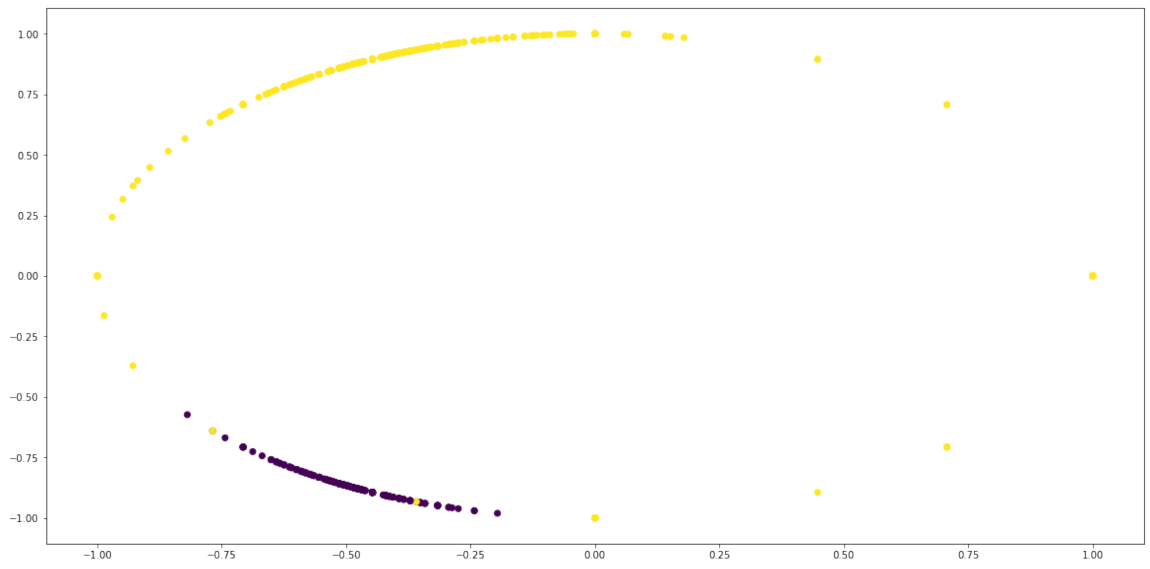

Figure 13: Letter A's strokes assigned to hidden states using von Mises distribution

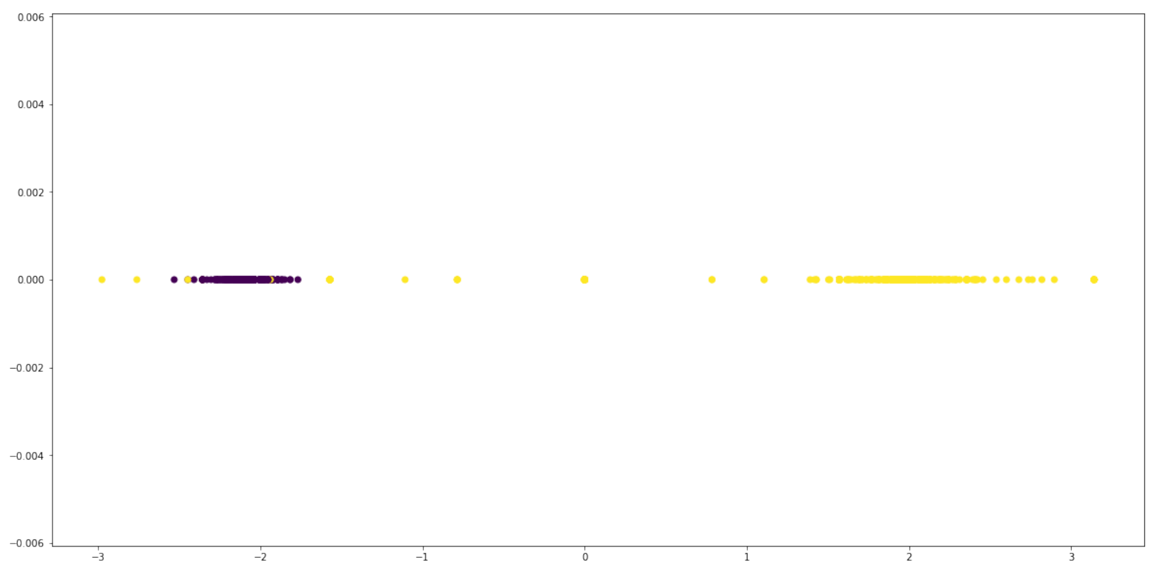

Figure 14: Letter A'strokes, represented in angular forum, assigned to hidden states using Gaussian distribution

4. For each of the the letters A, E, H, L, O and Q, we run the estimator proposed in question 1 to estimate the optimal number states of the hidden Markov model using Mises emission distributions, we got the results shown in 15

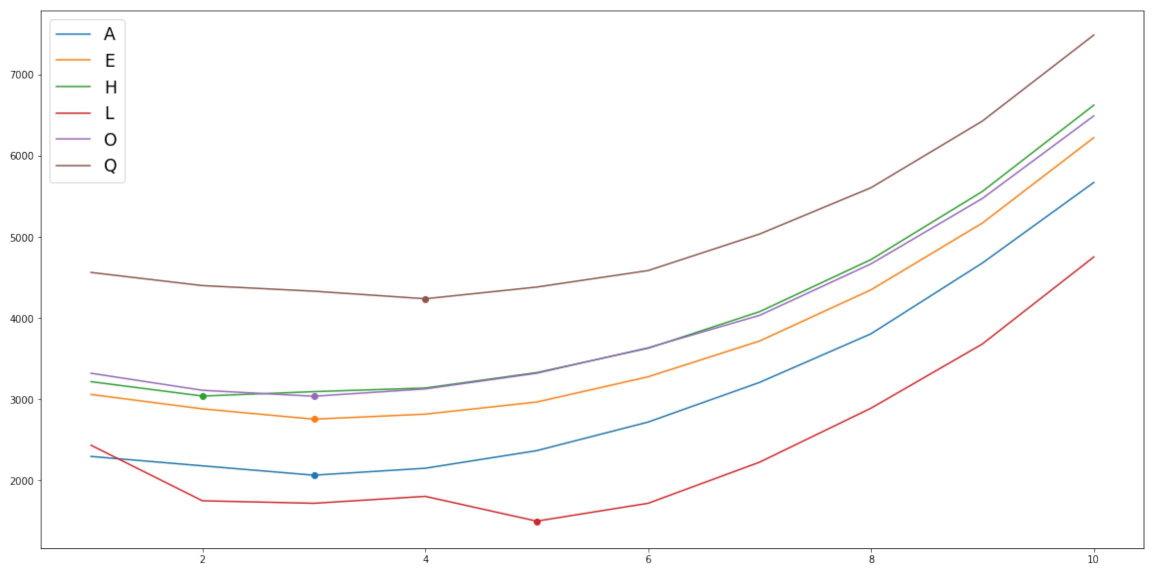

Figure 15: The BIC scores with respect to the number of states for each letter

The estimator we proposed suggested 3, 3, 2, 5, 3, and 4 states for the letter A, E, H, L, O, and Q respectively. The estimated number of states were close the results we primitively expected. There might be a slight error in expectation however because the number of sequences used in the estimation is finite.

5. Using the hyperparameters estimated in the previous question, we now build a multi-class classifier by combining 6 HMModels, where each model correspond to one of the letters A, E, H, L, O and Q.

   we use the following prediction rule to predict the most likely class $\hat{y}$ given the sequence x:

$$\hat{y} = \arg \max_{y \in \{A,E,H,L,O,Q\}} P(y|x) = \arg \max_{y \in \{A,E,H,L,O,Q\}} \frac{P(x|y)P(y)}{P(x)}$$

since the sequences are equally distributed among classes, the previous rule can be simplified to this form :

$$\hat{y} = \arg \max_{y \in \{A,E,H,L,O,Q\}} \log P(x|y)$$

$P(x|y)$ is simply the $P(x|M_y)$ where M_y is the HMM model of the letter y, the classification therefore comes down to finding the model that returns the biggest log likelihood of x.

By combining this approach and the k-fold cross validation, we built a model that gave the following confusion matrix :

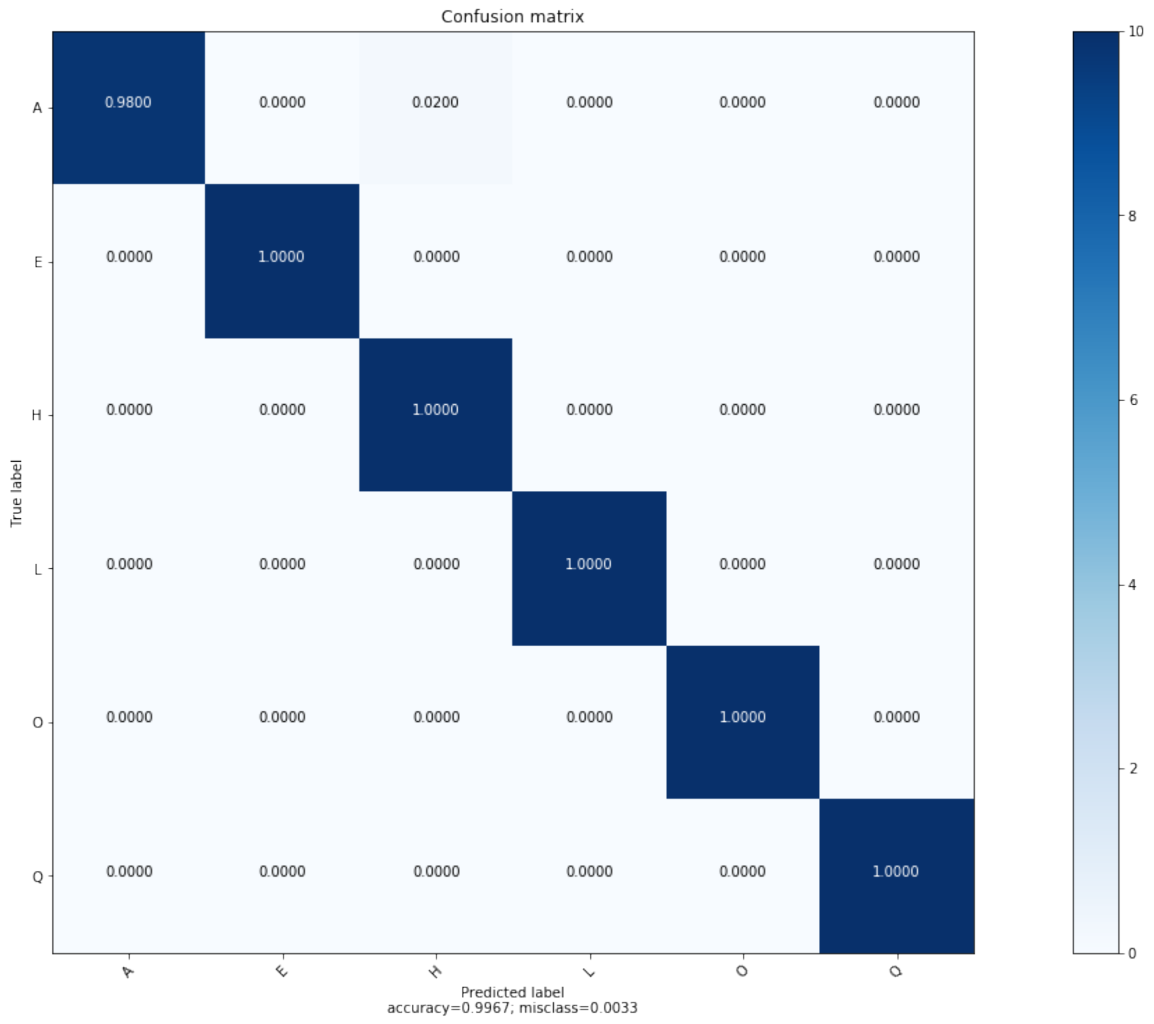

Figure 16: Confusion matrix of the 6-class classifier

As we can see by looking at the confusion matrix, the classifier did almost perfectly, with an accuracy of 99.9967%, with the only a tiny error in classifying the letter A, which we believe is due to the slightly inaccurate prediction of the number of states which itself is due to the limited number of sample used for estimation.

The parameters of the models that allowed us to get the mentioned accuracy are the following:

```
Letter A:
Covariance matrices:
        [[[ 8.59249461,  0.        ],
        [ 0.        , 26.53059006]],

       [[ 8.27496698,  0.        ],
        [ 0.        , 24.46697176]],

       [[ 4.73645494,  0.        ],
        [ 0.        , 16.02482168]]]
Means:
       [[ 5.20028617,  0.0399591 ],
       [ 5.21414896,  0.09487922],
       [ 2.87466511, -0.06329199]]

Letter E:
Covariance matrices:
        [[[23.2524531 ,  0.        ],
        [ 0.        ,  7.46130233]],

       [[29.26265435,  0.        ],
        [ 0.        , 10.29343534]],

       [[33.43310311,  0.        ],
        [ 0.        , 12.0089503 ]]]
Means:
        [[ 6.18823224, -0.70935444],
       [-4.7577105 , -4.33438416],
       [ 0.87985666, -3.4831973 ]]

Letter H:
Covariance matrices:
        [[[ 8.7799085 ,  0.        ],
        [ 0.        , 28.55783121]],

       [[ 6.81143923,  0.        ],
        [ 0.        , 26.53152426]]]
Means:
        [[ 2.27603485, -0.50683744],
       [ 1.56351884, -2.08144259]]

Letter L:
Covariance matrices:
        [[[ 6.77233333,  0.        ],
        [ 0.        ,  4.94073849]],

       [[ 1.16916127,  0.        ],
        [ 0.        ,  4.61666575]],

       [[ 1.39634453,  0.        ],
        [ 0.        ,  4.7730378 ]],

       [[11.4480246 ,  0.        ],
        [ 0.        ,  0.65336663]],

       [[ 5.23079057,  0.        ],
        [ 0.        ,  3.60451042]]]
Means:
        [[ 2.46361954, -4.10584735],
       [ 0.53304719, -3.97802405],
       [ 0.69920266, -3.78724849],
       [ 7.51194937, -0.60265259],
       [ 1.45183959, -2.74498399]]

Letter O:
Covariance matrices:
        [[[15.13124823,  0.        ],
         [ 0.        , 17.27839563]],

        [[19.39645303,  0.        ],
         [ 0.        , 14.98918955]],

        [[13.31556184,  0.        ],
         [ 0.        , 19.24125857]]]
Means:
        [[-0.89188002,  2.2237268 ],
        [ 0.62348211, -2.29838902],
        [ 3.33595281,  1.18338618]]

Letter Q:
Covariance matrices:
        [[[22.02721275,  0.        ],
         [ 0.        , 21.71259948]],

        [[23.58911293,  0.        ],
         [ 0.        , 17.12324371]],

        [[28.05105579,  0.        ],
         [ 0.        , 15.44195325]],

        [[23.43386459,  0.        ],
         [ 0.        , 17.38335119]]]
Means:
        [[ 0.03607371,  1.52498257],
        [ 2.52811677,  0.2978714 ],
        [-1.79319986, -1.92662995],
        [ 2.43415912,  0.18880749]]
```

References

[1] S. Calderara et al. (2011) *Mixtures of von Mises Distributions for People Trajectory Shape Analysis.*

[2] C. Bishop. (2006) *Pattern Recognition and Machine Learning.*