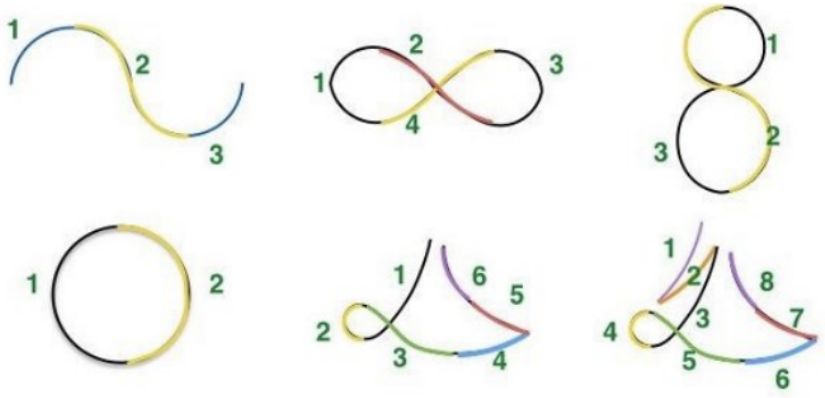
Sam Willenson (shw58)
ECE 5242

Project 2 Write-up

Write a project report including the following sections: Introduction, Problem Formulation, Technical Approach, Results and Discussion. • For each gesture model, you must include your training plot of your log-likelihood per epoch in your results. • Clearly presenting your approach in the report and having good algorithm performance are equally important.

Introduction

For this assignment, I was tasked with building a set of HMM models to predict/recognize different gestures. The data I was given was IMU sensor readings from accelerometers and gyroscopes, which was used to train my models. After training, I was able to classify unknown gestures using my algorithm. The gestures were arm motion gestures with 6 options: Wave, Infinity, Eight, Circle, Beat3, and Beat4 seen below.



Utilizing KMeans clustering, a forward/backward algorithm and Baum Welch, I was able to train my HMM models and give my top 3 predicted values with a given confidence level for unknown gestures.

Problem Statement

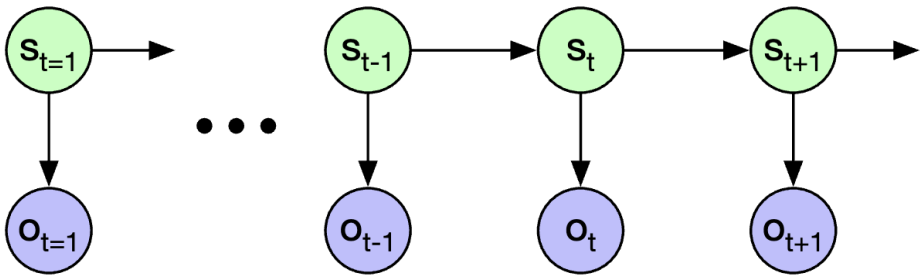
How do you build a model to predict unknown gestures, given a training set of data for these gestures?

Technical Approach

The main technical challenge for this project was building my HMM models for each gesture. Before the model can be built/trained however, the training data must be preprocessed. The first step of this process was to break up my training data into different arrays for each gesture. Once this was done, I took all of my data for all the gestures and put that into one array. I then implemented KMeans clustering (using sklearn for the KMeans) to fit this data into 60 clusters,

quantizing my data. I did not include the time column for this clustering as it was not needed. Once this clustering was done, I was able to use this KMeans model to assign clusters to each data segment from each gesture array I created.

After the clustering was done, I was able to move onto building my HMM models. Using the resources given to me (on the lecture slides and the Rabiner paper) I was able to build my model by implementing the following formulas to calculate my probabilities for transitioning states given a certain observation:



HMM flowchart for states and observations

Transition Probabilities	$A_{ij} = P(s_{t+1} = \mathcal{S}_i \mid s_t = \mathcal{S}_j)$
Observation (Emission) Probabilities	$B_{jk} = P(o_t = \mathcal{O}_k \mid s_t = \mathcal{S}_j)$
Initial State Distribution	$\pi_i = P(s_{t=1} = \mathcal{S}_i)$

The first step was to initialize A, the probability of transitioning from one hidden state to another. My number of hidden states was set to 10. Then initializing B, the probability of being at a hidden state for a given observation. As well as initializing pi, the probability of starting at a hidden state, which was given to be [1, 0, 0, …, 0] because the fact that the data starts at the beginning of the gesture was given.

$$\alpha_t(i) = P_\lambda(o_{t=1}, o_{t=2}, ..., o_t, s_t = i)$$

$$\beta_t(i) = P_\lambda(o_{t+1}, o_{t+2}, ..., o_T | s_t = i)$$

After A, B, and Pi, I then found and updated my values for alpha and beta. Alpha represents the forward probabilities, which are the probabilities of being in a certain state at a certain time, given all my previous observations. Beta represents my backward probabilities, which are the probabilities of observing the rest of the sequence from a certain state at a certain time. To find alpha, I implemented a forward algorithm for the HMM which recursively updates my alpha values. To find beta, I implemented a backward algorithm. To prevent under/overflows, I implemented scaling as defined in the Rabiner paper.

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)}. \qquad \hat{\beta}_t(i) = c_t \beta_t(i). \qquad \log [P(O|\lambda)] = -\sum_{t=1}^{T} \log c_t.$$

Next I implemented the E and M step of my Baum Welch algorithm. The E step gives me my gamma and xi matrices. The M step gives me my updated A and B matrices.

Data: $\mathcal{O}_{1:T}$
Definitions: K: num. possible states, M: num. possible measurements
Initialize: A, B, π

for $l = 1, \ldots, l_{max}$ **do**

Forward-Backward calculations:

$\alpha_1(i) \leftarrow \pi_i$

for $1 < t \leq T,\ 1 \leq i \leq K$ **do**

$$\alpha_t(i) \leftarrow \sum_{j=1}^{K} \alpha_{t-1}(j) A_{ij} B_i(\mathcal{O}_t)$$

end

$\beta_T(i) \leftarrow 1$

for $T > t \geq 1,\ 1 \leq i \leq K$ **do**

$$\beta_t(i) \leftarrow \sum_{j=1}^{K} A_{ji} B_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)$$

end

E-Step:

for $1 \leq i \leq K, 1 \leq t \leq T$ **do**

$$\gamma_t(i) \leftarrow \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{k} \alpha_t(j)\beta_t(j)}$$

end

for $1 \leq i \leq K, 1 \leq j \leq K, 1 \leq t < T$ **do**

$$\xi_t(i,j) \leftarrow \frac{\alpha_t(i) A_{ji} B_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{K} \sum_{j=1}^{k} \alpha_t(i) A_{ji} B_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)}$$

end

M-Step:

for $1 \leq i \leq K, 1 \leq j \leq K$ **do**

$$A_{ji} \leftarrow \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{i=1}^{K} \xi_t(i,j)}$$

end

for $1 \leq i \leq K, 1 \leq o \leq M$ **do**

$$B_i(o) \leftarrow \frac{\sum_{\{t \text{ s.t. } \mathcal{O}_t = o\}} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

end

end

Baum Welch Algorithm

Putting this all together for my Baum Welch Algorithm completed my iterative process. This involved updating alphas and betas from my current values for A and B as well as the most recent values for alpha and beta, and then updating my A and B values using my most recent alpha and beta matrices. By summing up the log of my scale matrix I was able to see my current likelihood for the gesture get smaller with each iteration.

After my training was complete, I had saved HMM models for each of my 6 gestures. The next and final step of my process was to test these models. There was no need for cross validation, as we were given 2 separate folders of data, one to train on and another that had one single matrix of IMU data for 1 complete gesture of each type. By testing my models on these single gestures, I was able to see if my predictions matched the correct gesture. By clustering this test data using my previously saved KMeans model, I quantize this test data. I then load my previously saved HMM models for each gesture to get my A and B values. Running this test data through one forward pass algorithm lets me get my scale values, and sum up the log of these scale values (negative) to get my likelihood for each gesture. Whichever of these models gives me the largest value (smallest negative number) is the winner for my prediction.

The main parameters for my training/testing were the number of training epochs, number of clusters, and number of hidden states. I set training epochs = 10, number of clusters = 60, and number of hidden states = 10, which was found through trial and error. I found my confidence level by taking the log-likelihood of the top prediction, and dividing that by the sum of the top 2 predictions. I then did 1 minus this number.

Results

My results on the test data set are seen below:

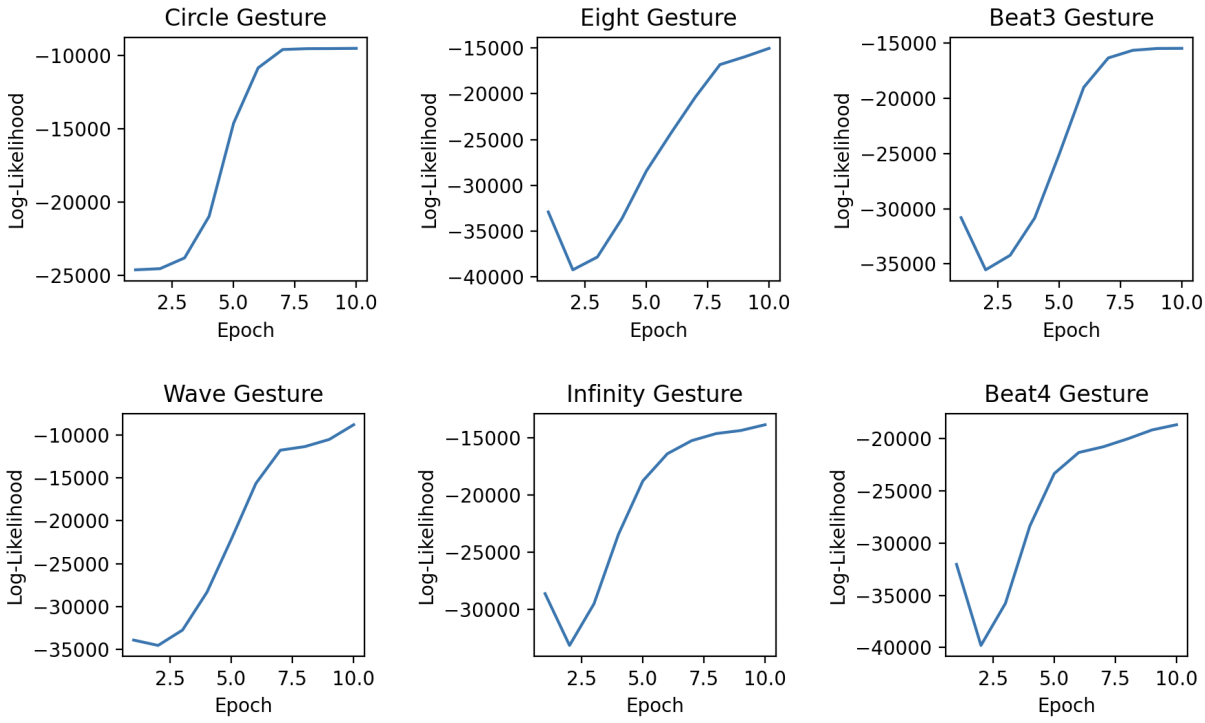
```
Predictions for test7.txt 1st: Eight , 2nd: Infinity , 3rd: Beat3 , with Confidence:  0.8807438505831734
Predictions for test6.txt 1st: Infinity , 2nd: Eight , 3rd: Beat3 , with Confidence:  0.9431284870299661
Predictions for test4.txt 1st: Beat3 , 2nd: Beat4 , 3rd: Wave , with Confidence:  0.5043746121447913
Predictions for test5.txt 1st: Circle , 2nd: Beat4 , 3rd: Beat3 , with Confidence:  0.9118311442696665
Predictions for test1.txt 1st: Wave , 2nd: Eight , 3rd: Infinity , with Confidence:  0.9240486650322048
Predictions for test2.txt 1st: Beat4 , 2nd: Beat3 , 3rd: Wave , with Confidence:  0.8756366115989606
Predictions for test3.txt 1st: Infinity , 2nd: Eight , 3rd: Beat3 , with Confidence:  0.8924166179068146
Predictions for test8.txt 1st: Beat4 , 2nd: Beat3 , 3rd: Wave , with Confidence:  0.8770499971016467
```

Here is a print out of my log-likelihoods for each gesture on each test file:

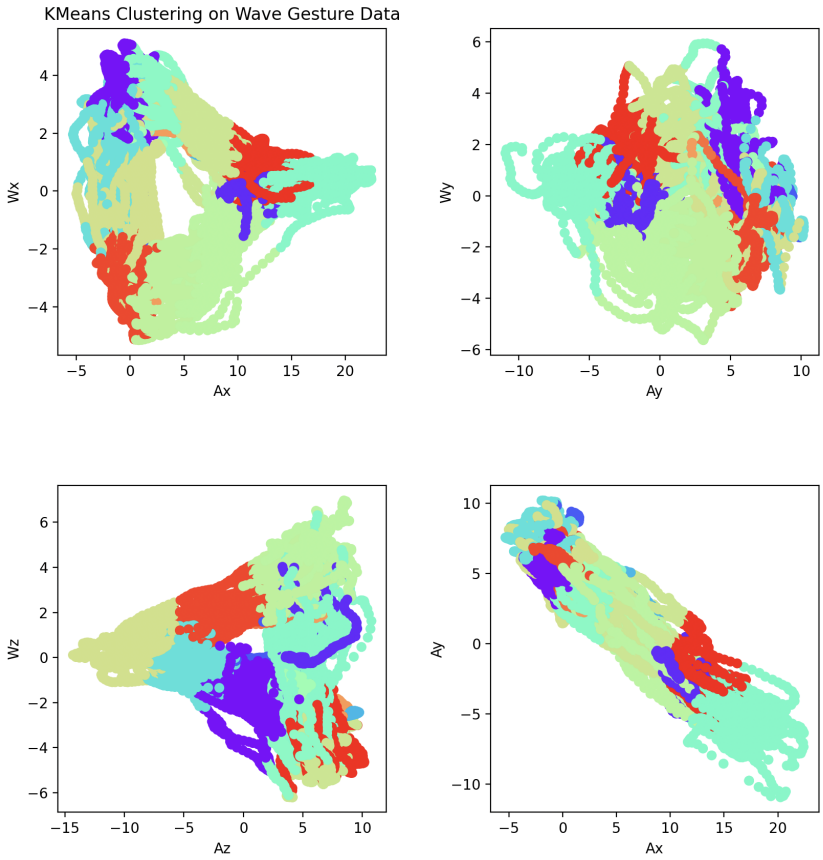
```
log likelihoods for:  test7.txt 1st: Eight -1191.6777967491457 , 2nd: Infinity -8800.912124831906 , 3rd: Beat3
-10835.308400014437
log likelihoods for:  test6.txt 1st: Infinity -723.5968097518999 , 2nd: Eight -11999.76453520069 , 3rd: Beat3 -
13448.37216289602
log likelihoods for:  test4.txt 1st: Beat3 -1015.8765403058483 , 2nd: Beat4 -1033.8097049892071 , 3rd: Wave -11
070.823232636674
log likelihoods for:  test5.txt 1st: Circle -636.0637422179451 , 2nd: Beat4 -6578.090699837654 , 3rd: Beat3 -70
49.4301810389925
log likelihoods for:  test1.txt 1st: Wave -542.8428276961174 , 2nd: Eight -6604.402548916334 , 3rd: Infinity -6
902.3123857091705
log likelihoods for:  test2.txt 1st: Beat4 -1292.8558190123736 , 2nd: Beat3 -9102.935383164038 , 3rd: Wave -126
05.486641869997
log likelihoods for:  test3.txt 1st: Infinity -1207.0592176291807 , 2nd: Eight -10012.69604702371 , 3rd: Beat3
-11363.016962292817
log likelihoods for:  test8.txt 1st: Beat4 -1220.3639238833598 , 2nd: Beat3 -8705.328594337021 , 3rd: Wave -119
31.126662054878
```

These results are fairly accurate, but my model was not very confident at predicting Beat3. The likelihood for Beat3 and Beat4 being the prediction for test4 were extremely similar. However, it was much more confident at predicting Beat4 as Beat4. Therefore, I would not be overly surprised if it turns out that test4.txt was in fact Beat4, but the fact that my model was not very sure on test4.txt and was more sure on test2.txt and test8.txt makes me think it is indeed a Beat3 gesture.

Here are plots of the likelihood as my HMM models were being trained:



Here is a visualization of my raw Wave gesture data, clustered by KMeans:



Discussion

Overall, my model performs fairly well. There is one bug in my code somewhere that makes my first log-likelihood calculation smaller than the second, however this does not seem to be a huge problem since the final likelihood value is much smaller than the initial value. Besides that, a main flaw in my algorithm is that it does not confidently predict Beat3. It predicts Beat4 with fairly high confidence, however the log-likelihood for Beat3 and Beat4 were fairly similar when I tested my model with a Beat3 gesture input. All of this aside, my model performs fairly accurate predictions. It is possible that with a more methodical approach (rather than guess and check) to calculating my number of hidden states and clusters, my model would perform even more accurately. However, as it stands I think my model can be deemed a success.