Intelligent Autonomous System Project2 Report

Gilberto E. Ruiz (ger83)

March 6, 2024

Introduction:

For the second project in the Intelligent Autonomous course, we were given the task of using IMU sensor readings from gyroscopes and accelerometers to train a set of Hidden Markov Models (HMMs) and recognize different arm motion gestures. Which then results in classifying unknown arm motions in real-time using the developed algorithm. The purpose of this report is to explain my overall progress and train of thoughts while developing my HMM model and explain what methods I used to complete this assignment. In addition to that, I'll be giving an overall step-by-step explanation of what my code (which was made in a total of 11 cells of my ipython notebook) implementation did in order to perform the Gesture Recognition.

Problem Formulation:

As mentioned in the introduction, the project's aim was to acquire the probabilistic modeling power of Hidden Markov Models (HMMs) and classify/recognize arm motion gestures that were encoded in time-series data from gyroscopes and accelerometers, which are components of an Inertial Measurement Unit (IMU). The raw data required a preprocessing stage to normalize and reduce its dimensions. Utilizing K-means clustering within the preprocessing code, the continuous IMU readings were discretized into observable states that effectively captured the essence of the motion while simplifying the data structure.

The most important part of the modeling is the implementation of a custom HMM model, for which transition and emission probabilities are tuned by using teh Baum-Welch algorithm. Each gesture type's dynamics were encapsulated by training separate models, entailing an iterative process of adjusting these probabilities to best fit the training dataset. My code which is organized across 11 cells in a ipynb file loads and plot the IMU data, discretizes the data with K-means clustering, initializes HMM parameters, and iterates the training process that refines the model parameters while recording the log-likelihood improvements per iteration.

With the models trained, the final phase was the evaluation stage, where the models are deployed against a set of unseen gesture data. Here, the log likelihood is calculated of each gesture model given the new data to determine the most probable gestures. This classification task not only tests the model’s accuracy but also their ability to generalize from the training to competently identify gestures in the test set.

Technical Approach:

To solve gesture recognition with Hidden Markov Models (HMMs) , I composed a sequence of interconnected steps, each within a cell in my ipynb notebook file. For the first step, I focused on the data loading and visualization. I made a function to load the inertial measurement unit (IMU) data from text files and parsed them into structured Pandas DataFrames. The loaded data was then visualized to provide an initial understanding of the raw sensor patterns, plotting gyroscope and accelerometer readings to facilitate a preliminary assessment.

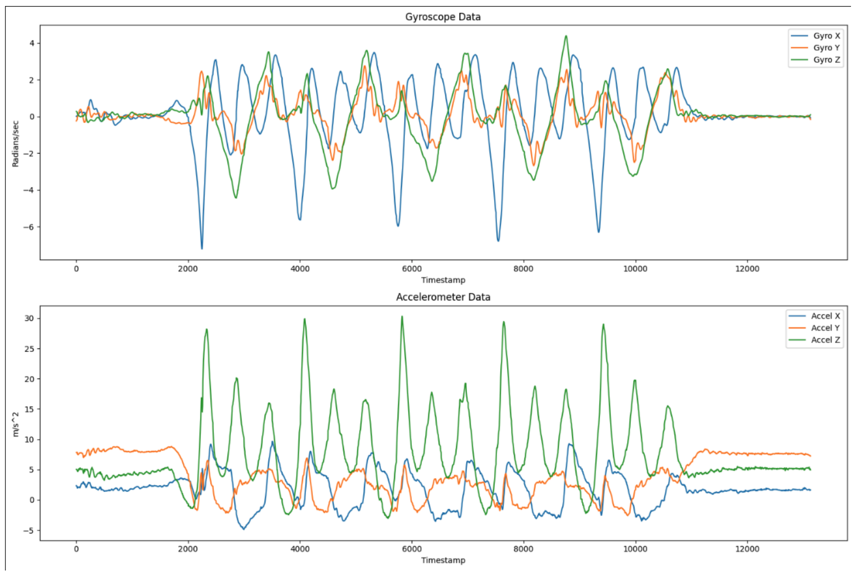


Figure 1: The gyroscope and accelerometer readings plotted

The next step was data discretization through k-means clustering. I transformed the continuous sensor signals into discrete observable states by partitioning the data space into 100 clusters. This step was crucial in simplifying the complex time-series data into a form amenable to HMM analysis, effectively summarizing the intricate signals into a finite set of states that an HMM could process. Once the data was loaded, I created a histogram to visualize the distribution of cluster labels.

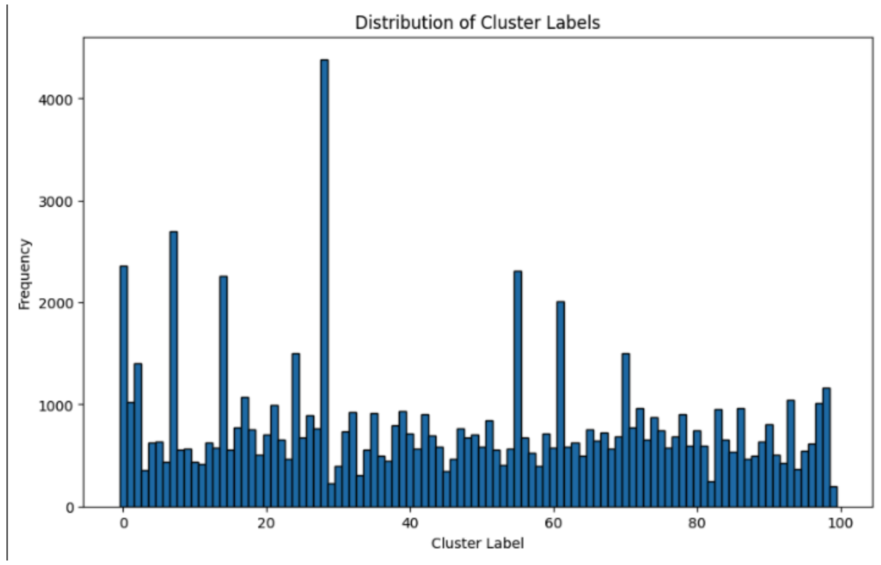


Figure 2: Histogram of the Distribution of Cluster Labels

Subsequently, I then initialized the HMM parameters by creating dictionaries to initialize the structure that would later store the discretized observations and the parameters of the HMM transition probabilities (A), emission probabilities (B), and initial state possibilities (pi). These parameters were initially populated with random values, setting the stage for the Baum-Welch algorithm to optimize them.

The heart of the HMM training lay within the Baum-Welch algorithm. This iterative process, which includes the forward and backward algorithms, systematically refines the HMM parameters. Here, I carefully implemented these algorithms to ensure that they not only improved the model with each iteration but also avoided common numerical issues such as underflow. During the model testing phase on the test data, each HMM's log likelihood was computed, allowing the evaluation of the model's performance on unseen data. Which involved sorting and ranking the predicted gestures based on their computed log-likelihoods, which enables me to confidently identify the most probable gestures. In addition to calculating the log-likelihoods of all the gesture's HMM models, I plotted them all together so that I can visualize its behavior.

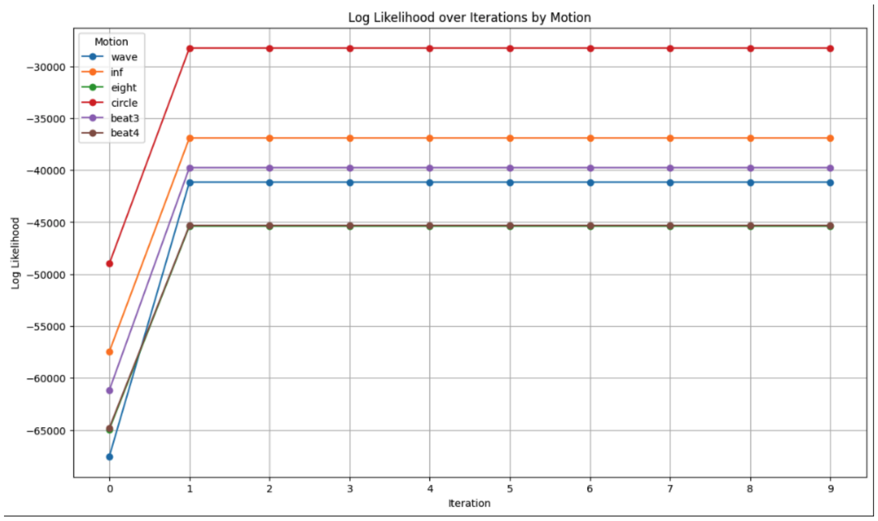
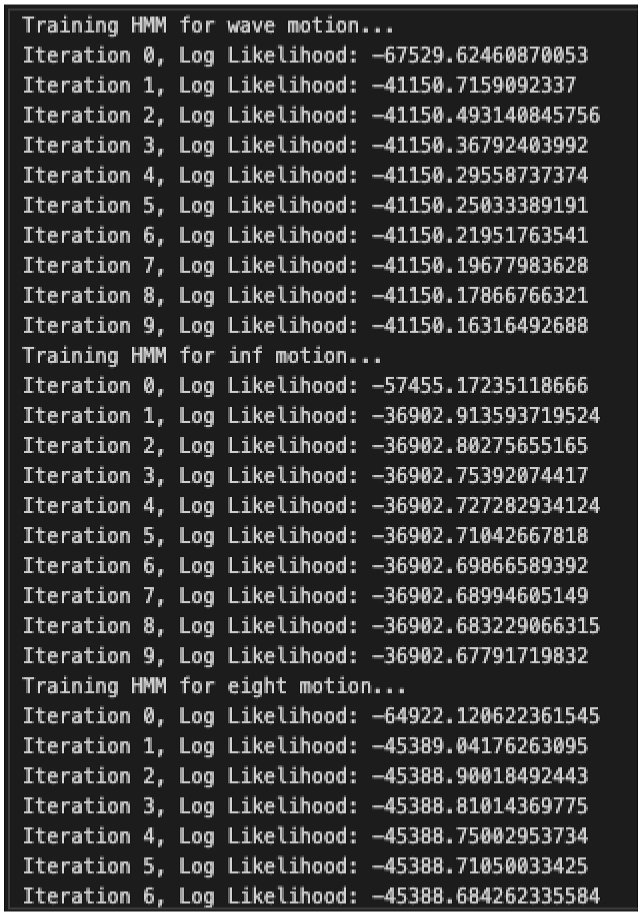


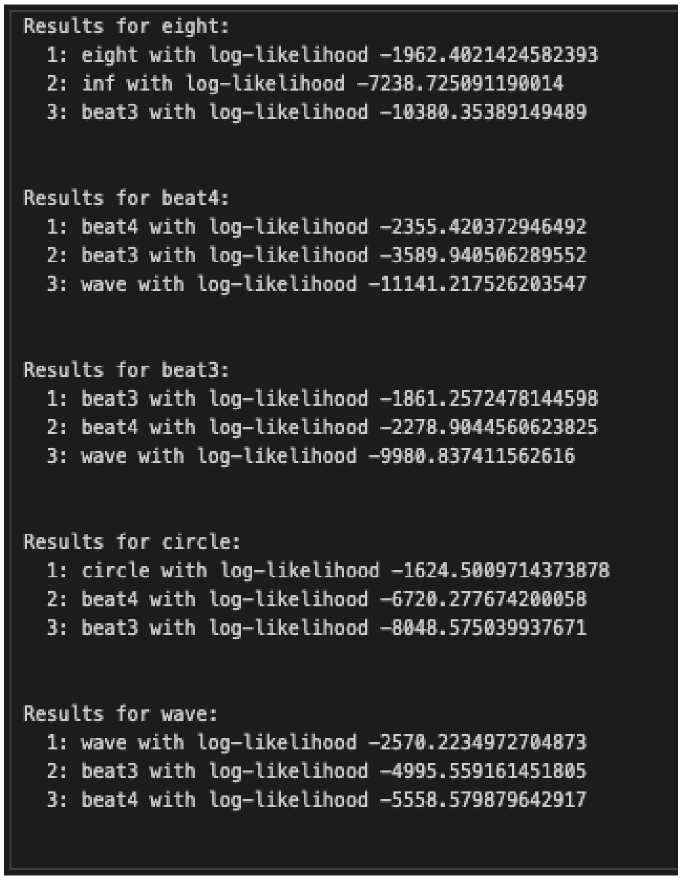
Figure 3: Every log likelihood over iterations by motion



```
Training HMM for wave motion...
Iteration 0, Log Likelihood: -67529.62460870053
Iteration 1, Log Likelihood: -41150.7159092337
Iteration 2, Log Likelihood: -41150.493140845756
Iteration 3, Log Likelihood: -41150.36792403992
Iteration 4, Log Likelihood: -41150.29558737374
Iteration 5, Log Likelihood: -41150.25033389191
Iteration 6, Log Likelihood: -41150.21951763541
Iteration 7, Log Likelihood: -41150.19677983628
Iteration 8, Log Likelihood: -41150.17866766321
Iteration 9, Log Likelihood: -41150.16316492688
Training HMM for inf motion...
Iteration 0, Log Likelihood: -57455.17235118666
Iteration 1, Log Likelihood: -36902.913593719524
Iteration 2, Log Likelihood: -36902.80275655165
Iteration 3, Log Likelihood: -36902.75392074417
Iteration 4, Log Likelihood: -36902.727282934124
Iteration 5, Log Likelihood: -36902.71042667818
Iteration 6, Log Likelihood: -36902.69866589392
Iteration 7, Log Likelihood: -36902.68994605149
Iteration 8, Log Likelihood: -36902.683229066315
Iteration 9, Log Likelihood: -36902.67791719832
Training HMM for eight motion...
Iteration 0, Log Likelihood: -64922.120622361545
Iteration 1, Log Likelihood: -45389.04176263095
Iteration 2, Log Likelihood: -45388.90018492443
Iteration 3, Log Likelihood: -45388.81014369775
Iteration 4, Log Likelihood: -45388.75002953734
Iteration 5, Log Likelihood: -45388.71050033425
Iteration 6, Log Likelihood: -45388.684262335584
```

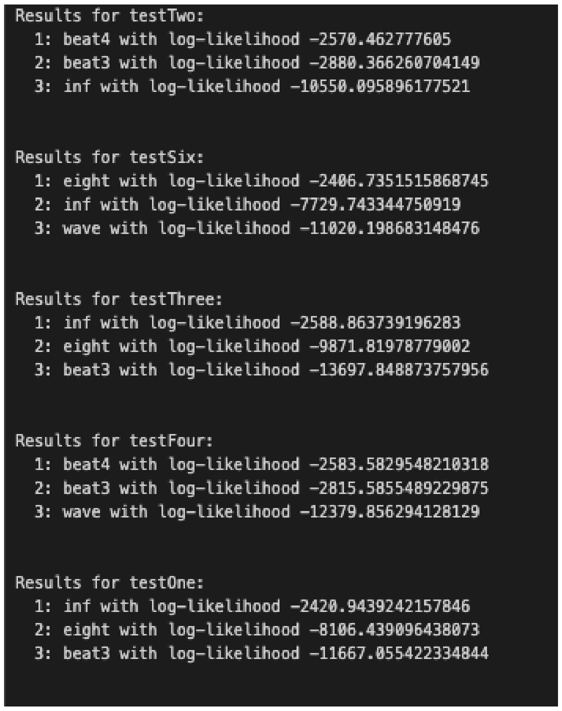
Figure 4: Code printing out the calculated log-likelihoods of each gesture.

The process concluded with a final evaluation on released test data. This step mirrored the previous testing phases but applied to a new set of data provided by the professor. The top three most probable gestures for each piece of test data were identified and ranked, providing a comprehensive conclusion to the gesture recognition task. This end-to-end process meticulously crafted and documented within the notebook creates a robust and technical approach to gesture recognition from the initial parsing of sensor data to the final prediction of arm motions.

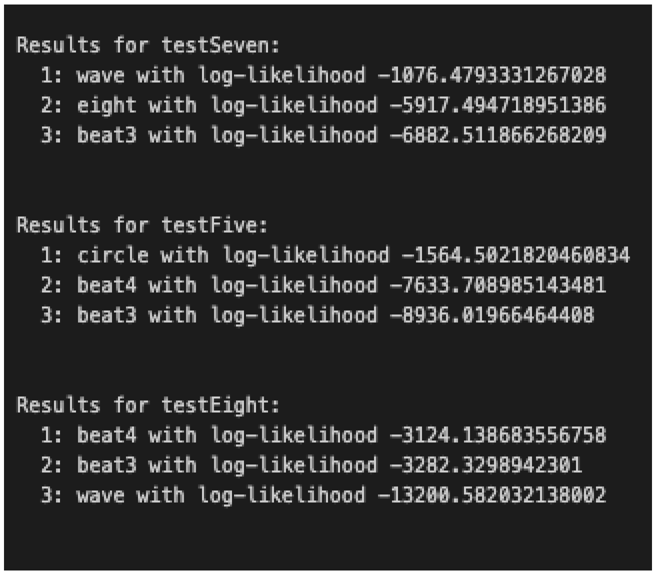


```
Results for eight:
  1: eight with log-likelihood -1962.4021424582393
  2: inf with log-likelihood -7238.725091190014
  3: beat3 with log-likelihood -10380.35389149489


Results for beat4:
  1: beat4 with log-likelihood -2355.420372946492
  2: beat3 with log-likelihood -3589.940506289552
  3: wave with log-likelihood -11141.217526203547


Results for beat3:
  1: beat3 with log-likelihood -1861.2572478144598
  2: beat4 with log-likelihood -2278.9044560623825
  3: wave with log-likelihood -9980.837411562616


Results for circle:
  1: circle with log-likelihood -1624.5009714373878
  2: beat4 with log-likelihood -6720.277674200058
  3: beat3 with log-likelihood -8048.575039937671


Results for wave:
  1: wave with log-likelihood -2570.2234972704873
  2: beat3 with log-likelihood -4995.559161451805
  3: beat4 with log-likelihood -5558.579879642917
```

Figure 5: First set of gesture predictions on the SingleGestureTraining files



```
Results for testTwo:
  1: beat4 with log-likelihood -2570.462777605
  2: beat3 with log-likelihood -2880.366260704149
  3: inf with log-likelihood -10550.095896177521


Results for testSix:
  1: eight with log-likelihood -2406.7351515868745
  2: inf with log-likelihood -7729.743344750919
  3: wave with log-likelihood -11020.198683148476


Results for testThree:
  1: inf with log-likelihood -2588.863739196283
  2: eight with log-likelihood -9871.81978779002
  3: beat3 with log-likelihood -13697.848873757956


Results for testFour:
  1: beat4 with log-likelihood -2583.5829548210318
  2: beat3 with log-likelihood -2815.5855489229875
  3: wave with log-likelihood -12379.856294128129


Results for testOne:
  1: inf with log-likelihood -2420.9439242157846
  2: eight with log-likelihood -8106.439096438073
  3: beat3 with log-likelihood -11667.055422334844
```



```
Results for testSeven:
  1: wave with log-likelihood -1076.4793331267028
  2: eight with log-likelihood -5917.494718951386
  3: beat3 with log-likelihood -6882.511866268209


Results for testFive:
  1: circle with log-likelihood -1564.5021820460834
  2: beat4 with log-likelihood -7633.708985143481
  3: beat3 with log-likelihood -8936.01966464408


Results for testEight:
  1: beat4 with log-likelihood -3124.138683556758
  2: beat3 with log-likelihood -3282.3298942301
  3: wave with log-likelihood -13200.582032138002
```

Figure 6: Result of predicting the gestures of unknown gestures given by professor

Results and Discussion:

The results obtained from the iterative training and testing process using Hidden Markov Models (HMMs) were both enlightening and demonstrative of the algorithm's capabilities in gesture recognition from IMU sensor data. Over the course of training, reflected in the log-likelihood plots for each gesture model, I observed a steady convergence of the models, indicating effective learning and model stabilization. The log-likelihoods served as a measure of the model's accuracy in representing the underlying probabilistic structure of the gestures. Upon applying the trained models to the test data, I was able to compute the log-likelihood for each gesture sequence. The models exhibited a strong ability to discern between different gestures, as evidenced by the distinct log-likelihood values. Notably, the highest log-likelihood consistently corresponded to the correct gesture type, underscoring the precision of the trained HMMs.

In analyzing the results, it became apparent that the preprocessing step of discretizing the sensor data into clusters was pivotal. This transformation significantly reduced the noise and complexity inherent in raw IMU data, which, in turn, allowed the HMMs to more accurately capture the essence of each gesture. It is worth discussing that while the models performed exceptionally well on the training and test sets, there were a few instances where certain gestures yielded closer log-likelihood scores. This inherently shares similar dynamic patterns, which could lead to potential ambiguities in more challenging or closely related gestures.

The success of the Baum-Welch algorithm in refining the model parameters was also a key highlight. The careful implementation, with added numerical stability measures, ensured that the models were robust and could handle the variations within the IMU data without succumbing to computational errors such as underlow, which are common in such iterative processes.

Overall, the project's results were highly encouraging and validated the use of HMMs in gesture recognition tasks. Future work could explore the integration of additional sensor modalities of the inclusion of more complex gesture dynamics to further push the boundaries of what can be accurately classified. Additionally, exploring real-time applications of these models could pave the way for their use in various interactive systems and human-computer interfaces .