ECE 5242 Intelligent Autonomous Systems

Project 2: Gesture Recognition

Alana Crognale arc232

Mar 8, 2020

Introduction

Gesture Recognition aims to use machine learning algorithms to be able to learn and detect different gestures based on a set of observations (in this case, we have IMU data corresponding to 3 degrees of accelerometer measurements and 3 degrees of gyroscope measurements). In this project, our goal is to train a Hidden Markov Model (HMM) for each gesture, such that we can classify unknown motions into these gestures. We do this by quantizing our data with K-Means, training an HMM model for each of the 6 gestures (wave, infinity, eight, circle, beat3, and beat4) using the Baum-Welch algorithm, and predicting the gesture with the highest likelihood of occurrence based on the HMM models and the unknown test IMU sensor readings to be classified.

Problem Formulation

How can we train a set of Hidden Markov Models to classify unknown arm motions in real-time given IMU sensor readings from gyroscopes and accelerometers?

Technical Approach

Vector Quantization:

Prior to creating our models, we first quantize our continuous-time data so that we can train a discrete HMM rather than a continuous HMM. This is because continuous HMMs require Gaussian-described observation distributions, while discrete HMMs associate continuous feature vectors with discrete values, a much more straightforward model in this case. In order to quantize our data, we perform K-means to cluster our training data (multiple repetitions of each of the 6 gestures). A K value of 50 clusters was initially chosen to correspond to the number of observations each data sample can be mapped to, and due to generally positive performance with this metric as well as run-time constraints, other values of K were not used.

Training HMM:

For each of the 6 gestures, we found parameters for and trained a discrete, ergodic HMM with 10 hidden states and 50 observations symbols corresponding to the 50 clusters, which similarly due to positive performance and time constraints was not altered.

For each gesture's model, we perform the following:

> First, we randomly initialize the transition matrix A, observation matrix B, and initial distribution vector pi. Experimentally, random initialization performs well for this type of task, as A, B, and pi will be recursively updated – these matrices can be initialized in many different ways; however, it is important to note that the rows of each matrix should be normalized and non-uniform. Next, we want to find the model parameters (A,B, and pi) such that the probability of the observation sequence is maximized. To do this, we use the recursive forward algorithm to

solve for alpha, where alpha(i) is the probability of the observation sequence up to time t, given that we are in state i at time t. If not scaled properly, we end up with underflow issues which lead to NaN values and consequently an improper model/classifier. To rectify this, we keep track of a vector of constants to scale alpha, such that we can divide alpha(i) by a constant c(i) to increase its value so that it can be appropriately stored. We similarly use the backward algorithm to solve for, and scale, beta. Note throughout this process that, if any entry in our observation matrix B is 0, we will get NaN values. To remedy this and to ensure that we have an ergodic model, we may add very small nonzero values on the order of 10^-10 to entries of the B matrix which will effectively map these entries to close to 0 probability. These alpha and beta values are initially calculated on our randomized parameters, so we compute the log likelihood based on this model and continue to iterate, updating alpha and beta based on each new A, B, and pi matrix, until the log likelihood stops increasing, i.e. until the probability of the observation sequence given the model parameters is maximized.

Classifying new data:

Once each model's parameters are learned and optimized, we can now find the model parameters to maximize the probability of the observation sequence, i.e. find which of the 6 HMM models corresponds best to an unknown set of observation sequences. After loading in a new dataset of IMU data, we first quantize this based on our previously implemented K-means, assigning each datapoint to its closest cluster. We then calculate the log likelihood for each HMM model given the observations sequence via a similar method as before, recursively computing alpha and beta using the HMM parameters and the new set of observations.

Results and Discussion

Below are each of the 8 test text files' log likelihood for each gesture, as well as a table summarizing the top 3 gesture predictions per text file:

	Top Prediction	**Second Prediction**	**Third Prediction**
test1.txt	Wave	Beat3	Eight
test2.txt	Beat4	Beat3	Infinity
test3.txt	Infinity	Eight	Beat3
test4.txt	Beat4	Beat3	Infinity
test5.txt	Circle	Beat4	Beat3

test6.txt	Infinity	Eight	Beat4
test7.txt	Eight	Infinity	Wave
test8.txt	Beat4	Beat3	Infinity

Fig 1. Top Predictions

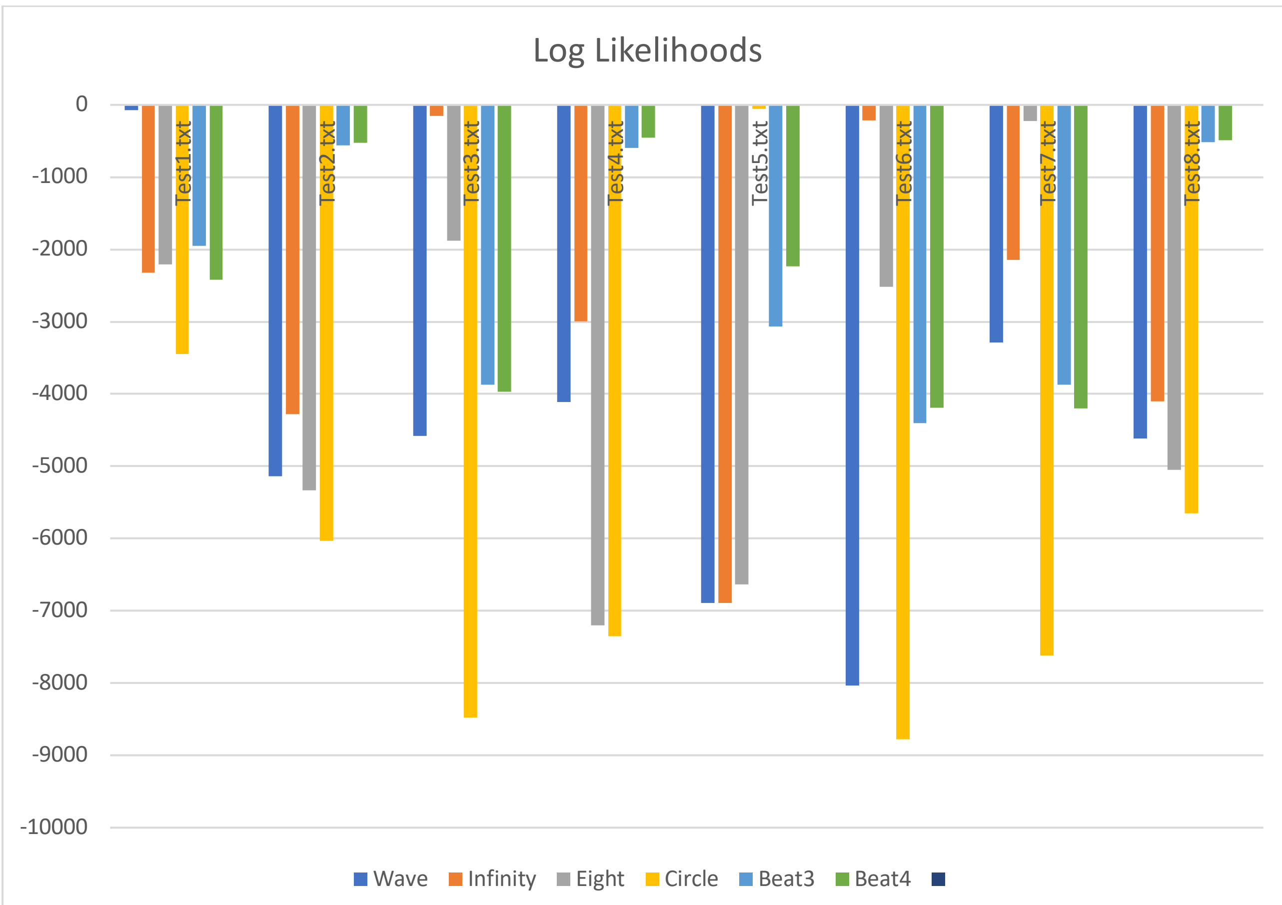

Fig 2. Summary of results

```
wave log probability:
-75.45046516201175
inf log probability:
-2325.4339406027025
eight log probability:
-2206.3038555087433
circle log probability:
-3449.6612848509576
beat3 log probability:
-1945.9316636817975
beat4 log probability:
-2419.500951407214
```

Fig 3. test1.txt log probabilities

```
wave log probability:
-5139.7319919170895
inf log probability:
-4279.884450533603
eight log probability:
-5334.756251762353
circle log probability:
-6035.470776306208
beat3 log probability:
-557.3729967629981
beat4 log probability:
-527.2395464142404
```

Fig 4. test2.txt log probabilities

```
wave log probability:
-4582.100654692628
inf log probability:
-154.21354972513095
eight log probability:
-1876.1514041485575
circle log probability:
-8479.142792612382
beat3 log probability:
-3868.983154279277
beat4 log probability:
-3969.709462433551
```

Fig 5. test3.txt log probabilities

```
wave log probability:
-4111.099654026928
inf log probability:
-2994.15603938315
eight log probability:
-7201.031378692309
circle log probability:
-7349.537073230463
beat3 log probability:
-592.8473088558658
beat4 log probability:
-453.0879131942843
```

Fig 6. test4.txt log probabilities

```
wave log probability:
-6894.088047662011
inf log probability:
-6889.999683072583
eight log probability:
-6636.58527105243
circle log probability:
-56.17888872616462
beat3 log probability:
-3070.203389191949
beat4 log probability:
-2232.321321060954
```

Fig 7. test5.txt log probabilities

```
wave log probability:
-8033.260524943895
inf log probability:
-212.23228519618658
eight log probability:
-2513.717116121179
circle log probability:
-8779.142792612382
beat3 log probability:
-4404.586894009786
beat4 log probability:
-4192.177722073795
```

Fig 8. test6.txt log probabilities

```
wave log probability:
-3289.0963378875963
inf log probability:
-2145.8243770367953
eight log probability:
-224.70031070070846
circle log probability:
-7619.142792612381
beat3 log probability:
-3870.362982581306
beat4 log probability:
-4195.013738577907
```

Fig 9. test7.txt log probabilities

```
wave log probability:
-4615.091818691269
inf log probability:
-4103.75089425945
eight log probability:
-5052.2453517770955
circle log probability:
-5653.870231096385
beat3 log probability:
-511.4529814861765
beat4 log probability:
-490.78207197387184
```

Fig 10. test8.txt log probabilities

We can see that in each of the above test datasets, there are only one or two gestures which have significantly higher log probabilities than the rest, signifying that the gestures are classified with fairly strong certainty. In particular, the wave, infinity, circle, and eight gestures all seem to be able to be distinctly classified, while the beat3 and beat4 gestures often have very similar log probabilities. These results makes sense since beat3 and beat4 are such similar gestures. We can see other patterns of expected similar probabilities, such as the infinity and eight gestures typically following one another. In this context, we already know the gestures and are using the observed sequences to classify; however, we can see from the log probability results how insightful it can also be to use HMMs to help us learn more about patterns, similarities, and differences in our states.