Will Whitney

Multimodal Signal Fusion Using HMMs

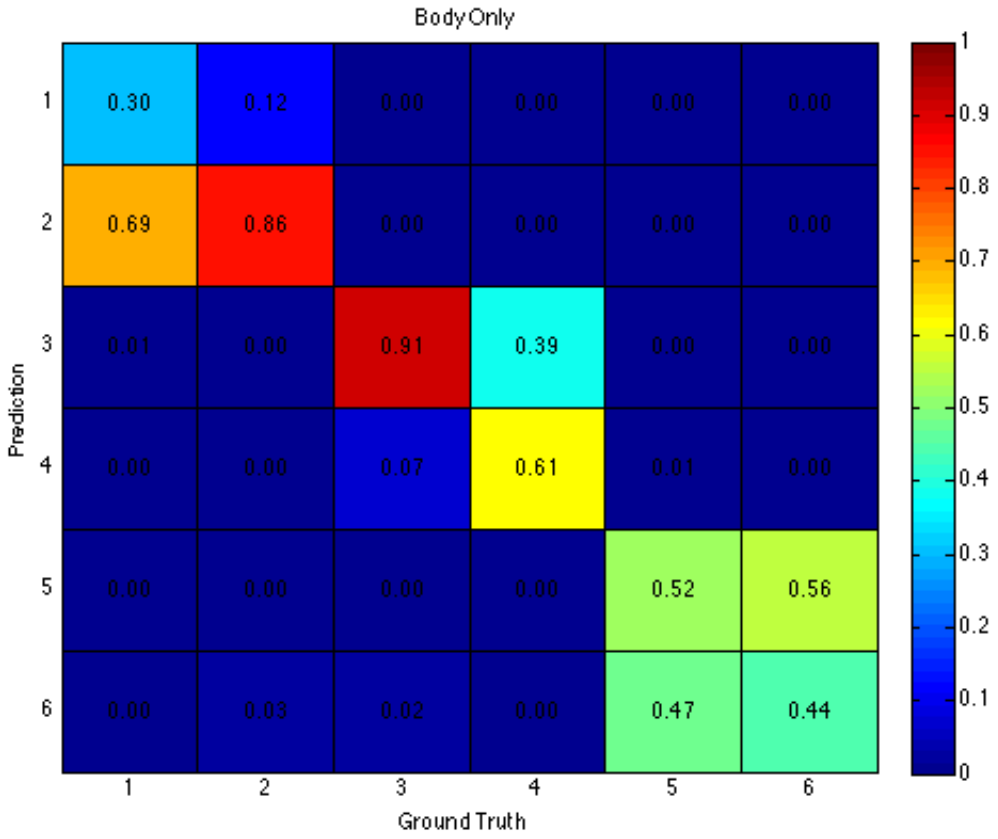
Question 1: From the experiments using body features (Part 1a) and hand features (Part 1b), what are the best classification accuracies you obtained? What parameter values did you validate and what was your strategy for finding the best parameter value?

Using body features, I obtained a classification accuracy of 0.606250 with parameters `H=12, G=1`. I used the grid search method given in `run.m` to select this parameters.

Using hand features, the maximum classification accuracy was 0.642708, with `H=8, G=2`. This result was also obtained using the given grid search methodology.

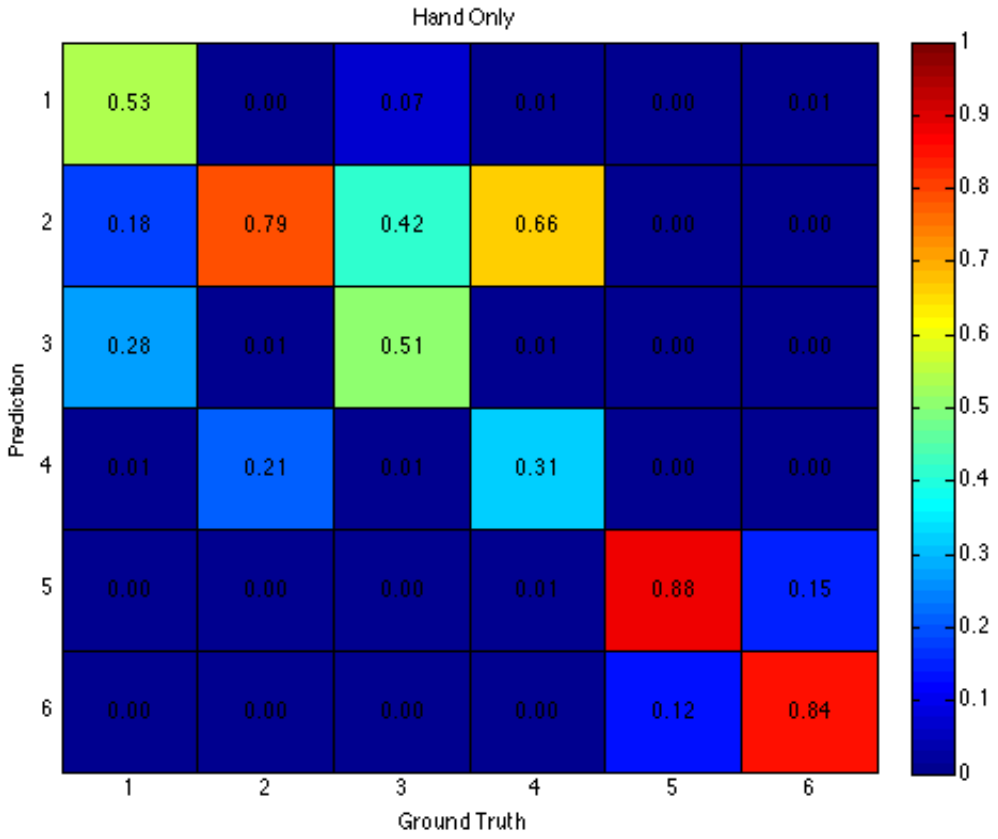
For both of these tests, all combinations of the parameters `H = {8, 4, 12}` and `G = {1, 3}` were tested.

Question 2: For each experiment, the run.m script will automatically generate a confusion matrix for the best performing model. Submit and interpret the two confusion matrices you obtained: For each modality explain which pairs of gestures are confused the most and *why* you think they were. See the gesture pairs in Figure 1 and use them in explaining the *why*.



For the body-only interpretation, gesture 1 was the most misinterpreted, as almost 70% of the time it was predicted as gesture 2. Gestures 5 and 6 were also thoroughly confused, with gesture 6 being interpreted more than half of the time as gesture 5. Gesture 5 itself was only barely more than half the time correctly interpreted. While gestures 3 and 4 were sometimes confused with each other, they were more distinguishable than the other confused pairs.z

From examining the diagrams, these confused pairs of gestures make perfect sense. These pairs have the same arm movements, and differ only in the hand position used for the gesture. The fact that gestures 3 and 4 are usually distinguishable is actually somewhat surprising, and probably reflects a difference in the arm movements caused by the wrist orientation change. Perhaps the signaler does not bring his hands as close together when the thumbs are pointing inward, towards each other.



The hand-only predictions exhibit some confusion between gestures 2 and 3, 1 and 3, and 2 and 4. This confusion was asymmetrical in the cases of 1 <--> 3 and 2 <--> 3, in that gesture 1 was misread as gesture 3, but 3 was not misread as 1. Similarly, 3 was read as 2, but not vice versa. Throughout gestures 1-4, likelihood of erroneous interpretation as gesture 2 was very high, with gesture 1 having the lowest (0.18) chance of such misreading.

Confusion between gestures 4 and 2, and gestures 3 and 1, makes perfect sense; these involve the exact same right hand orientation and pose on the features included in our data. The confusion between 2 and 3, however, is somewhat more perplexing, as the only features they (should) share are right hand palm state. Gesture 2 is thumb down, gesture 3 is thumb up, and gesture 2 only uses the right hand while gesture 3 uses both.

Question 3: Follow the steps in Part 2a. What is the best classification accuracy you obtained using early fusion HMM? What parameter values did you try and what was your strategy for finding the best

parameter value?

The early fusion HMM achieved an accuracy of 0.872917 with the parameters `H = 8` and `G = 1`. I found this value using the grid search method implemented in `run.m`, which tested `H = {8, 4, 12}` and `G = {1, 3}`.

Question 4: Implement testLateHMM.m to perform late fusion, and explain your implementation with pseudocode in your writeup. Note that you must follow the type signature of the function, as the function's input and output parameters are used in the function experiment_late_hmm.m.

```
Given:
- the data, `seqs`
- ground truth `labels`
- the body and hand HMMs as `hmm{1, 2}`
- the `featureMap` of which features go with which HMMs
- the `weightsMV`, a list of weightings to try

Calculate the log-likelihoods for each option on each gesture for just t
Calculate the log-likelihoods for each option on each gesture for just t

For each weighting in `weightsMV`:
    For each sample in `seqs`:
        Compute the weighted average of the likelihoods given by body- a
        Select the most likely interpretation and store it in `stat`

    Determine the percentage of the highest-probability guesses that mat
```

Question 5: Follow the steps in Part 2b. What is the best classification accuracy you can get using late fusion HMM? What parameter values did you validate and what was your strategy for finding the best parameter value? Which weight value tends to give you the best performance in terms of the classification accuracy? Why do you think that weight value give the best result?

I obtained a classification accuracy of 0.846875 with parameters `H=[12 8], G=[3 2], W=[0.50 0.50]`. I used the grid search method given in `run.m` to select this parameters.

A 50/50 weighting seems to give me the best performance here. I would theorize this is because the two independent classifiers have very similar accuracy rates, and so neither one is strongly preferred over the other. If this weighting had been very unbalanced, it would have indicated that one medium actually provided far better information than the other.

Question 6: Describe the differences between early and late fusion algorithms in terms of the underlying assumptions, how the classifiers are trained and then used to test new samples.

In an early fusion algorithm, there's a basic assumption that the data are in fact fuse-able in the raw; that is, they are aligned in time and in information carried. For an early fusion strategy to be successful, the sets of data need to be very similar to one another. If there is a time offset or uncertainty between them, or a mismatch in temporal correlation between the two modes, early fusion will provide no gains. Furthermore, early fusion works best when the two modalities have a fixed relative reliability; if one modality is sometimes effective and other times not, early fusion doesn't provide the flexibility to cope intelligently with the shifts.

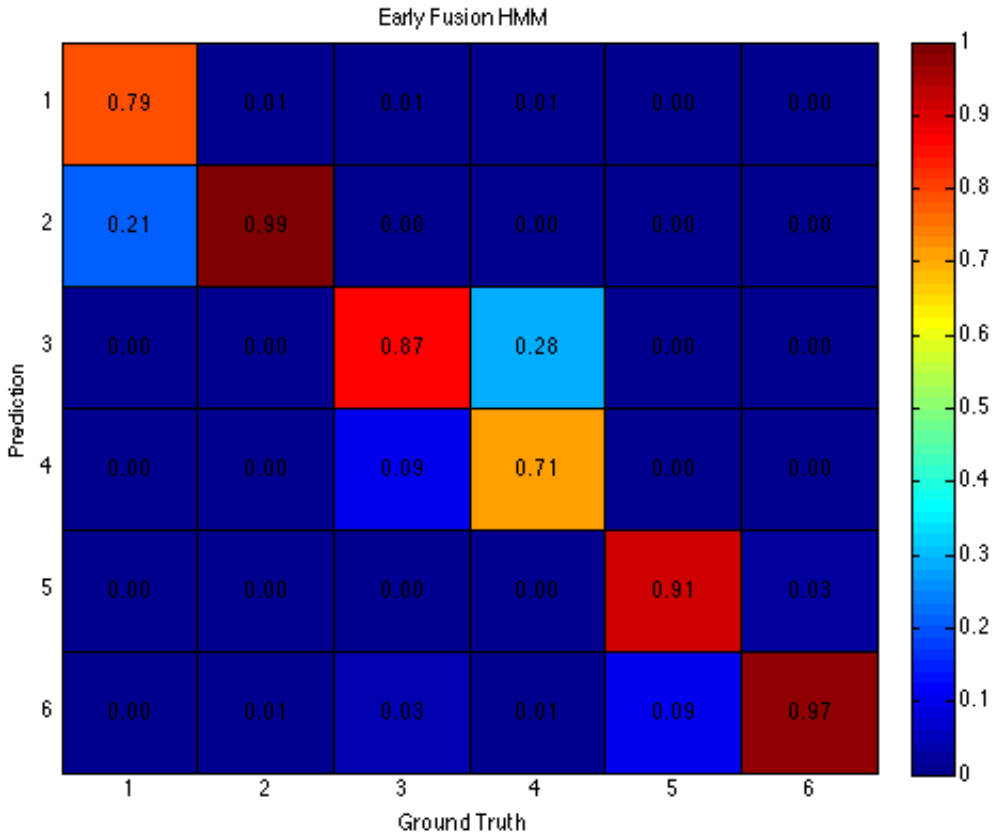
Early fusion simply concatenates the data from the two sources and trains a single classifier on the conjoined data. The same methodology is employed in testing; simply combine the data from all sources, then see how the predictions of the classifier line up with the ground truth.

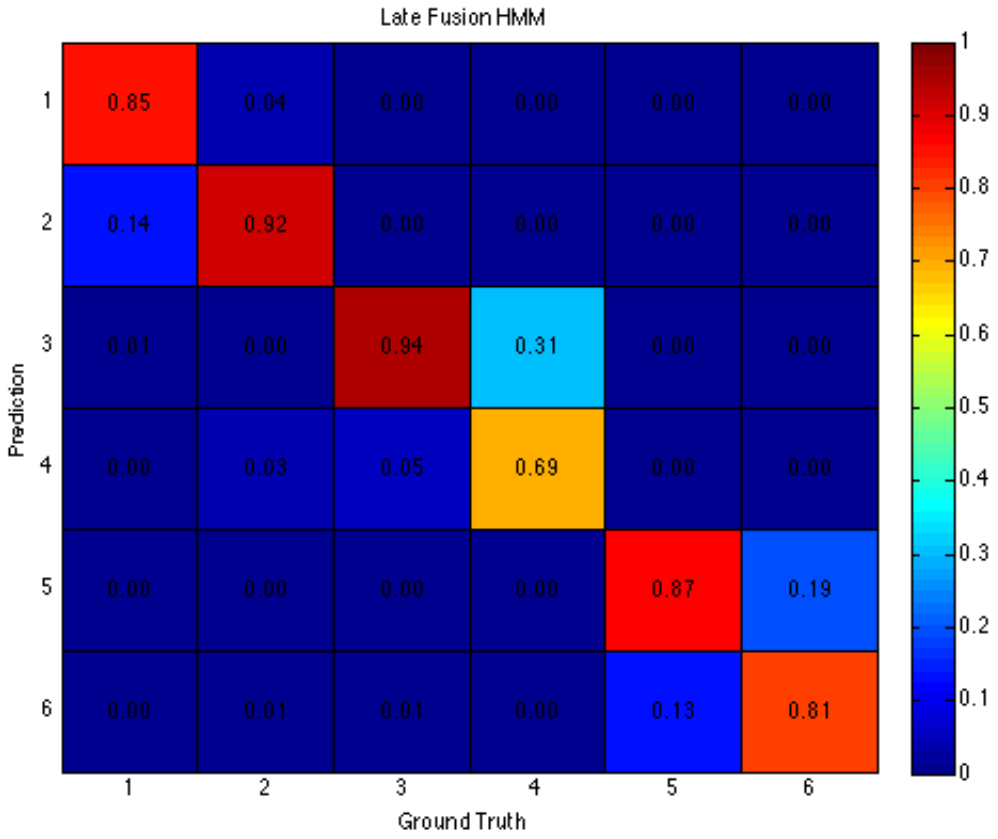
Late fusion, on the other hand, is more flexible about the types of data employed and their relative strengths and weaknesses. While it can't provide the depth of data cross-pollination that early fusion can, it allows the system to have deeper knowledge of the semantics of the data at the time of fusion. For example, a system could fuse high-level semantic events that did not line up perfectly in time by seeing them as semantic occurrences which are related. Furthermore, when one modality has a variable reliability, that reliability can be assessed and weighted independently, improving the system's durability.

Practically speaking, late fusion involves training two classifiers independently, then combining their results intelligently. This might involve heavier weighting for the more reliable modality or detection of offset events, such as conversation about something that is yet to come. Late fusion works well with data sources that are more different from one another than early fusion does.

Question 7: Submit and interpret the two confusion matrices you obtained (both from early fusion and late fusion). Which approach (early versus late) performed better? Pick the confusion matrix that

performed better and compare it to the two confusion matrices you obtained in Question 2. What differences do you see? Do you see a better classification accuracy on those gesture pairs that were confused the most in unimodal approach? Why do you think the performance has improved?





On average, early fusion performed better, giving a slightly higher accuracy. However, late fusion has a more balanced result; that is, the variance of the likelihood of a correct identification is lower. Early fusion does amazingly well on some gestures, but significantly worse on some others, whereas late fusion has fairly reliable performance across all the gestures. As such, I would pick late fusion as the winner.

Late fusion is a drastic improvement over either of the single-modality methods. It seems to have taken the best from each HMM and improved upon it. As it's a simple 50/50 average of the two precursor results, it doesn't do anything unexpected, but the results are still quite striking.

Most powerful to my mind is the results for gestures 1 and 2, especially gesture 1. Neither HMM on its own had a remotely good identification rate for gesture 1; the body HMM even identifies it a majority of the time as gesture 2. However, since their confusion was with *different* red herrings, that confusion gets averaged out entirely, leading to a solid 0.85 identification rate for gesture 1. This is a far better result than the complete confusion between gestures 1 and 2 that the body HMM exhibited.

The performance has improved so much because the pairs of gestures that the body HMM confuses are almost entirely not the pairs that the hand HMM confuses, and each of the HMMs independently puts the correct gesture at least in second place. When the signals are averaged, the unsynchronized noise is averaged out, and the real signal comes through strong.

Question 8: Implement the function `chmm = make_chmm(N,Q,X)` (located inside trainCHMM.m) that generates the graph structure of a coupled HMM. Explain your implementation with pseudocode in your writeup. Note that you must follow the type signature of the function, as the function's input and output parameters are used in the function trainCHMM.m.

My implementation of `chmm = make_chmm(N,Q,X)` is a modification of the `mk_chmm` function included with the BNT library. The `mk_dbn` function allows you to describe the high-level flow of data between time and nodes quite concisely without having to worry about implementation details.

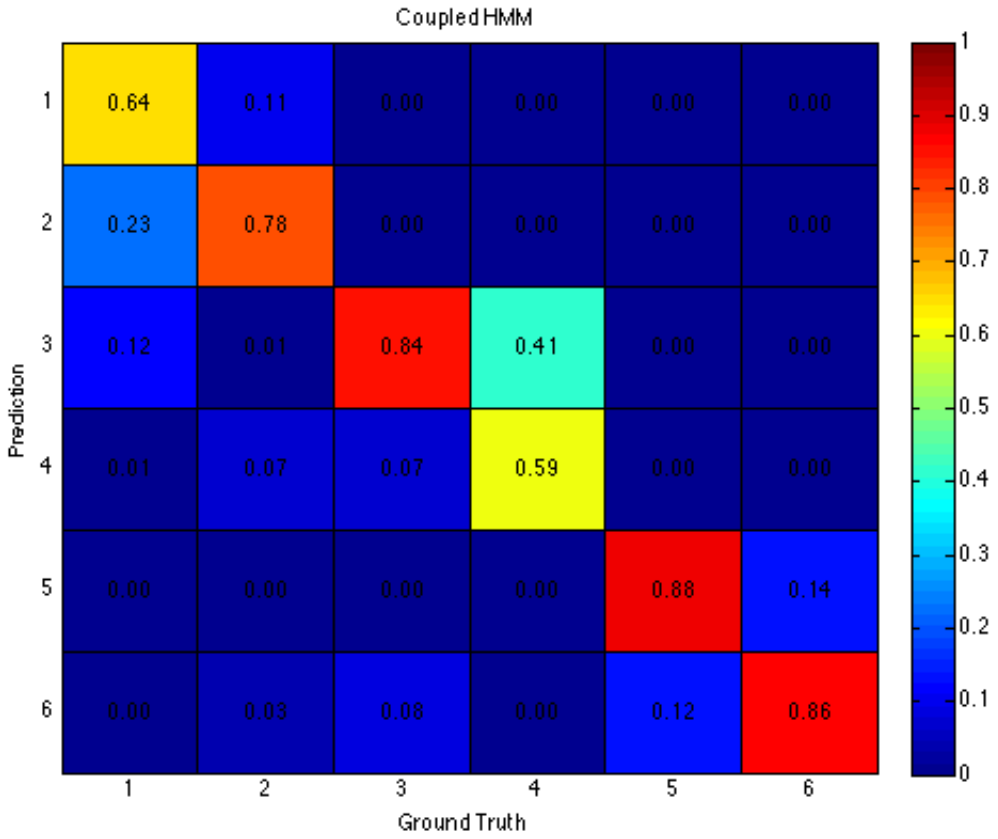
```
Goal: create N hidden nodes, with an observed node for each. Each of the

Define the indices of the hidden nodes to be [1 … N] and those of the ob

Define intra-slice connections between each of the hidden nodes and its
Define inter-slice connections between each of the hidden nodes at time

Specify that nodes [1 …, i, …, N] can take on Q[i] values, and that node

Instantiate a coupled HMM via mk_dbn.

Set the conditional probability distribution of hidden nodes to be tabul
Set the conditional probability distribution of observed nodes to be Gau

Return the coupled HMM.
```

Question 9: Follow the step in Part 3 and run an experiment using coupled HMM. What is the best classification accuracy you obtained using coupled HMM? What parameter values did you validate and what was your strategy for finding the best parameter value?

The best accuracy I obtained was 0.766667 with parameters `H = [8 8], G = [1 1]`. These values were determined using the grid search included in `run.m`, which tested values of `H` in

`{[4 4],[8 8],[12 12]}`.

Question 10: Submit and interpret the confusion matrix you obtained, comparing to the confusion matrices you obtained so far. Does coupled HMM tend to perform better than the early/late fusion HMMs? Why do you think it did (or did not)?



While the coupled HMM performed better than either of the unimodal HMMs alone, it performed rather worse than either of the fusion techniques. In particular, it proved significantly poorer at clarifying situations with multi-pair confusion, really the entire gesture 1-4 area. It did not demonstrate the same discriminating power in cases where each modality would be confused, but in a different pairing, that so impressed me about late fusion. Certainly given its rather extreme computational requirements, this was not a good fit for the data.

I believe this is because the coupled HMM is inherently designed for more complex systems than the one

we're actually examining now. Where fusion can simply take the best information from various sources, coupled HMMs are intended to take into account longer-timebase correlations and complex memories and interrelationships than traditional HMMs are. As the system being modeled was actually a very good fit for traditional Markovian assumptions, we did not reap the benefits of these capabilities.

Question 11: Describe the differences between coupled HMM and early/late fusion HMMs in terms of the underlying assumptions, how the classifiers are trained and then used to test new samples.

Coupled HMMs, while they may look similar to fusion HMMs, are actually entirely different beasts. They rely on the ideas of cross-informing and higher-order information and memory. In situations where the changes in one modality are likely to have some predictive power on the other modality, or where there are time-based processes working on a higher order scale than a single frame, coupled HMMs may work very well.

Fusion HMMs simply try to weigh and combine data from multiple sources; coupled HMMs look for higher-order patterns and behavior across time and the data sources.

A coupled HMM is one larger machine model stitched together out of smaller ones, where each smaller model is fed the data for its own modality while communicating from timestep to timestep with the other nodes. This configuration, used for both training and testing, is more complicated than that of either of the fusion methods, which either just combine the data at the beginning or the answers at the end.

Question 12: What are the two assumptions that a co-training algorithm makes? In the context of the NATOPS dataset, do those assumptions make sense?

Co-training algorithms are algorithms that employ more than one observable variable about the data, in this case the hand data and the body data. These algorithms rely on the idea that these different views into the data can correct one another's ambiguity, and that each of the views is itself able to classify the data. More formally, the two assumptions of co-training are:

1. Conditional Independence. That is, each view into the data must provide answers which are not correlated with those of the other view, except via the correct, ground-truth classification. If the two views are correlated, they will have the same errors, and thus not be able to cross-correct their output.

2. Sufficiency. Each view must be sufficient to accurately classify the data on its own. That way, each of the views can generate labels from a set of unlabeled training data using its own highest-confidence classifications, then train the other view using this newly-labeled data.

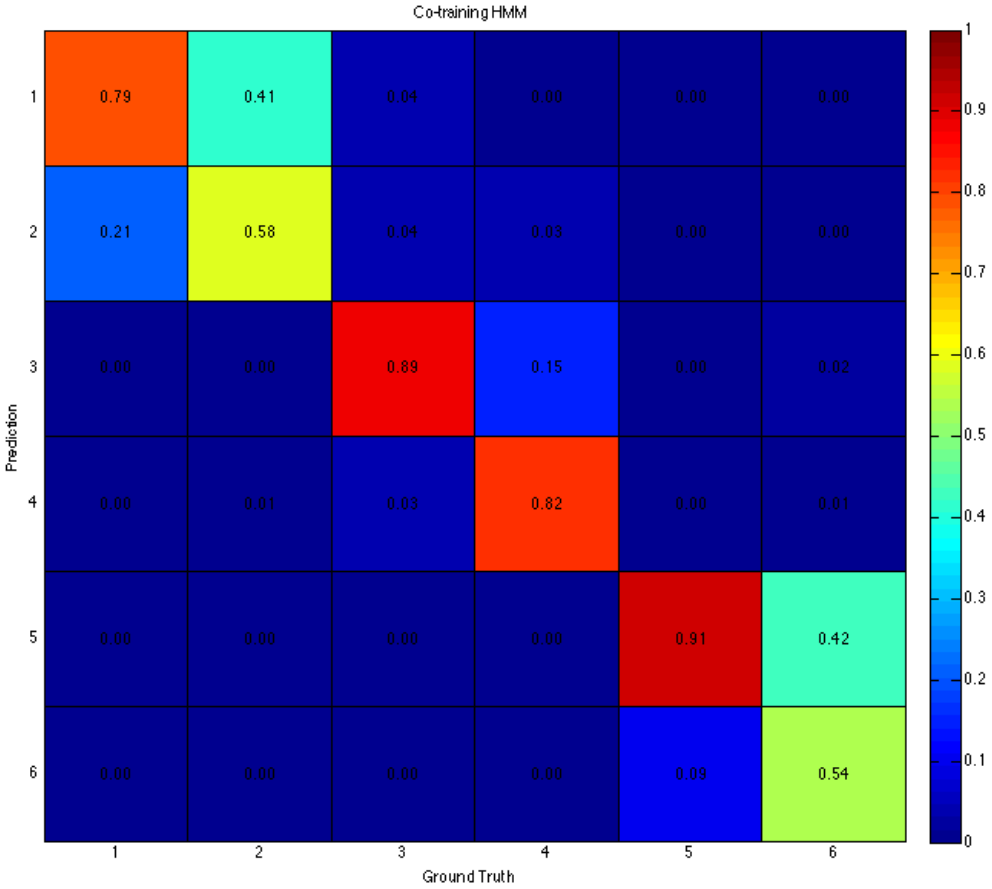
These assumptions seem fairly logical for this problem. Consulting the confusion diagrams for the body-only and hand-only classifiers, their mistakes don't seem to overlap most of the time. The body HMM, for example, frequently labels gesture 4 as gesture 3, but the hand HMM can determine conclusively that gesture 4 is not gesture 3. While neither of them has especially good accuracy for gesture 4, they don't have the *same* problems. They also seem to meet the sufficiency requirement, according to the individual accuracies of the HMMs (0.606250 and 0.642708).

Question 13: Implement trainCoHMM.m that performs co-training of HMMs. We have provided you with pseudocode for implementing co-training algorithms (Figure 3). Note that you must follow the type signature of the function, as the function's input and output parameters are used in the function experiment cotrain_hmm.m.

See included code.

Question 14: Follow the step in Part 4 and run an experiment using co-training HMM. What is the best classification accuracy you obtained using co-training HMM? What parameter values did you try out and what was your strategy for finding the best parameter value? Submit and interpret the confusion matrix you obtained.

My best accuracy was 0.755208 with parameters `H = [8 8], G = [2 2], W = [0.20 0.80]`. I tried parameter values `H = [12 8], G = [1 2]` first, thinking that the optimal parameters for individual performance might work well for co-training. When that didn't work as well as I'd hoped, I tried a grid search on `HMV = {[8 8],[12 12]}` and `GMV = {[2 2],[3 3]}`. This worked fairly well and gave me my results.



This confusion matrix indicates what we could have predicted ahead of time: the co-training HMM, while great at extracting information in some situations (especially gesture 4, which was especially weak for other HMMs), can be misled by outliers that are confident in their mistakes. Hence why gesture 6 is so uncertain; while the hand HMM does very well on gesture 6, the body HMM is majority incorrect on it, and was able to undermine the hand HMM by training it away from accuracy. A similar result occurred in gestures 1 and 2.