K-Means, GMMs and HMMs

Programming Assignment 3 Report

CS5691 - Pattern Recognition and Machine Learning

Team 29 - Abhigyan Chattopadhyay (EE19B146) & Nihal John George (EE19B131)

April 8, 2022

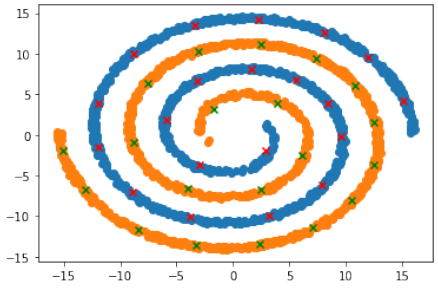
1 Part A – K-Means and Gaussian Mixture Models

We will first use the k-means clustering algorithm to produce the mean values that we will use to initialize our GMMs, followed by Expectation Maximization to calculate the covariances of the GMMs.

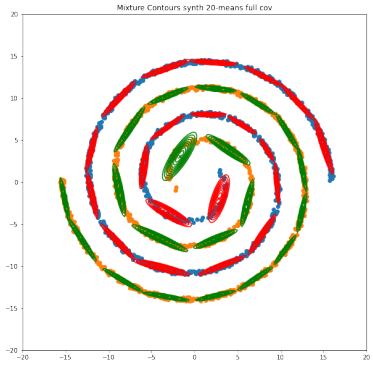
We will first work with the synthetic data.

1.1 Synthetic Data

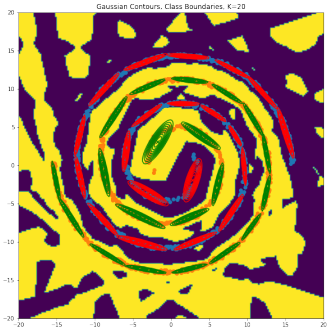
The data set given is 2 intertwined spirals, on which we apply k-means to get the following means (red and green crosses). After this, we apply Expectation Maximization to get the following contours and decision boundary:



(a) After k-means



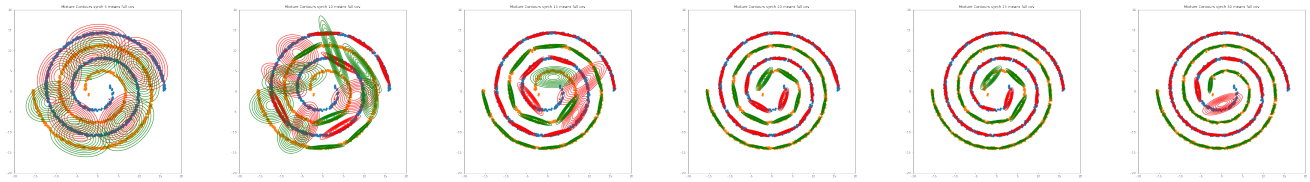
(b) After EM Algorithm



(c) Decision Boundary

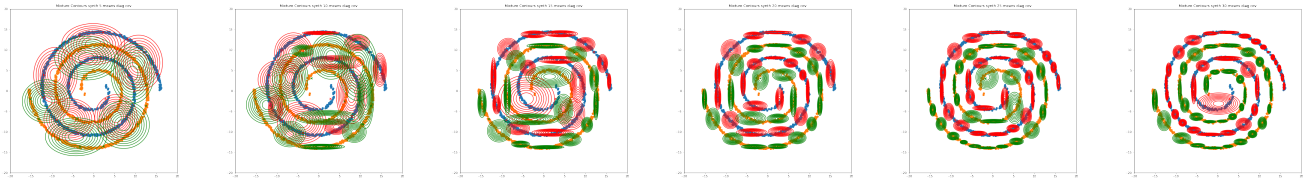
1.1.1 Different Values of k

Next, we iterate over k from 5 to 30 in steps of 5, with both non-diagonal and only diagonal entries in the covariance matrices, to get the following results:



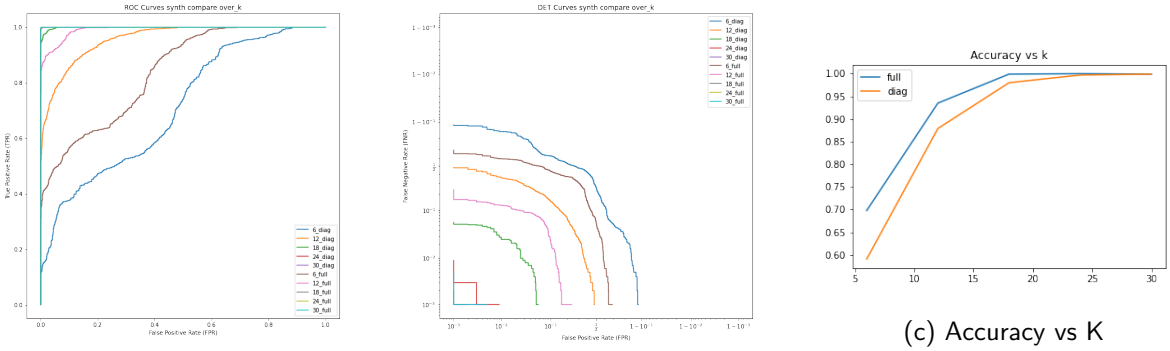
(a) k=5 (b) k=10 (c) k=15 (d) k=20 (e) k=25 (f) k=30

Figure 2: Using Non-Diagonal Covariance Matrices



(a) k=5 (b) k=10 (c) k=15 (d) k=20 (e) k=25 (f) k=30

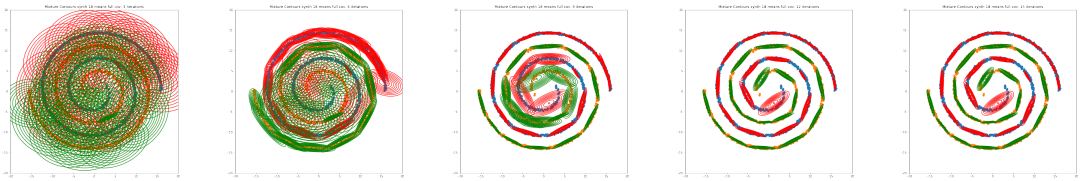
Figure 3: Using Diagonal Covariance Matrices



(a) ROC Curves (b) DET Curves (c) Accuracy vs K

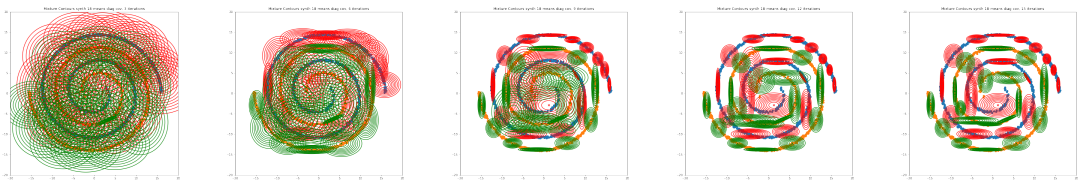
In general, using non-diagonal covariance matrices lead to higher accuracy for a fixed value of k. Using both diagonal and non-diagonal covariance matrices, the accuracy increases with increasing values of k. With large values of k, our model is able to predict values from the dev dataset with 100% accuracy.

1.1.2 Different number of Iterations



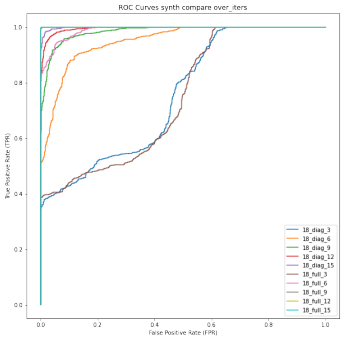
(a) 3 iterations (b) 6 iterations (c) 9 iterations (d) 12 iterations (e) 15 iterations

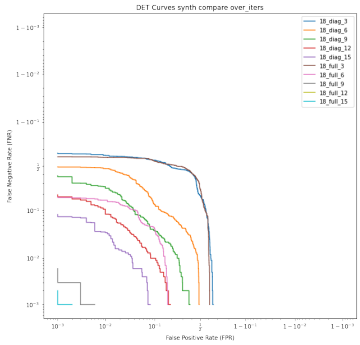
Figure 5: Non-Diagonal Covariance Matrices, k=18



(a) 3 iterations (b) 6 iterations (c) 9 iterations (d) 12 iterations (e) 15 iterations

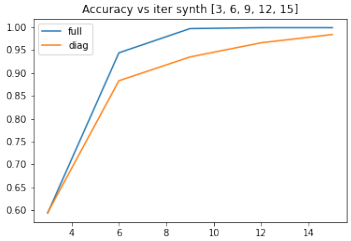
Figure 6: Diagonal Covariance Matrices, k=18





(a) ROC Curves using Diagonal and Non-Diagonal Covariance Matrices

(b) DET Curves using Diagonal and Non-Diagonal Covariance Matrices



(c) Accuracy vs Iterations using Diagonal and Non-Diagonal Covariance Matrices

1.2 Image Data

Synthetic Data - Accuracy vs k		
k	**full**	**diag**
6	69.8%	59.1%
12	93.5%	87.9%
18	99.9%	98%
24	100%	99.7%
30	99.9%	99.9%
Synthetic k=18, Accuracy vs Iterations		
Iterations	**full**	**diag**
3	59.5%	59.4%
6	88.3%	94.4%
9	93.5%	99.7%
12	96.6%	99.9%
15	98.4%	99.9%
Images - Accuracy vs k		
k	**full**	**diag**
6	70.4%	63.2%
12	71.8%	63.8%
18	72.7%	63.5%
24	70.1%	67.5%
30	70.7%	66.9%
Normalized Images - Accuracy vs k		
k	**full**	**diag**
6	53.7%	51.4%
12	63.5%	56.0%
18	64.9%	62.6%
24	66.9%	59.2%
30	69.3%	57.5%

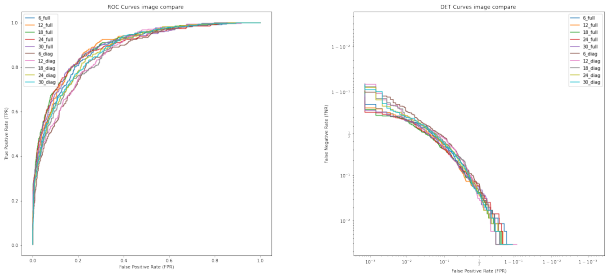


Figure 8: ROC and DET Curves for varying k for Images Dataset

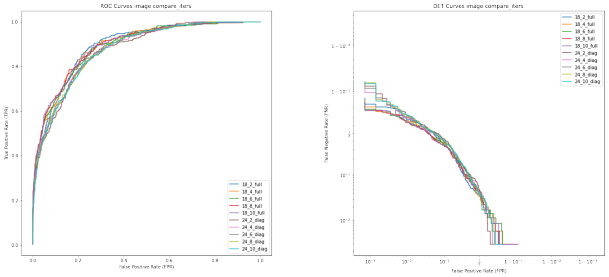


Figure 9: ROC and DET Curves for varying iterations in Images Dataset

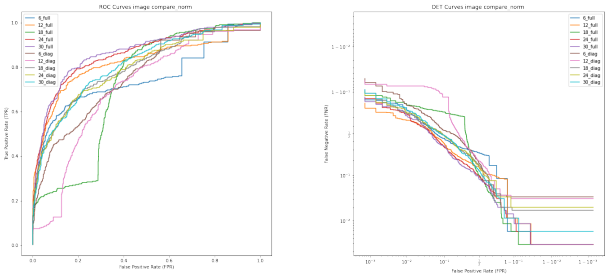


Figure 10: ROC and DET Curves for normalized Images Dataset

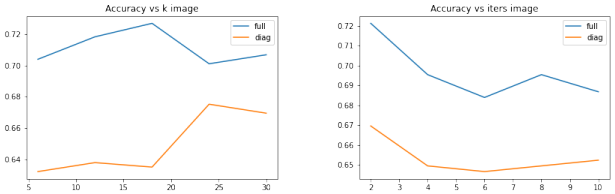


Figure 11: Accuracy vs k and iterations for Images Dataset

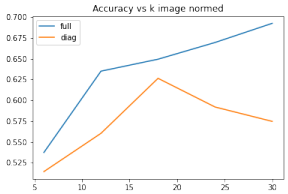


Figure 12: Accuracy vs k for Normalized Images Dataset

1.3 Inferences

1. More iterations of k-means and EM gives better accuracy, but only after the first few iterations. This is expected since the expectations are maximized, so by definition, the accuracy should rise. The first few iterations are just getting updated covariances, so we expect some lag in iteration count before the accuracies rise

2. More means in general gives better results, but only upto a certain extent. After that, the accuracy begins to drop, since too many means will cause overfitting on the points.

3. Non-diagonal covariances take more iterations of EM to reach convergence. Though their accuracy at the end is better, for time stressed applications, it might be better to use diagonal covariances instead.

4. We notice high accuracy on synthetic data since as observed from the pictures, the noise in the data is extremely low. For low k, the means start to come between the spirals, but for higher k, it fits extremely well

2 Part B – DTW + HMM

DTW was implemented from scratch. HMM was done using K-Means from our code for good mean initialization, followed by the C++ code given. Additionally, CWRT has also been implemented from scratch, which gave massive speedups for DTW.

2.1 Spoken Isolated Digits

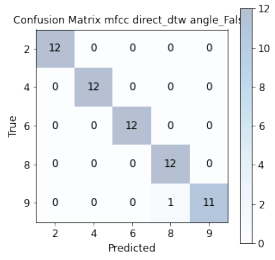
The MFCCs were used directly as the features.

2.1.1 DTW

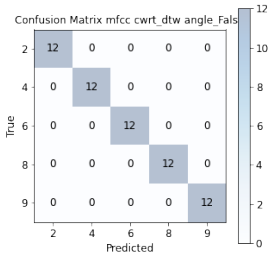
Here, two variants were explored –

Direct: In direct mode, we have no training phase. All train examples are used as templates while testing. The 5 train examples with smallest DTW distance to the dev example provide votes for their respective classes. The class with highest votes is chosen as the prediction.

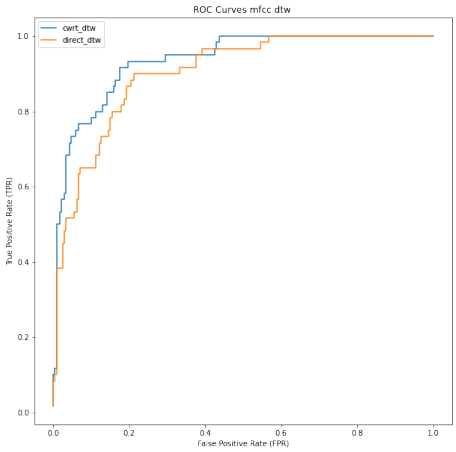
CWRT: In CWRT, we first align all the MFCC sequences of a particular class to the same length, and then average them. This forms the template for the class. For each dev example, the DTW distance of every template to the dev example is computed. The class of the template with min distance to the dev example is chosen as the prediction.



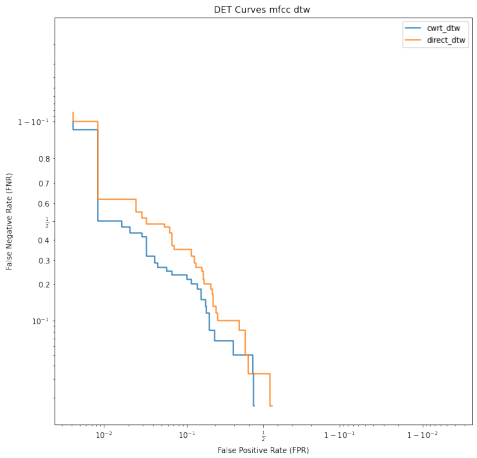
(a) Direct DTW



(b) CWRT DTW



(c) ROC of DTW variants



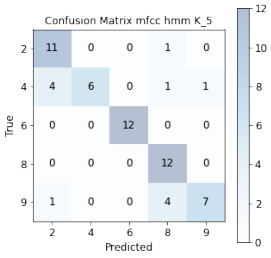
(d) DET of DTW variants

2.1.2 HMM

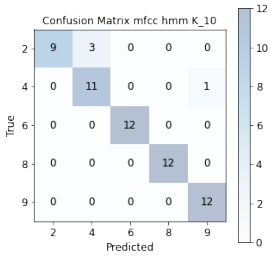
Vector Quantization: First, each frame vector was quantized using means obtained using K-Means clustering over all data across all classes. The symbol sequences were written to appropriately labelled files for use by the C++ code.

Training: A separate HMM was trained for each class by passing all sequences of that class. Random initialization was used for the transition and emission probabilities, while the symbol and state counts were varied from 5 to 24. The executables derived from C++ were compiled and run on Windows Subsystem for Linux 2 (WSL2).

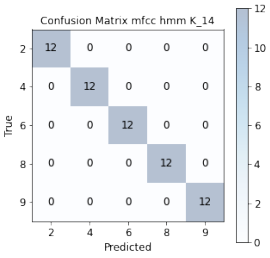
Testing: Dev examples were first quantized using the previously obtained means, and then written to sequence files. The testing C++ code was then used to obtain log probabilities of each sequence occurring on each model. The model with highest probability was chosen as the prediction.



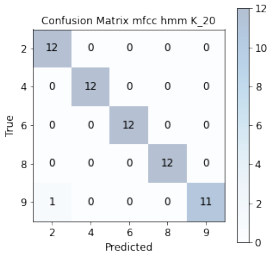
(a) HMM $K = 5$



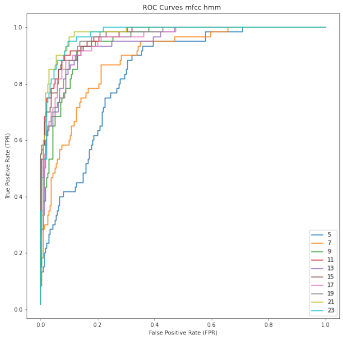
(b) HMM $K = 10$



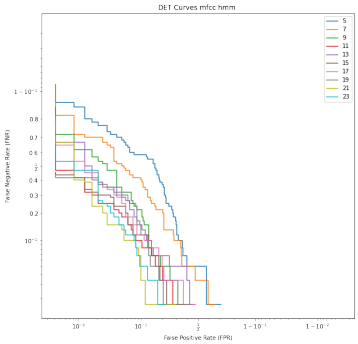
(c) HMM $K = 14$



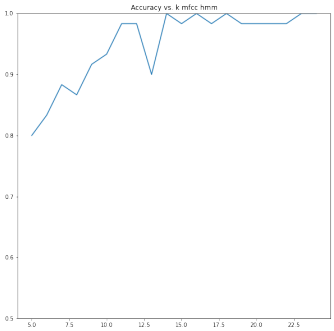
(d) HMM $K = 20$



(e) ROC of HMM for various K



(f) DET of HMM for various K



(g) MFCC HMM Accuracy vs. K

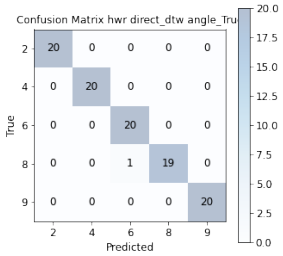
2.2 Handwritten Isolated Characters

The following feature combinations were tried:

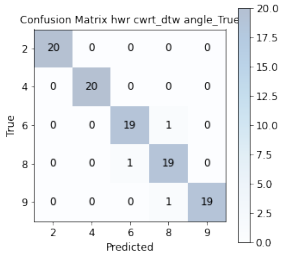
1. **x,y** – The coordinates were standardised by first subtracting the mean of all coordinates in one instance of the letter, then dividing by the standard deviation of points.

2. **angle** – The angle of line segments between two consecutive points was used. This feature is agnostic to the exact coordinates of parts of the letter, which vary over examples. Angle equally penalizes angular deviations, while slope penalizes deviations close to the vertical more heavily (due to high slopes for vertical lines)

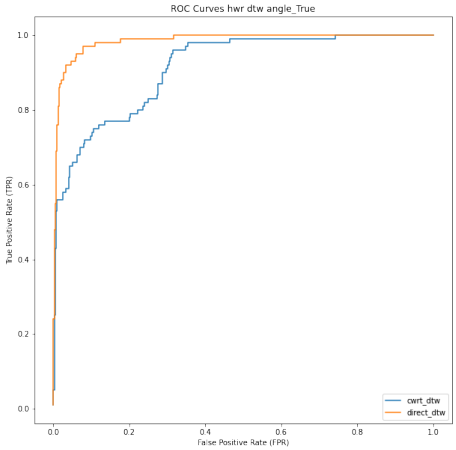
Once these features were extracted, the same procedure for DTW and HMM as done for spoken digits was followed here.



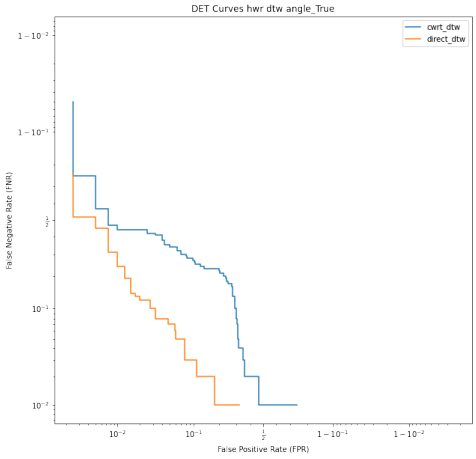
(a) HWR Direct DTW



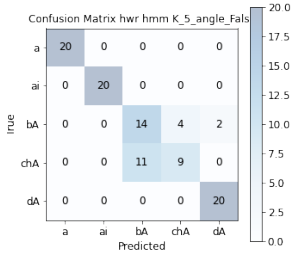
(b) HWR CWRT DTW



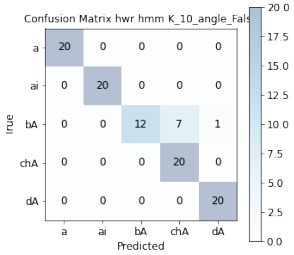
(c) HWR ROC of DTW variants



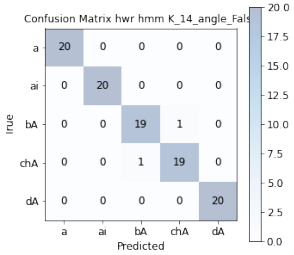
(d) HWR DET of DTW variants



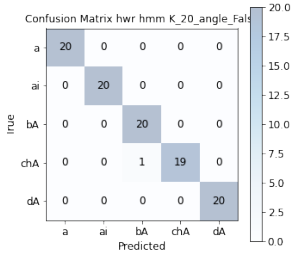
(a) x,y HMM $K = 5$



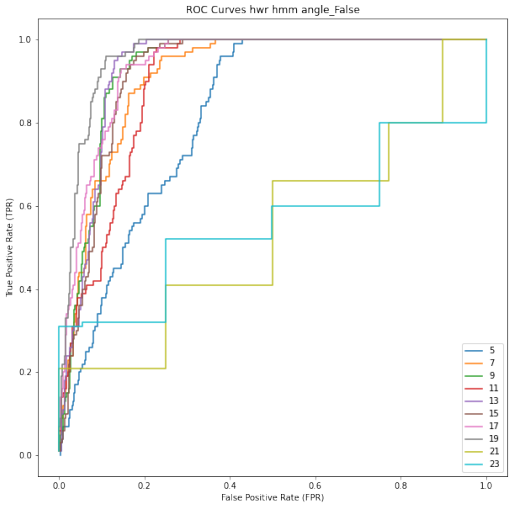
(b) x,y HMM $K = 10$



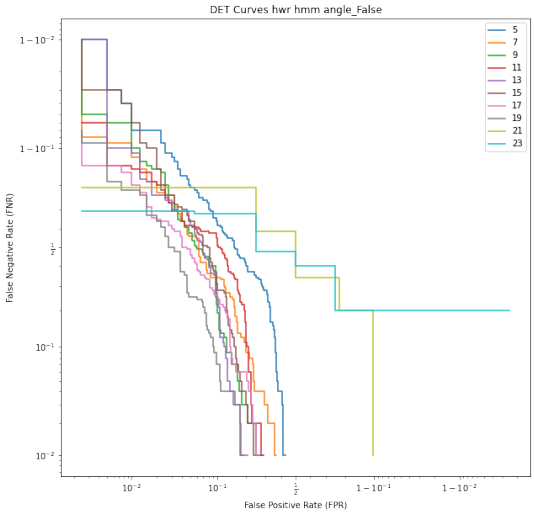
(c) x,y HMM $K = 14$



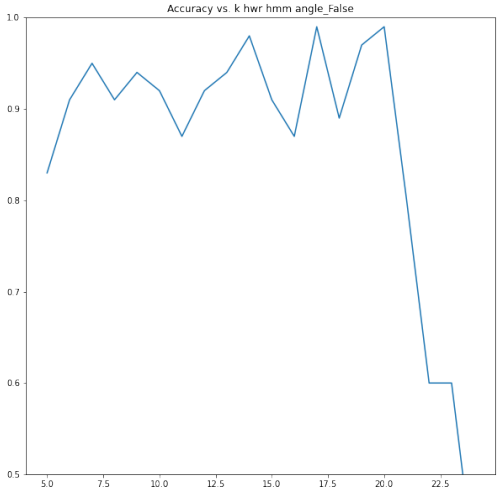
(d) x,y HMM $K = 20$



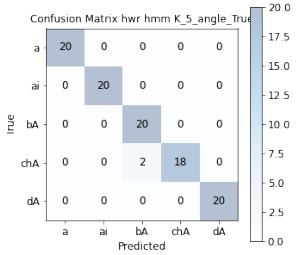
(e) ROC of x,y HMM for various K



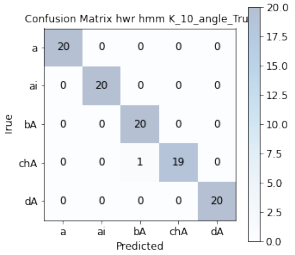
(f) DET of x,y HMM for various K



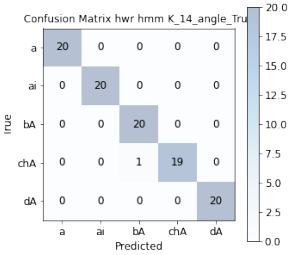
(g) x,y Coords HMM Accuracy vs. K



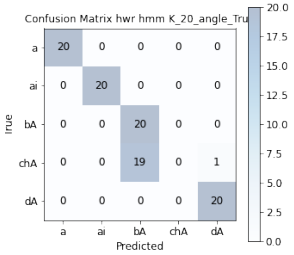
(a) Angle HMM $K = 5$



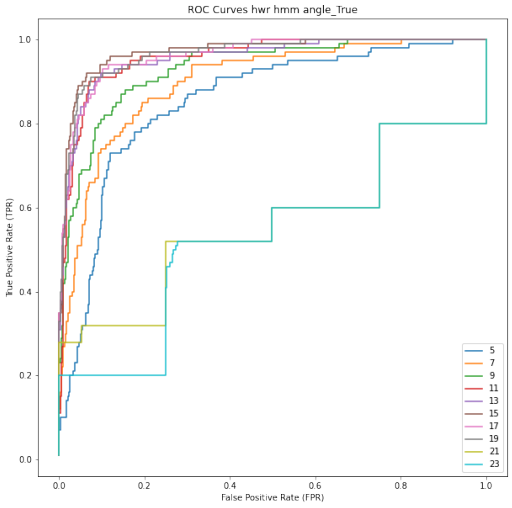
(b) Angle HMM $K = 10$



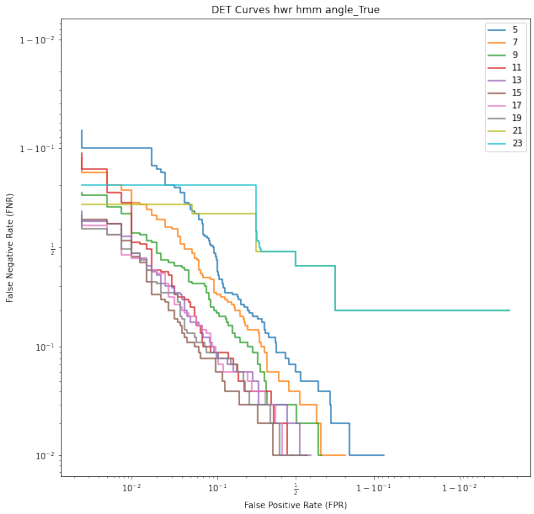
(c) Angle HMM $K = 14$



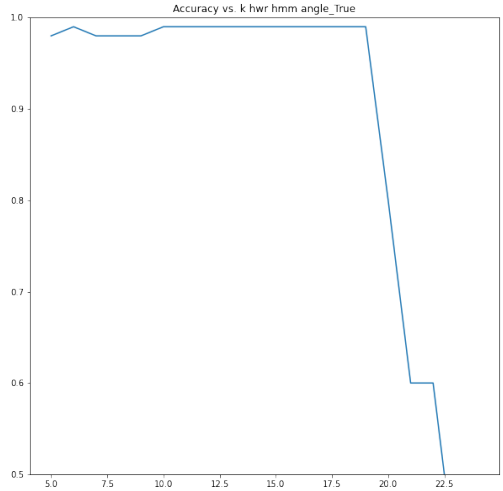
(d) Angle HMM $K = 20$



(e) ROC of Angle HMM for various K



(f) DET of Angle HMM for various K



(g) Angle HMM Accuracy vs. K

2.3 Table of Results

Results	
Dynamic Time Warping - Isolated Spoken Digit MFCC	
Cross Word Reference Template	**100**%
Direct	**100**%
Dynamic Time Warping - Handwriting Recognition	
CWRT only angle	97%
Direct only angle	**99**%
CWRT, x, y, mean, std normed	93%
Direct, x, y, mean, std normed	98%
CWRT x, y	73%
Direct x, y	82%
Hidden Markov Model - Isolated Spoken Digit MFCC	
Sym = State = 14, 16, 18, 23	**100**%
Hidden Markov Model - Handwriting Recognition	
Direct, Sym = State = 14	98%
Direct Sym = State = 17, 20	**99**%
Angle, Sym=5, State=10, 11, 12	**99**%

2.3.1 Inferences

1. There are massive speedups with CWRT since the number of DTW calls has reduced from O(CTD) to O(CD) (C=5 is no. of classes, T=40 is training examples per class, D=20 is no. of dev examples, approx values given). The template generation process is also fast (20 secs).

2. In terms of accuracy, we expected a rise in accuracy by using CWRT with the hypothesis that averaging over training examples weakens noisy examples, which can give erroneous output on direct comparison. However, the numbers show that direct performs equally well, or better than CWRT. The ROC and DET further show the improved performance of direct comparison.

3. Performing normalisation on handwriting data dramatically improved accuracy. On plotting the character data, it is noticed that characters are located at varying x and y coordinates, widths and heights. This variation pollutes the DTW distance function and reduces the contribution of the distances we need. In HMM, this ensures that K-Means will provide better means, since the x,y points are located in nearby places after normalisation.

4. Increasing state count for a particular number of symbols does helps in accuracy up to some extent, but accuracy falls after that. This is because the algorithm begins to overfit on the data, overassigning states to the same symbols.

5. Putting number of states equal to number of symbols is justified, at least for reasonable state count. For a left-to-right HMM, a state is visited only once, occupied for some timesteps, then left. Intuitively, one can assign a state for each location on the character, which corresponds to a symbol. Then on traversing the Markov model, the character strokes will be done in sequence as needed. This is validated by our high accuracies for sym=state

6. Angle has been found to be better as a feature as compared to x,y, slope and curvature due to the locality and equal penalty reasons explained earlier. An add-on feature for the future could be change of angle with time (delta-angle). In fact on observing the connected characters, the characters are printed one after the other with increasing x, indicating that x,y is not a scalable to connected characters.