Pattern Recognition
CS669

ASSIGNMENT 3

Bayes Classifier using KNN and DHMM

Group Number 13

Suryavanshi Virendrasingh	B16037
Abhishek Pal	B15203
Nemani Sri Hari	B15224

Contents

1. Problem Description

Data-sets:

- Real world data set: Consonant Vowel (CV) segment dataset (Speech)
- Data of each class is given separately. For all data in Dataset, 75% of the examples of a class is to be used as training data for that class, and the remaining data is to be used as test data for that class.

Classifiers:

- Bayes classifier using K-nearest neighbour method for class-conditional density estimation using DTW (Dynamic Time Warping) distance.
- Bayes classifier using Discrete HMM (DHMM)

Objective:

- Classification accuracy, precision for every class, mean precision, recall for every class, mean recall, F-measure for every class and mean F-measure on test data.
- Confusion matrix based on the performance for test data

2. Solution Approach

1 K-Nearest Neighbours

It is a non-parametric method used for classification and regression (supervised algorithm). First the distance of a point from all the points is taken, and then sorted according to distance. Then k least distant are taken into consideration. An object is classified by a majority vote of its neighbors. If k = 1, then the object is assigned to the class of that single nearest neighbor.

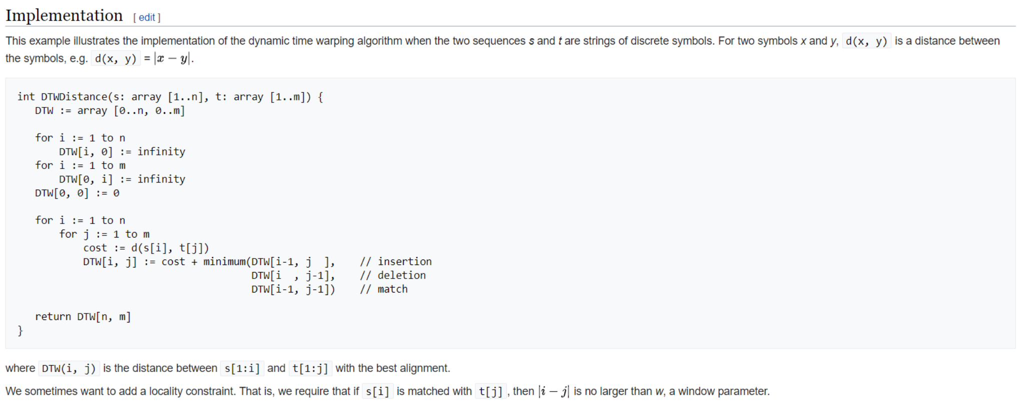
1.1 Algorithm

The training examples are vectors in a multidimensional feature space (here 39d), each with a class label. First all the feature vectors and class labels of the training samples are stored in a data-structure and then k (number of neighbors to consider) is chosen. A commonly used distance metric for continuous variables is Euclidean distance. We have used DTW (Dynamic Time Warping) to calculate the distance between two observation sequences.

1.2 Dynamic Time Warping

In time series analysis, dynamic time warping (DTW) is one of the algorithms for measuring similarity between two temporal sequences, which may vary in speed. For instance, similarities in walking could be detected using DTW, even if one person was walking faster than the other, or if there were accelerations and decelerations during the course of an observation. DTW has been applied to temporal sequences of video, audio, and graphics data indeed, any data that can be turned into a linear sequence can be analyzed with DTW. A well known application has been automatic speech recognition, to cope with different speaking speeds. Other applications include speaker recognition and online signature recognition. Also it is seen that it can be used in partial shape matching application.

Implementation:



Implementation [edit]

This example illustrates the implementation of the dynamic time warping algorithm when the two sequences *s* and *t* are strings of discrete symbols. For two symbols *x* and *y*, `d(x, y)` is a distance between the symbols, e.g. `d(x, y)` = $|x - y|$.

```
int DTWDistance(s: array [1..n], t: array [1..m]) {
   DTW := array [0..n, 0..m]

   for i := 1 to n
       DTW[i, 0] := infinity
   for i := 1 to m
       DTW[0, i] := infinity
   DTW[0, 0] := 0

   for i := 1 to n
       for j := 1 to m
           cost := d(s[i], t[j])
           DTW[i, j] := cost + minimum(DTW[i-1, j  ],    // insertion
                                       DTW[i  , j-1],    // deletion
                                       DTW[i-1, j-1])    // match

   return DTW[n, m]
}
```

where `DTW(i, j)` is the distance between `s[1:i]` and `t[1:j]` with the best alignment.

We sometimes want to add a locality constraint. That is, we require that if `s[i]` is matched with `t[j]`, then $|i - j|$ is no larger than *w*, a window parameter.

Source: Wikipedia
Figure 2..1

2 K-Means Clustering and Vector Quantization

The observations in observation sequences which we were given to do HMM upon was 39 dimensional, to make computation easier it was converted into 1dimensional. The k-means algorithm takes as input the number of clusters to generate, k, and a set of observation vectors to cluster. It returns a set of centroids, one for each of the k clusters. An observation vector is classified with the cluster number or centroid index of the centroid closest to it. A vector v belongs to cluster i if it is closer to centroid i than any other centroids. If v belongs to i, we say centroid i is the dominating centroid of v. The k-means algorithm tries to minimize distortion, which is defined as the sum of the squared distances between each observation vector and its dominating centroid. Each step of the k-means algorithm refines the choices of centroids to reduce distortion. The change in distortion is used as a stopping criterion: when the change is lower than a threshold, the k-means algorithm is not making sufficient progress and terminates. One can also define a maximum number of iterations. Since vector quantization is a natural application for k-means, information theory terminology is often used. The centroid index or cluster index is also referred to as a code and the table mapping codes to centroids and vice versa is often referred as a code book. The result of k-means, a set of centroids, can be used to quantize vectors. Quantization aims to find an encoding of vectors that reduces the expected distortion.

2.1 DHMM (Discrete Hidden Markov Model)

The discrete hidden Markov Model works on the similar to a FSM(Finite state machine). It is assumed that the system is a Markov process with hidden states. In HMM there are usually two tasks performed, one is to find the optimal state sequence and second one is classification task. To find the optimal state sequence we use Viterbi algorithm which is not part of this assignment. In this particular assignment we are asked to classify the speech data MFCC (Mel Frequency Cepstral Coefficient) features into three classes using DHMM. For this first we used vector quantization based on k-means clustering technique and converted all the 39 dimensional observations into symbols. Now we trained DHMM for every class separately using Baum–Welch algorithm. Next we used Forward-algorithm to evaluate the test data.

The Baum–Welch algorithm is similar to EM (Expectation-Maximization) method. We have two steps in this method, firstly in E-step we calculate zeta which is probability of being in state i at time t and in state j in time t+1 given the observation sequence and model. Then we compute for gamma which is prosterior probability of state or probability of transition to state i at time t.

$$\zeta_{\mathrm{t}}(\mathrm{i,j}) \;=\; \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \tag{2..1}$$

$$\gamma_{\mathrm{t}}(\mathrm{i}) \;=\; \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \tag{2..2}$$

Alpha and beta can be computed by forward-algorithm and backward-algorithm respectively. Alpha is the probability of observing partial sequence (1 to t) until time t and being at state i at time t given the Markov model. Beta is the probability of observing partial observation sequence (t+1 to T) given state i at time t and Markov model.

$$\alpha_{t+1}(j) \;=\; (\sum_{i=1}^{N}\alpha_t(i)a_{ij})bj(O_{t+1}) \tag{2..3}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}b_j(O_{t+1})\beta_{t+1}(j) \tag{2..4}$$

Now we can compute A, B matrices in M-step using zeta and gamma as follows:

$$a_{ij} \;=\; \frac{\sum_{t=1}^{T-1}\zeta_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \tag{2..5}$$

$$\underline{\ }j(V_k) \;=\; \frac{\sum_{t=1}^{T}\gamma_t(j) \mid O_t = V_k}{\sum_{t=1}^{T}\gamma_t(j)} \tag{2..6}$$

Now repeat this E and M steps till convergence. The convergence criteria is the change in probability of all the data in class given Markov model should be maximized. The probability of one observation sequence given Markov model is given by:

$$P(O \mid \lambda) \;=\; \sum_{i=1}^{N}\alpha_T(i) \tag{2..7}$$

After training is done,we can test the data by applying either Forward-algorithm (equations (2.6), (2.10)) or Backward-algorithm on all the Markov models obtained and maximum of which gives us the class it belongs. We used Forward-algorithm for testing.

3. Results

1 K-nearest neighbour using DTW distance

K-value : 4
Accuracy : 55.38%

	Class1	Class2	Class 3
Class1	13	13	1
Class2	1	19	4
Class3	1	9	4

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	75.38	58.46	76.92
Precision	86.67	46.34	44.44
Recall	48.15	79.17	29.00
F-Measure	61.00	58.46	34.78

(b) Analysis (in Percentage)

Table 3..1. Result for all the 3 classes

K-value: 8
Accuracy : 60.00%

	Class1	Class2	Class 3
Class1	13	14	0
Class2	1	21	2
Class3	1	9	5

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	76.92	60.00	83.08
Precision	92.86	47.73	71.43
Recall	48.15	87.50	36.00
F-Measure	63.41	61.76	47.62

(b) Analysis (in Percentage)

Table 3..2. Result for all the 3 classes

K-value : 16
Accuracy : 58.46%

	Class1	Class2	Class 3
Class1	14	13	1
Class2	3	20	1
Class3	1	10	4

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	75.38	58.46	83.08
Precision	82.35	46.51	80.00
Recall	51.85	83.33	29.00
F-Measure	63.64	59.70	42.11

(b) Analysis (in Percentage)

Table 3..3. Result for all the 3 classes

K-value : 32
Accuracy : 58.46%

	Class1	Class2	Class 3
Class1	11	16	1
Class2	1	22	1
Class3	1	9	5

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	73.85	58.46	84.62
Precision	91.67	46.81	83.83
Recall	40.74	91.67	36.00
F-Measure	56.41	61.97	50.00

(b) Analysis (in Percentage)

Table 3..4. Result for all the 3 classes

2 Bayes Classifier Using Discrete HMM

Length of Codebook: 8

Number of States : 2
Accuracy : 44.61%

	Class1	Class2	Class 3
Class1	25	2	0
Class2	23	0	1
Class3	10	0	4

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	46.15	59.99	83.07
Precision	43.10	0	79.99
Recall	92.59	0	28.57
F-Measure	58.82	0	42.10

(b) Analysis (in Percentage)

Table 3..5. Result for all the 3 classes

Number of States : 3
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	23	4	0
Class2	18	5	1
Class3	6	2	6

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	56.92	61.53	86.15
Precision	48.93	45.45	87.71
Recall	85.18	20.83	42.85
F-Measure	62.16	28.57	57.14

(b) Analysis (in Percentage)

Table 3..6. Result for all the 3 classes

Number of States : 4
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	23	4	0
Class2	19	4	1
Class3	5	2	7

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	56.92	59.99	87.69
Precision	48.93	39.99	87.49
Recall	85.18	16.66	49.99
F-Measure	62.16	23.52	63.63

(b) Analysis (in Percentage)

Table 3..7. Result for all the 3 classes

Number of States : 5
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	23	4	0
Class2	19	4	1
Class3	5	2	7

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	56.92	59.99	87.69
Precision	48.93	39.99	87.49
Recall	85.18	16.66	49.99
F-Measure	62.16	23.52	63.63

(b) Analysis (in Percentage)

Table 3..8. Result for all the 3 classes

Length of Codebook: 16

Number of States : 2
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	16	9	2
Class2	11	10	3
Class3	4	2	8

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	59.99	61.53	83.07
Precision	51.61	47.61	61.53
Recall	59.25	41.66	57.14
F-Measure	55.17	44.44	59.25

(b) Analysis (in Percentage)

Table 3..9. Result for all the 3 classes

Number of States : 3
Accuracy : 58.46%

	Class1	Class2	Class 3
Class1	17	8	2
Class2	10	11	3
Class3	3	1	10

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	64.61	66.15	86.15
Precision	56.66	54.99	66.66
Recall	62.96	45.83	71.42
F-Measure	59.64	49.99	68.96

(b) Analysis (in Percentage)

Table 3..10. Result for all the 3 classes

Number of States : 4
Accuracy : 58.46%

	Class1	Class2	Class 3
Class1	17	8	2
Class2	10	11	3
Class3	3	1	10

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	64.61	66.15	86.15
Precision	56.66	54.99	66.66
Recall	62.96	45.83	71.42
F-Measure	59.64	49.99	68.96

(b) Analysis (in Percentage)

Table 3..11. Result for all the 3 classes

Number of States : 5
Accuracy : 58.46%

	Class1	Class2	Class 3
Class1	17	8	2
Class2	10	11	3
Class3	3	1	10

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	64.61	66.15	86.15
Precision	56.66	54.99	66.66
Recall	62.96	45.83	71.42
F-Measure	59.64	49.99	68.96

(b) Analysis (in Percentage)

Table 3..12. Result for all the 3 classes

Length of Codebook: 32

Number of States : 2
Accuracy : 50.70%

	Class1	Class2	Class 3
Class1	18	7	2
Class2	13	8	3
Class3	6	1	7

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	56.92	63.07	81.53
Precision	48.64	49.99	58.33
Recall	66.66	33.33	49.99
F-Measure	56.24	39.99	53.84

(b) Analysis (in Percentage)

Table 3..13. Result for all the 3 classes

Number of States : 3
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	18	7	2
Class2	12	9	3
Class3	6	1	7

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	58.46	64.61	81.53
Precision	49.99	52.94	58.33
Recall	66.66	37.48	49.99
F-Measure	57.14	43.90	53.84

(b) Analysis (in Percentage)

Table 3..14. Result for all the 3 classes

Number of States : 4
Accuracy : 55.38%

	Class1	Class2	Class 3
Class1	19	7	1
Class2	11	10	3
Class3	5	2	7

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	63.07	64.61	83.07
Precision	54.28	52.63	63.63
Recall	70.37	41.66	49.99
F-Measure	61.29	46.51	55.99

(b) Analysis (in Percentage)

Table 3..15. Result for all the 3 classes

Number of States : 5
Accuracy : 52.30%

	Class1	Class2	Class 3
Class1	16	9	2
Class2	11	10	3
Class3	4	2	8

(a) Confusion Matrix

	Class1	Class2	Class3
Accuracy	59.99	61.53	83.07
Precision	51.61	47.61	61.53
Recall	59.25	41.66	57.14
F-Measure	55.17	44.44	59.25

(b) Analysis (in Percentage)

Table 3..16. Result for all the 3 classes

4. Inferences

- In case of two states most of the data is mis-classified into class1 i.e 'di' as there only two states it is difficult get the variations in 'd' and 'D' sounds as well as 'i' and 'I' sounds as it gets just the variations in D-group i.e. 'd', 'D' and I-group i.e. 'I', 'i'. It can be observed from Table 3.5, 3.9, 3.13.
- We observed that there is not much difference in the accuracy when 3,4,5 states are used. The reason can be that the sounds 'di', 'dI' and 'DI' can at most be divided into 3 different types of sounds that needs to joined to produce them. This can be observed from Table 3.6-8, 3.10-12, 3.14-16.
- The value of alpha in Forward-algorithm goes on decreasing as it has negative exponent of 10 and gets multiplied with values less than 1. In order to obtain good results logarithm and scaling can be applied over alpha calculation.