Using Vector Quantization with Concavities

Luiz S. Oliveira and Marisa Morita

Federal University of Parana (UFPR)
Department of Informatics (DInf)
R. Rua Cel. Francisco H. dos Santos, 100, Curitiba, PR, Brazil
lesoliveira@inf.ufpr.br

1 Introduction

In this report we report the results achieved by the concavity feature vector introduced in Report #4. Since the concavity feature vector is composed of real values, it needs to be quantized so that we can have a discrete feature vector to use with the HMMs models. With this in mind, before discussing the experiments we describe the vector quantisation process and then present the performance of the concavity feature vector on both databases we have been using in this project, Brazilian and Chilean.

2 Vector Quantization

To use the HMMs described in the previous reports, we need to represent an image as a sequence of discrete symbols. To do so, each feature vector extracted from the image that contains real values (low-level-feature) needs to be quantized to one of the discrete symbols (high-level feature) available in a previously computed codebook. To create this codebook, it is necessary to apply the concept of vector quantization [1].

Let us assume that $X = [X_1, X_2, \ldots, X_d]$ is a d-dimensional vector whose components $\{X_k, 1 \leq k \leq d\}$ are real-valued, continuous-amplitude random variables. In vector quantization, the vector X is mapped onto another real-valued, discrete-amplitude d-dimensional vector Z. It is used to say that X is quantized as Z. This can be denoted by:

$$Z = q(X) \tag{1}$$

where $q()$ is the quantization operator. Z takes one of a finite set of values $W = \{Z_i, 1 \leq i \leq L\}$ where $Z_i = [Z_{i1}, Z_{i2}, \ldots, Z_{id}]$. The set W is referred to as the codebook, L is the size of codebook

(or number of levels in the codebook), and $\{Z_i\}$ is the set of codewords. To design a codebook, we divide the d-dimensional space of the original random vector X into L regions or cells $\{C_i, 1 \leq i \leq L\}$ and associate with each cell C_i a vector Z_i. The quantizer then assigns the codeword Z_i if X is in C_i.

$$q(X) = Z_i, \quad if \quad X \in C_i \tag{2}$$

The mapping of X onto Z results in a quantization error, and a distortion measure $d(X, Z)$ can be defined between X and Z to measure the quality of quantization. In this case, we have used the Euclidean distance which is the most commonly used distortion measure.

Quantization is optimal when the overall average distortion defined in Equation 5 is minimized over all L-levels of the quantizer. There are two necessary conditions for optimality. The first condition is that the optimal quantizer is realized by using a nearest neighbor selection rule.

$$q(X) = Z_i, \quad iff \quad d(X, Z_i) \leq d(X, Z_j), \quad j \neq i, \quad 1 \leq j \leq L \tag{3}$$

This means that the quantizer selects the codeword vector that results in the minimum distortion with respect to X. The second condition for optimality is that each codeword Z_i is chosen to minimize the average distortion in cell C_i. Let us consider $\{X(n), 1 \leq n \leq M\}$ as a set of training vectors and K_i as a subset of vectors located in cell C_i. The average distortion D_i in cell C_i and the overall average distortion $D_{overall}$ are then given by:

$$D_i = \frac{1}{K_i} \sum_{X \in C_i} d(X, Z_i) \tag{4}$$

$$D_{overall} = \sum_{i=1}^{L} D_i \tag{5}$$

The vector that minimizes the average distortion in cell C_i is called the centroid of C_i, and it is denoted as:

$$Z_i = cent(C_i) \tag{6}$$

Z_i is then obtained from:

$$Z_i = \frac{1}{K_i} \sum_{X \in C_i} X \tag{7}$$

One well-known method for codebook design is an iterative clustering algorithm known in the

pattern recognition literature as the K-Means algorithm [1]. The basic idea of such an algorithm is to divide the set of training vectors into L clusters $C_i\{1 \leq i \leq L\}$ in such a way that the two necessary conditions for optimality are satisfied. The algorithm can be described as follows:

- Initialization step: choose randomly a set of initial centroids $(Z_i, 1 \leq i \leq L)$.
- Classification step: classify each element of training vectors $X(n)$ into one of the clusters C_i by choosing the nearest codeword $Z_i(X \in C_i, iff \quad d(X, Z_i) \leq D(X, Z_j) \quad for \quad all \quad j \neq i)$.
- Codebook updating step: update the codeword of each cluster by computing the centroid of the training vectors in each cluster $(Z_i = cent(C_i), 1 \leq i \leq L)$.
- Termination step: if the decrease in the overall distortion $D_{overall}$ at the current iteration relative to the overall distortion at the previous iteration is below a certain threshold, stop; otherwise go to the classification step.

3 Experimental Results

As stated previously, our goal was to create a discrete version of our 18-dimensional concavity feature vector so that we could use these features to train the HMM models. Since we do not know the optimal number of centroids for this feature set, our first experiment consisted in looking for this number. To do that, we have tried the following number of centroids: 15, 20, 40, 60, 80, and 100. Figure 1 depicts the performance for Top1 and Top2 of the concavity feature vector on the Brazilian database for six different number of centroids.

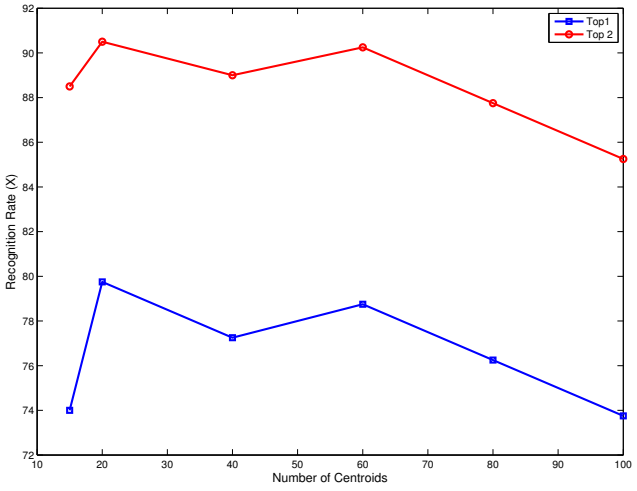

Figure 1: Performance of the concavity feature vector for six different number of centroids

As we can notice, the best results, 79.7% (Top1) and 90.5% (Top2), were achieved when using 20 centroids in the vector quantization. After 20 centroids we get no improvement in terms of performance. However, after analysing the confusion matrices for 20 and 60 centroids, we may observe that the classifiers make different mistakes in some classes. The same holds when we compare the confusion matrices of the concavities and global features. This may indicate that combining different configurations of vector quantization and global features could bring some benefits. This will be subject of future investigation.

Regarding the Chilean database, we have selected the best configuration of centroids we have found in the Brazilian dataset, i.e., 20, to replicate the experiment. Table 1 shows the confusion matrix for this experiment. Compared with the experiment using global features (Report #5), the concavity feature set brought an improvement of about 7 percentage points, achieving a recognition rate of 49.1%, against 42.2% of the global features.

Table 1: Confusion Matrix for the Chilean dataset using Concavity Features with 20 centroids (%) - TOP1

Class	1	2	3	4	5	6	7	8	9
1	**57.1**	13.2	1.1	3.3	3.3	2.2	12.1	4.4	3.3
2	9.6	**64.9**	1.1	5.3	1.1	1.1	4.3	4.3	8.5
3	20.0	25.3	**21.1**	0.0	10.5	8.4	3.2	10.5	1.1
4	16.8	11.6	2.1	**36.8**	9.5	4.2	6.3	9.5	3.2
5	24.0	16.7	17.7	2.1	**25.0**	2.1	8.3	3.1	1.0
6	17.5	4.1	1.0	1.0	4.1	**62.9**	8.2	1.0	0.0
7	10.0	3.0	0.0	4.0	1.0	14.0	**66.0**	0.0	2.0
8	10.1	16.2	6.1	5.1	3.0	2.0	5.1	**51.5**	1.0
9	5.9	0.0	0.0	0.0	0.0	0.0	0.0	5.9	**88.2**

4 Conclusions

In this report we have used a vector quantization strategy to convert the real-valued concavity feature vector into a discrete one. Different number of centroids were tried out and the best results were achieved using 20 centroids. For the Brazilian database, the performance of the concavity feature set is slightly inferior to the one achieved by the classifier trained with global features. It is worth of remark, though, that an error analysis may suggest that the combination of these two feature sets may bring some improvements in terms of performance.

The best configuration found in the experiments using the Brazilian dataset was replicated to the Chilean database. In such a case, the concavity feature set with 20 centroids achieved a recognition rate of 49.1%, 7 percentage points better than the experiment using global features.

Combination of different number of centroids will also considered in this case to see if we can further improve these results. It is still worth of remembering that the HMM models should be retrained as soon as we get all the characters available in the Chilean month words.

References

[1] X. D. Huang, Y. Ariki, and M. A. Jack, *Hidden Markov Models for Speech Recognition*, Edinburgh Univ. Press, 1990.