Discriminative clustering

Nikolaj Tatti, ntatti@cc.hut.fi

16th March 2004

Summary based on [2, 3]

1 Introduction

Let us assume that we have (finite or infinite) paired data in form of (x, c), where $x \in R^M$ and c is an integer $c \in \{1, \ldots, N\}$. We refer the vector x as primary data and the integer c as auxiliary data or class. Our problem is to cluster primary data such that we use the information in auxiliary data. This is different than simply to divide the primary data into auxiliary data classes since we may have different number of clusters and also primary data in a cluster should be compact.

2 Learning vector quantization

The definition of discriminative clustering is based on learning vector quantization (LVQ). The idea is to minimise the average distortion

$$E = \sum_{j=1}^{K} \int_{V_j} D(x, m_j) p(x) dx, \tag{1}$$

where K is the number of clusters, m_j is the prototype of the cluster j, and $D(x, m_j)$ is the distortion function. The Voronoi cell V_j is defined

$$x \in V_j \iff D(x, m_j) \leq D(x, m_k), \text{ for all } k.$$

Let us now consider the probability of the class given the data point $p(c \mid x)$. Assume also a distribution ψ defined on a domain $\{1, \ldots, N\}$. The Kulback-Leibler divergence between these two distributions is

$$D_{KL} = \sum_{i=1}^{N} p(c = i \mid x) \log \left(\frac{p(c = i \mid x)}{\psi(i)} \right). \tag{2}$$

Let us now modify the average distortion. For each cluster j we introduce a distributional prototype ψ_j which is a distribution defined on a domain $\{1, \ldots, N\}$. These prototypes try to represent the conditional distributions $p(c \mid x)$ in each Voronoi cell V_j. Our new average distortion is

$$E_{KL} = \sum_{j=1}^{K} \int_{V_j} D_{KL}(p(c \mid x), \psi_j) p(x) dx. \tag{3}$$

However, the Voronoi cells are still defined according to the original distortion $D(x, m_j)$. We choose to use the Euclidean distance as $D(x, m_j)$. Thus, the Voronoi cells are defined

$$x \in V_j \iff \|x - m_j\| \leq \|x - m_k\|, \text{ for all } k.$$

Our minimisation problem involves finding the optimal distributional prototypes ψ_j and the primary data prototypes m_j.

3 Finite data

Let us now assume that we have a finite number of data points. The average distortion in Eq. 3 is not feasible since we do not the probability $p(x)$ and the conditional probabilities $p(c \mid x)$. We replace the $\int_{V_j} \cdots p(x)dx$ with $\sum_{x \in V_j}$ and set $p(c \mid x)$ to be

$$p(c = i \mid x) = \begin{cases} 1 & i = c(x) \\ 0 & \text{otherwise} \end{cases}.$$

The distortion transforms into

$$E_{KL} = \sum_{j=1}^{K} \sum_{x \in V_j} D_{KL}(p(c \mid x), \psi_j) = -\sum_{j=1}^{K} \sum_{x \in V_j} \log \psi_j(c(x)). \tag{4}$$

Thus minimising the average distortion is equivalent to maximising

$$L = \sum_{j=1}^{K} \sum_{x \in V_j} \log \psi_j(c(x)) = \log \prod_{j=1}^{K} \prod_{x \in V_j} \psi_j(c(x)) = \log p(D \mid \{\psi_j\}, \{V_j\}).$$

This is the log-likelihood of data given the Voronoi cells V_j and distributional prototypes ψ_j.

4 Bayesian approach

We are not actually interested in finding distributional prototypes - they are nuisance parameters, so we marginalise them out. Also instead of using log-likelihood we calculate the log-MAP by introducing the prior

$$p(\{V_j\}, \{\psi_j\}) = \prod_{j=1}^{K} p(\psi_j) = \prod_{j=1}^{K} \prod_{i=1}^{N} \psi_j(i)^{s_i - 1}.$$

In other words, we set improper prior to the Voronoi cells $\{V_j\}$ and Dirichlet prior to distributional prototypes $\{\psi_j\}$. Let us now set r_{ji} to be the number of

data samples of class i in cluster j. Let also $R_j = \sum_{i=1}^{N} r_{ji}$ to be the number of samples in cluster j. Set also $S = \sum_{i=1}^{N} s_i$. Using these notations the log-MAP is equal to

$$\begin{aligned}
\log p(\{V_j\} \mid D) &= \log \int_{\{\psi_j\}} p(\{V_j\}, \{\psi_j\} \mid D) \\
&= \log \int_{\{\psi_j\}} p(D \mid \{V_j\}, \{\psi_j\}) p(\{V_j\}, \{\psi_j\}) \\
&= \log \int_{\{\psi_j\}} \prod_{j=1}^{K} \prod_{i=1}^{N} \psi_j(i)^{s_i + r_{ji} - 1} \\
&= \sum_{j=1}^{K} \sum_{i=1}^{N} \log \Gamma(s_i + r_{ji}) - \sum_{j=1}^{K} \log \Gamma(S + R_j).
\end{aligned} \tag{5}$$

This is the function that is minimised during the discriminative clustering. Note that this cost function should be used when we have finite data. The cost function given in Equation 3 should be used when we have infinite or very large data set.

5 Connection to the contingency tables

Good [1] represented a Bayesian test for checking whether the contingency table. The idea is to use the Bayes factor

$$\frac{P(D \mid H)}{P(D \mid \bar{H})},$$

where $\bar{H}$ is hypothesis such that the variables in the contingency table are independent.

Form a contingency table such that the first variable are the classes and the second are the clusters. The entries of this table are equal to r_{ji} for class i and cluster j (See notation in the previous section). If we use Dirichlet priors for the entries r_{ji}, then the Bayes factor is equal to

$$\frac{P(D \mid H)}{P(D \mid \bar{H})} = \prod_{j=1}^{K} \prod_{i=1}^{N} \frac{\Gamma(s_i + r_{ji})}{\Gamma(S + R_j)} \times \text{const.}$$

This is virtually the same formula as in Equation 5. Thus discriminative clustering tries to maximise the dependency between the clusters and auxiliary data under the constraints.

6 Inferring algorithms

There is no to our knowledge easy way to infer the codebook for hard clustering. Thus we introduce *soft* clusters. Let $y_j x$ be the membership function of cluster j such that $0 \leq y_j(x) < 1$ and $\sum_{j=1}^{K} y_j(x) = 1$. For example, $y_j(x) = Z(x) \exp\left(\|x - m_j\| / \sigma^2\right)$, where $Z(x)$ is the normalisation constant. In this case we search optimal m_j.

Let us now introduce an online inferring algorithm for large data set. The function to be minimised is given in Equation 3. To ease the notation set γ_j such that $\log \psi_j = \gamma_j - \log \sum_{i=1}^{N} \exp(\gamma_j(i))$. Let $x(t)$ and $c(t)$ be the data samples at step t. Draw two clusters k and l according to the distribution $y(x(t)) = [y_1(x(t)), y_2(x(t)), \ldots, y_N(x(t))]$. Update the prototypes m_l and ψ_l (that is γ_l)

$$\begin{aligned} m_l &\Leftarrow m_l - \alpha(t)\,[x(t) - m_l] \log \frac{\psi_k(c(t))}{\psi_l(c(t))} \\ \gamma_l(i) &\Leftarrow \gamma_l(i) - \alpha(t)\,[\psi_l(i) - \delta_{il}]\,, \end{aligned}$$

where δ_{il} is Kronecker delta and i ranges over $\{1, \ldots, N\}$. $\alpha(t)$ is the learning schedule and it is chosen traditionally.

Let us now look at the finite data set. We use again soft clustering. In this case the 'number' of data points of class i in cluster j is equal to $r_{ji} = \sum_{c(x)=i} y_j(x)$. Also set $R_j = \sum_x y_j(x)$. The function to be minimised is equal to

$$\log p(\{V_j\} \mid D) = \sum_{i=1}^{N} \sum_{j=1}^{K} \log \Gamma(s_i + \sum_{c(x)=i} y_j(x)) - \sum_{j=1}^{K} \log \Gamma(S + \sum_{x} y_j(x)).$$

If the membership functions are chosen wisely e.g.,

$$y_j(x) = Z(x) \exp\left(\|x - m_j\| / \sigma^2\right),$$

then we can solve the gradient of the cost function and apply some known gradient descent method.

7 Soft clusters vs. Information Bottleneck

The membership functions $y_j(x)$ can be considered as a random variable $v = [v_1, v_2, \ldots, v_N]$ in the sense that $p(v_j \mid x) = y_j(x)$. The cost function for soft clusters is equal to

$$E_{KL} = \sum_{j=1}^{K} \int p(v_j \mid x) D_{KL}(p(c \mid x), \psi_j) p(x) dx. \tag{6}$$

Given some data point x, we assume that $p(v \mid x)$ and $p(c \mid x)$ are independent. Thus, $p(v, c \mid x) = p(v \mid x)p(c \mid x)$.

It is a known fact that at the minimum of the cost function in Eq. 6 the distributional prototypes ψ_j are equal to

$$\psi_j(i) = p(c_i \mid v_j) = \frac{1}{p(v_j)} \int_x p(v_j, c_i \mid x) p(x) = \frac{1}{p(v_j)} \int_x y_j(x) p(c_i \mid x) p(x).$$

Applying this fact to Eq. 6 lead us to equation

$$\begin{aligned}
E_{KL} &= \sum_{j=1}^{K} \int p(v_j \mid x) D_{KL}(p(c \mid x), \psi_j) p(x) dx \\
&= \sum_{j=1}^{K} \sum_{i=1}^{N} \int p(v_j \mid x) p(c_i \mid x) \log \left(\frac{p(c_i \mid x)}{p(v_j \mid c_i)} \right) p(x) dx \\
&= -\sum_{j=1}^{K} \sum_{i=1}^{N} \int p(v_j, c_i \mid x) \log \left(p(v_j \mid c_i) \right) p(x) dx + \text{const}_1 \\
&= -\sum_{j=1}^{K} \sum_{i=1}^{N} \int p(v_j, c_i \mid x) \log \left(\frac{p(v_j, c_i)}{p(v_i)} \right) p(x) dx + \text{const}_1 \\
&= -\sum_{j=1}^{K} \sum_{i=1}^{N} \int p(v_j, c_i \mid x) \log \left(\frac{p(v_j, c_i)}{p(c_j) p(v_i)} \right) p(x) dx + \text{const}_2 \\
&= -\sum_{j=1}^{K} \sum_{i=1}^{N} p(v_j, c_i) \log \left(\frac{p(v_j, c_i)}{p(c_j) p(v_i)} \right) + \text{const}_2 \\
&= -I(c, v) + \text{const}_2,
\end{aligned}$$

where $I(c, v)$ is the mutual information between random variables c and v and const_1 and const_2 are expressions independent of v. Thus, in our problem we are maximising the mutual information between the auxiliary data and the membership functions.

In information bottleneck [4], the function to be minimised is $I(D, v) - \beta I(c, v)$, where D is primary data, v are the soft clusters (membership functions) and c is the auxiliary data. The cost is almost the same as in our case except that there is a term $I(D, v)$ whose purpose is to regularise the solution: Otherwise we could simply optimise this function by selecting $p(v_j, c_i) = \delta_{ji} p(c_i)$. This would imply that $I(D, v) = I(D, c)$. This value is assumed to be high. In discriminative clustering the regularisation is handled by parametrising the membership functions and therefore there is no need for any regularisation term.

The major difference between these two approaches is that the primary data in discriminative clustering is real valued and discrete in the information bottleneck.

References

[1] I. J. Good. On the application of symmetric dirichlet distributions and their mixtures to contingency tables. *The Annals of Statistics*, 4(6):1159–1189, Nov. 1976.

[2] Janne Sinkkonen, Same Kaski, and Janne Nikkila. Discriminative clustering: Optimal contingency tables by learning metrics. In Heikki Mannila Tapio Elomaa and Hannu Toivonen, editors, *Proceedings of the ECML'02, 13th European Conference on Machine Learning*, pages 418–430. Springer, 2002.

[3] Janne Sinkkonen and Sami Kaski. Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14:217–239, 2002.

[4] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.