Discriminative clustering

Nikolaj Tatti
Department of Computer Science
Helsinki University of Technology
ntatti@cc.hut.fi

January 2004

Problem setting

- Consider that data comes in pairs (x, c), where x is primary data $x \in R^M$ and c is auxiliary data. In our case auxiliary data is discrete classes $c \in \{1, \ldots, N\}$.
- We want to cluster data such that we use the information available in auxiliary data.
- This is different than dividing primary data into auxiliary data classes since we have some constraints: For example, the number of clusters may be different than the number of auxiliary data classes. Also primary data should be compact in each cluster.

Menu

- Some basic definitions.
- The first version of the discriminative clustering.
- Discriminative clustering for finite data (ML/MAP).
- Connection to contingency tables.
- Inferring algorithms.
- Soft clusters vs. Information Bottleneck.

Learning Vector Quantization

- In theory, LVQ involves with the minimisation of the average distortion

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

  where K is the number of cluster, m_j is a prototype of the cluster j, $D(x, m_j)$ is a distortion function.
- V_j is a Voronoi cell defined $x \in V_j$, if $D(x, m_j) \leq D(x, m_k)$, for all k.

Kullback-Leibler

- Let $p(c \mid x)$ be the distribution of auxiliary data given the primary data point.
- Define

$$D_{KL}(p(c \mid x), \psi) = \sum_{j=1}^{N} p(c = j \mid x) \log \left(\frac{p(c = j \mid x)}{\psi(j)} \right),$$

where ψ is some distribution defined on the domain $\{1, \ldots, N\}$.

Discriminative Clustering

- Associate with each cluster an additional prototype ψ_j that represents the conditional probabilities $p(c \mid x)$ in each Voronoi cell V_j.
- Define the average distortion

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

- However, Voronoi cells are defined traditionally: $x \in V_j$, if $\|x - m_j\| \leq \|x - m_k\|$, for all k.
- Thus, minimising the average distortion involves finding optimal primary data prototypes m_j and optimal conditional probabilities prototypes ψ_j.

Finite data

- Assume now that we have a finite number of data samples. In this case we do not know the probabilities $p(x)$ and $p(c \mid x)$.
- We replace $\int_{V_j} \cdots p(x)dx$ with $\sum_{x \in V_j}$ and set

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

- Thus E_{KL} transforms into

$$E_{KL} = -\sum_{j=1}^{K} \sum_{x \in V_j} \log \psi_j(c(x)).$$

Maximum Likelihood

- The distortion $-E_{KL}$ can be expressed in form

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

  where D is data (primary and auxiliary), $\{\psi_j\}$ is the family of the prototypes and $\{V_j\}$ is the family of the Voronoi cells.
- In other words, L is the log-likelihood of data D given the prototypes $\{\psi_i\}$ and the Voronoi cells $\{V_j\}$.
- Minimising the average distortion for the finite data is equal to maximising the likelihood.

Bayesian approach

- Instead of maximising the log-likelihood $\log p(D \mid \{\psi_j\}, \{V_j\})$ we can marginalise the prototypes out and maximise the log-posterior $\log p(\{V_j\} \mid D)$.
- We set the prior $p(\{V_j\}, \{\psi_j\}) \propto \prod_{j=1}^{K} p(\psi_j)$ and

$$p(\psi_j) \propto \prod_{i=1}^{N} \psi_j(i)^{s_i - 1}.$$

Bayesian approach cont'd

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

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

- This is the function that should be maximised during the discriminative clustering.

Contingency tables

- Form a contingency table such that the first margin is the auxiliary data classes and the second is the clusters. The element of the table r_{ji} is the number of data samples of class i in cluster j.
- Bayesian test for dependency in the contingency is a Bayes factor

$$\frac{p(\{r_{ij}\} \mid \bar{H})}{p(\{r_{ij}\} \mid H)},$$

  where H is the independence hypothesis between the clusters and the classes.
- This factor result large values if there is a strong dependence between the clusters and the classes.

Connection to the contingency tables

- The Bayes factor (under some specific priors) is equal to

$$\frac{p(\{r_{ij}\} \mid \bar{H})}{p(\{r_{ij}\} \mid H)} \propto \frac{\prod_{i=1}^{K} \prod_{j=1}^{N} \Gamma(r_{ji} + s_i)}{\prod_{j=1}^{N} \Gamma(R_j + S)} = p(\{V_j\} \mid D).$$

- Thus, discriminative clustering maximises dependency in the contingency table under the constraints.

Inferring algorithms

- We represent two algorithms. The first algorithm is of online type and assumes that we have infinite (or very large) data set and minimises the original cost function E_{KL}. The second algorithm assumes that we have finite number of data samples and maximises the log-posteriori.
- It is hard to give inferring algorithms for hard clusters, so we use soft clustering.
- Each cluster has a membership function $1 \leq y_j(x) \leq 0$ such that $\sum_{j=1}^{K} y_j(x) = 1$.
- For example, $y_j(x) = Z(x) \exp\left(\|x - m_j\| / \sigma^2\right)$, where $Z(x)$ is a normalisation constant.

Online algorithm

- Let $(x(t), c(t))$ be the data sample at step t.
- Draw independently two clusters k and l according to distribution $y_j(x(t))$.
- Reparametrise the distributional prototypes $\log \psi_j = \gamma_j - \log \sum_{i=1}^{N} \exp(\gamma_j(i))$.
- 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 a Kronecker delta.

Offline algorithm

- The smoothed version of the number of samples r_{ij} of class i in cluster j is $r_{ji} = \sum_{c(x)=i} y_j(x)$.
- Also, $R_j = \sum_x y_j(x)$.
- The function to be maximised is now

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

- If the membership functions y_j are chosen wisely, then we can calculate the gradient of $\log p(\{V_j\} \mid D)$ respect to the prototypes m_j and maximise the function by using our favourite gradient algorithm.

Soft clusters vs. Information Bottleneck

- The membership functions $[y_1(x), y_2(x), \ldots, y_K(x)]$ can be interpreted as random variable $p(V \mid x)$, where $V = [v_1, v_2, \ldots, v_K]$ and $p(v_j \mid x) = y_j(x)$.
- At minimum of E_{KL} the prototypes are equal to $\psi_j(i) = p(c_i \mid v_j)$.

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

Soft clusters vs. Information Bottleneck cont'd

- Thus $E_{KL} = -I(C,V) + const$, where $I(C,V)$ is the mutual information of auxiliary data and cluster probabilities.
- In Information Bottleneck a codebook V is formed such that $I(X,V)$ is maximised under the constraints $I(V,C) < k$.
- The functional to be minimised is $I(X,V) - \beta I(C,V)$. This is almost the same as the function in the discriminative clustering.
- The purpose of $I(X,V)$ is to regularise the solution. This is not needed in the discriminative clustering since the solution is constrained by the form and the number of the membership functions.
- Also X is multinomial in Information Bottleneck and continuous in the discriminative clustering.

Conclusions

- We derive discriminative clustering cost function by adding the Kullback-Leibler divergence of auxiliary data into standard LVQ technique.
- For finite data we showed that this cost function is the same as the log-likelihood. We extend this by marginalising the distributional prototypes out and obtaining the MAP solution.
- We represent two different algorithms. The first is online and is meant for large data sets and it uses the original formula. The second algorithm uses the MAP solution by maximising the probability by using some gradient algorithm.