8. Supervised learning of HMMs: Empirical risk minimisation

Given: i.i.d. training data $\mathcal{T} = \{(x^j, s^j) \mid x^j \in \mathcal{X}^n, s^j \in K^n, j = 1, \dots, m\}$
and the loss function $\ell(s, s') = [\![s \neq s']\!]$

Recall that the optimal predictor $h: \mathcal{X}^n \to K^n$ for 0/1 loss is

$$h_u(x) \in \operatorname*{argmax}_{s \in K^n} p_u(x, s) = \operatorname*{argmax}_{s \in K^n} \langle \varphi(x, s), u \rangle$$

Learning the model parameters u by empirical risk minimisation

$$\frac{1}{m} \sum_{(x,s) \in \mathcal{T}} [\![s \neq h_u(x)]\!] \to \min_u$$

is not tractable because the objective function is piece-wise const.

Special case: Suppose $\exists\, u^*$ s.t. the empirical risk on $\mathcal{T}$ is zero.
Conditions for such u^* are

$$s \in \operatorname*{argmax}_{s' \in K^n} p_{u^*}(x, s') \quad \forall (x, s) \in \mathcal{T}$$

or, equivalently

$$\langle \varphi(x, s), u^* \rangle > \langle \varphi(x, s'), u^* \rangle \quad \forall s' \neq s, \; \forall (x, s) \in \mathcal{T}$$

This is a system of linear inequalities $\Rightarrow$ perceptron alg.
Start with $u = 0$ and iterate

(1) find $\tilde{s} = \operatorname*{argmax}_{s' \in K^n} \langle \varphi(x, s'), u \rangle \quad \forall (x, s) \in \mathcal{T}$.

This can be done by dynamic programing (Sec. 4)

(2) if for some $(x, s) \in \mathcal{T}$: $\tilde{s} \neq s$, update u by

$$u \to u + \varphi(x, s) - \varphi(x, \tilde{s})$$

The algorithm converges to a solution u^* in a finite number of steps (provided it exists).

General case: overcome the intractability be replacing the loss (as a function of u) by a convex upper bound. E.g. „margin rescaling" loss

$$[\![s \neq h_u(x)]\!] = [\![s \neq \operatorname*{argmax}_{s' \in K^n} \langle \varphi(x, s'), u \rangle]\!] \leq$$

$$\leq \max_{s' \in K^n} \left\{ [\![s \neq s']\!] + \langle \varphi(x, s') - \varphi(x, s), u \rangle \right\}$$

The empirical risk minimisation for this loss reads

$$\frac{1}{m} \sum_{(x,s) \in T} \max_{s' \in K^n} \left\{ [\![s \neq s']\!] + \langle \varphi(x, s') - \varphi(x, s), u \rangle \right\} \rightarrow \min_u$$

Solve it by (stochastic) subgradient descent or cutting plane algorithm or ... . The inner optimisation tasks $\max_{s' \in K^n} \{\ldots\}$ are solved by an algorithm like the one in Sec. 4

Remark 1 This approach is designated as „structured output SVM" and can be generalised for more complex losses as e.g. the Hamming distance.

9. Unsupervised learning: EM algorithm for HMMs

Given: i.i.d. training data $\mathcal{T} = \{x^j \in \mathcal{X}^n \mid j = 1, \dots, m\}$

The ML estimator reads $u^* \in \arg\max_u \underbrace{\frac{1}{|\mathcal{T}|} \sum_{x \in \mathcal{T}} \log \sum_{s \in K^n} p_u(x,s)}_{L(u)}$

Recall the EM algorithm

$$L(u) = \frac{1}{m} \sum_{x \in \mathcal{T}} \log \sum_{s \in K^n} \frac{\alpha(s \mid x)}{\alpha(s \mid x)} p_u(x,s)$$

for any $\alpha(s \mid x) \geq 0$ s.t. $\sum_{s \in K^n} \alpha(s \mid x) = 1 \quad \forall\, x \in \mathcal{T}$.

Using concavity of log, we get the lower bound

$$\cancel{L_B(u)} \quad L_B(u, \alpha) = \frac{1}{m} \sum_{x \in \mathcal{T}} \sum_{s \in K^n} \alpha(s \mid x) \log \frac{p_u(x,s)}{\alpha(s \mid x)}$$

or equivalently

$$L_B(u, \alpha) = \mathbb{E}_{\mathcal{T}} \left[\log p_u(x) - D_{KL}\big(\alpha(s \mid x) \,\|\, p_u(s \mid x)\big) \right]$$

The EM algorithm maximises $L_B(u, \alpha)$ by block-coordinate ascent w.r.t. α and u. Start with some $u^{(0)}$.

E-step maximise $L_B(u^{(t)}, \alpha)$ w.r.t. α $\Rightarrow$

$$\alpha^{(t)}(s \mid x) = p_{u^{(t)}}(s \mid x) \quad \forall\, s \in K^n, \forall\, x \in \mathcal{T}$$

M-step maximise $L_B(u, \alpha^{(t)})$ w.r.t. u $\Rightarrow$

$$u^{(t+1)} \in \arg\max_u \frac{1}{m} \sum_{x \in \mathcal{T}} \sum_{s \in K^n} \alpha^{(t)}(s \mid x) \log p_u(x,s)$$

Let us analyse the M-step for HMMs. The objective is

$$\frac{1}{m} \sum_{x \in \mathcal{T}} \sum_{s \in K^n} \alpha^{(t)}(s \mid x) \langle \varphi(x,s), u \rangle - \log Z(u) \to \max_u$$

By denoting

$$\Psi = \frac{1}{m} \sum_{x \in T} \sum_{s \in K^n} \alpha^{(t)}(s \mid x)\, \varphi(x, s)$$

we get

$$\langle \Psi, u \rangle - \log Z(u) \to \max_u .$$

This is equivalent to the supervised learning task in Sec. 7. We know how to solve it, provided we can compute the components of Ψ.

Computing Ψ

For each $x \in T$ we want to compute

$$\Psi(x) = \sum_{s \in K^n} \alpha^{(t)}(s \mid x)\, \varphi(x, s) = \mathbb{E}_{p_{u^{(t)}}(s \mid x)} \big[\varphi(x, s) \big]$$

i.e. we have to compute the posterior pairwise marginals $p(s_{i-1}, s_i \mid x)$ $\forall\, i = 2, \dots, n$ and $s_{i-1}, s_i \in K$. This can be done by an algorithm similar to the one discussed in Sec. 5. The components of Ψ are then obtained by averaging the components of $\Psi(x)$ over all $x \in T$, i.e. $\Psi = \mathbb{E}_T[\Psi(x)]$.

Theorem 1 (w/o proof)

The sequence $L(u^{(t)})$ is monotonously increasing and the sequence $\alpha^{(t)}$ is convergent.

Remark 1 The EM algorithm for HMMs is known as Baum-Welch algorithm.