# Assignment 8

## 1 Structured Prediction [25 points]

Consider a part-of-speech (PoS) tagging task, in which instances $\mathbf{x}$ are sentences and labels $\mathbf{y}$ are sequences of PoS tags. For example, a labeled example for this problem might be

$$\begin{array}{rcccc} \mathbf{y}: & \mathsf{N} & \mathsf{V} & \mathsf{D} & \mathsf{N} \\ \mathbf{x}: & \text{John} & \text{threw} & \text{the} & \text{ball} \end{array}$$

Given several such labeled examples as training data, the goal is to learn a prediction model which given a new sentence, can accurately predict the sequence of PoS tags for the sentence.

Suppose we have a vocabulary of $M$ words and a set of $K$ possible PoS tags. Given a sentence-label pair $(\mathbf{x}, \mathbf{y})$, let's say that for each possible word $u$ and each possible PoS tag $k$, we define a joint feature $\phi_{u,k}(\mathbf{x}, \mathbf{y})$ that counts how many times $u$ appears in $\mathbf{x}$ together with $k$ in the corresponding position in $\mathbf{y}$. Thus if $\mathbf{x}$ and $\mathbf{y}$ are of length $T$, then denoting by $x_t$ the $t$-th word in $\mathbf{x}$ and by $y_t$ the $t$-th PoS tag in $\mathbf{y}$, we have

$$\phi_{u,k}(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{T} \mathbf{1}(x_t = u, y_t = k)\,.$$

Similarly, for each possble pair of words $(u, u')$ and each possible pair of PoS tags $(k, k')$, we define a joint feature $\phi_{u,u',k,k'}(\mathbf{x}, \mathbf{y})$ that counts how many times $(u, u')$ appears in consecutive positions in $\mathbf{x}$ together with $(k, k')$ in the corresponding positions in $\mathbf{y}$. Thus

$$\phi_{u,u',k,k'}(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{T-1} \mathbf{1}(x_t = u, x_{t+1} = u', y_t = k, y_{t+1} = k')\,.$$

Thus there are a total of $d = MK + M^2K^2$ features (note that different sentence-label pairs in the training set can be of different lengths $T$; any sentence-label pair is represented by the same number of features). For any given sentence-label pair $(\mathbf{x}, \mathbf{y})$, only a small number of these features are non-zero. We denote by $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y})$ the (sparse) $d$-dimensional joint feature vector derived from $(\mathbf{x}, \mathbf{y})$ as above.

Recall that, given a joint feature representation as above, both conditional random fields (CRFs) and StructSVM learn a weight vector $\mathbf{w} \in \mathbb{R}^d$, and given a new sentence $\mathbf{x}$ of length $T$, predict a label $\widehat{\mathbf{y}}$ (a PoS tag sequence of length $T$) as follows:

$$\widehat{\mathbf{y}} \;\in\; \arg\max_{\mathbf{y}} \; score_{\mathbf{w}}(\mathbf{x}, \mathbf{y})$$

where

$$\begin{aligned} score_{\mathbf{w}}(\mathbf{x}, \mathbf{y}) &= \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) \\ &= \sum_{u,k} w_{u,k} \cdot \phi_{u,k}(\mathbf{x}, \mathbf{y}) + \sum_{u,u',k,k'} w_{u,u',k,k'} \cdot \phi_{u,u',k,k'}(\mathbf{x}, \mathbf{y})\,. \end{aligned}$$

1. (10 points) Suppose you have learned a weight vector $\mathbf{w}$ as above (using either a CRF or StructSVM learning algorithm). You are given a new sentence $\mathbf{x}$ = "The dog chased the cat", and are considering two possible labelings for it:

$$\begin{array}{rccccc} \mathbf{y}^1: & \mathsf{N} & \mathsf{N} & \mathsf{V} & \mathsf{D} & \mathsf{N} \\ \mathbf{x}: & \text{The} & \text{dog} & \text{chased} & \text{the} & \text{cat} \end{array} \qquad \begin{array}{rccccc} \mathbf{y}^2: & \mathsf{D} & \mathsf{N} & \mathsf{V} & \mathsf{D} & \mathsf{N} \\ \mathbf{x}: & \text{The} & \text{dog} & \text{chased} & \text{the} & \text{cat} \end{array}$$

Thus the feature representations $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}^1)$ and $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}^2)$ have the following non-zero components (verify that you understand this):

$$\begin{aligned} \phi_{\text{the},\mathsf{N}}(\mathbf{x}, \mathbf{y}^1) &= 1 \qquad & \phi_{\text{the,dog},\mathsf{N},\mathsf{N}}(\mathbf{x}, \mathbf{y}^1) &= 1 \\ \phi_{\text{dog},\mathsf{N}}(\mathbf{x}, \mathbf{y}^1) &= 1 & \phi_{\text{dog,chased},\mathsf{N},\mathsf{V}}(\mathbf{x}, \mathbf{y}^1) &= 1 \\ \phi_{\text{chased},\mathsf{V}}(\mathbf{x}, \mathbf{y}^1) &= 1 & \phi_{\text{chased,the},\mathsf{V},\mathsf{D}}(\mathbf{x}, \mathbf{y}^1) &= 1 \\ \phi_{\text{the},\mathsf{D}}(\mathbf{x}, \mathbf{y}^1) &= 1 & \phi_{\text{the,cat},\mathsf{D},\mathsf{N}}(\mathbf{x}, \mathbf{y}^1) &= 1 \\ \phi_{\text{cat},\mathsf{N}}(\mathbf{x}, \mathbf{y}^1) &= 1 \end{aligned}$$

$$\begin{aligned} \phi_{\text{the},\mathsf{D}}(\mathbf{x}, \mathbf{y}^2) &= 2 \qquad & \phi_{\text{the,dog},\mathsf{D},\mathsf{N}}(\mathbf{x}, \mathbf{y}^2) &= 1 \\ \phi_{\text{dog},\mathsf{N}}(\mathbf{x}, \mathbf{y}^2) &= 1 & \phi_{\text{dog,chased},\mathsf{N},\mathsf{V}}(\mathbf{x}, \mathbf{y}^2) &= 1 \\ \phi_{\text{chased},\mathsf{V}}(\mathbf{x}, \mathbf{y}^2) &= 1 & \phi_{\text{chased,the},\mathsf{V},\mathsf{D}}(\mathbf{x}, \mathbf{y}^2) &= 1 \\ \phi_{\text{cat},\mathsf{N}}(\mathbf{x}, \mathbf{y}^2) &= 1 & \phi_{\text{the,cat},\mathsf{D},\mathsf{N}}(\mathbf{x}, \mathbf{y}^2) &= 1 \end{aligned}$$

Suppose the relevant components of the weight vector $\mathbf{w}$ you have learned are as follows:

$$\begin{aligned} w_{\text{the},\mathsf{N}} &= -6 \qquad & w_{\text{the,dog},\mathsf{N},\mathsf{N}} &= -2 \\ w_{\text{the},\mathsf{D}} &= 7 & w_{\text{the,dog},\mathsf{D},\mathsf{N}} &= 4 \\ w_{\text{dog},\mathsf{N}} &= 3 & w_{\text{dog,chased},\mathsf{N},\mathsf{V}} &= 0 \\ w_{\text{chased},\mathsf{V}} &= 2 & w_{\text{chased,the},\mathsf{V},\mathsf{D}} &= 3 \\ w_{\text{cat},\mathsf{N}} &= 4 & w_{\text{the,cat},\mathsf{D},\mathsf{N}} &= 4 \end{aligned}$$

Find $score_{\mathbf{w}}(\mathbf{x}, \mathbf{y}^1)$ and $score_{\mathbf{w}}(\mathbf{x}, \mathbf{y}^2)$. Show your calculations. Which of the two labelings receives a higher score under $\mathbf{w}$?

2. (15 points) Again, suppose you have learned a weight vector $\mathbf{w}$ as above (using either a CRF or StructSVM learning algorithm). In general, given a new sentence $\mathbf{x}$ of length $T$, there are $K^T$ possible labelings $\mathbf{y}$, and a brute-force search over all of them to find a labeling $\widehat{\mathbf{y}}$ with maximal score under $\mathbf{w}$ is expensive. Give an efficient algorithm to find a highest-scoring label sequence $\widehat{\mathbf{y}}$ under $\mathbf{w}$. Your algorithm should need only $O(K^2T)$ computations.

   *(Hint: Use a dynamic programming approach similar to the Viterbi algorithm for finding the most likely hidden state sequence in an HMM. The inputs to your problem are a sentence $\mathbf{x}$ of length $T$ and a weight vector $\mathbf{w} \in \mathbb{R}^d$ (with components indexed by $w_{u,k}$ and $w_{u,u',k,k'}$ as described above); the output is a predicted PoS tag sequence $\widehat{\mathbf{y}}$, also of length $T$, with maximal score under $\mathbf{w}$.)*

# 2 Semi-Supervised Learning [35 points]

In this problem you will consider a semi-supervised extension of Naïve Bayes that incorporates unlabeled data via EM. In particular, you will simulate one step of the EM algorithm on a small data set.

For simplicity, consider a binary classification task with 2-dimensional Boolean instances $\mathbf{x} \in \{0, 1\}^2$ and labels $y \in \{\pm 1\}$. Recall that the Naïve Bayes classifier assumes a generative probabilistic model of the form

$$p(\mathbf{x}, y) = p(y) \prod_{j=1}^{2} p(x_j \,|\, y)\,.$$

In our setting, there are 5 parameters, which we will collectively denote as $\boldsymbol{\theta}$:

$$\begin{aligned}\theta_{+1} &= \mathbf{P}(Y = +1)\,;\\ \theta_{j|k} &= \mathbf{P}(X_j = 1 \,|\, Y = k)\,, \quad \text{for each feature } j \in \{1,2\} \text{ and each label } k \in \{\pm 1\}\,.\end{aligned}$$

Thus the probability of a labeled example $\mathbf{x} = (1,0), y = -1$ under the above model would be

$$\mathbf{P}(Y=-1)\,\mathbf{P}(X_1 = 1 \,|\, Y=-1)\,\mathbf{P}(X_2 = 0 \,|\, Y=-1) = (1-\theta_{+1})\theta_{1|-1}(1-\theta_{2|-1})\,.$$

**Standard (Supervised) Naïve Bayes.** In the supervised setting, given labeled training data $S_L = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{m_L}, y_{m_L}))$, one computes maximum likelihood estimates as follows:

$$\begin{aligned}\widehat{\theta}_{+1} &= \frac{1}{m_L}\sum_{i=1}^{m_L} \mathbf{1}(y_i = +1)\\ \widehat{\theta}_{j|k} &= \frac{\sum_{i=1}^{m_L} \mathbf{1}(y_i = k, x_{ij} = 1)}{\sum_{i=1}^{m_L} \mathbf{1}(y_i = k)}\end{aligned}$$

Given a new instance $\mathbf{x}$, one then uses the estimated parameters together with Bayes rule to compute the probability $P(Y = +1 \,|\, \mathbf{x})$ under the learned model, and classifies the instance as $+1$ if this probability is greater than $\frac{1}{2}$ and as $-1$ otherwise.

**Semi-Supervised Naïve Bayes.** In the semi-supervised setting, given labeled training data $S_L$ as above and additional unlabeled data $S_U = (\mathbf{x}_{m_L+1}, \ldots, \mathbf{x}_{m_L+m_U})$, one treats the missing labels in $S_U$ as unobserved variables and uses EM to find maximum likelihood parameter estimates. In particular, one starts with initial parameter estimates $\widehat{\boldsymbol{\theta}}^0$ learned as above from the labeled data $S_L$ alone, and then iteratively estimates posterior distributions on the missing labels of the unlabeled data points (E-step) and updates the parameter estimates via a weighted maximum likelihood estimation (M-step). Specifically, on each round $t$, in the E-step, for each unlabeled example $\mathbf{x}_i$ one computes the posterior distribution over labels, $q^t(k \,|\, \mathbf{x}_i) = P(Y_i = k \,|\, \mathbf{x}_i; \widehat{\boldsymbol{\theta}}^t)$, under the current parameter estimates $\widehat{\boldsymbol{\theta}}^t$; in the M-step, one then updates the parameter estimates as follows:

$$\begin{aligned}\widehat{\theta}^{t+1}_{+1} &= \frac{1}{m_L + m_U}\left(\sum_{i=1}^{m_L} \mathbf{1}(y_i = +1) + \sum_{i=m_L+1}^{m_L+m_U} q^t(+1 \,|\, \mathbf{x}_i)\right)\\ \widehat{\theta}^{t+1}_{j|k} &= \frac{\sum_{i=1}^{m_L} \mathbf{1}(y_i = k, x_{ij} = 1) + \sum_{i=m_L+1}^{m_L+m_U} q^t(k \,|\, \mathbf{x}_i)\cdot \mathbf{1}(x_{ij} = 1)}{\sum_{i=1}^{m_L} \mathbf{1}(y_i = k) + \sum_{i=m_L+1}^{m_L+m_U} q^t(k \,|\, \mathbf{x}_i)}\end{aligned}$$

On convergence, one uses the final parameter estimates to make predictions on new instances in the same manner as before.

**Problem.** Suppose you have a training sample of 8 labeled examples and 4 unlabeled examples, distributed as follows (note that since there are only 4 possible instances in our simple setup and 2 possible labels, some examples are repeated in the training sample below; this would be unlikely in real data, but the key ideas present in this example would carry over to real data as well):

| **Labeled data $S_L$** | **Unlabeled data $S_U$** |
|---|---|
| $\mathbf{x} = (x_1, x_2), y$ | $\mathbf{x} = (x_1, x_2)$ |
| $(1,1), +1$ | $(1,1)$ |
| $(1,1), +1$ | $(1,1)$ |
| $(1,0), +1$ | $(0,0)$ |
| $(0,0), +1$ | $(0,0)$ |
| $(1,0), -1$ | |
| $(0,1), -1$ | |
| $(0,0), -1$ | |
| $(0,0), -1$ | |
| 8 | 4 |

1. (5 points) Calculate the initial maximum likelihood parameter estimates based on the labeled data only: $\widehat{\boldsymbol{\theta}}^0 = (\widehat{\theta}^0_{+1}, \widehat{\theta}^0_{1|+1}, \widehat{\theta}^0_{2|+1}, \widehat{\theta}^0_{1|-1}, \widehat{\theta}^0_{2|-1})$.

2. (8 points) For each instance $\mathbf{x} = (x_1, x_2)$ that appears in the unlabeled data, compute the posterior distribution over the label under the parameter estimates computed in the first part above. In particular, compute each of the following:

$$
\begin{aligned}
q^0(+1\,|\,\mathbf{x} = (1,1)) &= P(Y = +1\,|\,\mathbf{x} = (1,1); \widehat{\boldsymbol{\theta}}^0) \\
q^0(+1\,|\,\mathbf{x} = (0,0)) &= P(Y = +1\,|\,\mathbf{x} = (0,0); \widehat{\boldsymbol{\theta}}^0)
\end{aligned}
$$

   Show your calculations.

   *(Hint: Use Bayes' rule.)*

3. (10 points) Using the results of the first two parts above, find the updated parameter estimates after one step of EM: $\widehat{\boldsymbol{\theta}}^1 = (\widehat{\theta}^1_{+1}, \widehat{\theta}^1_{1|+1}, \widehat{\theta}^1_{2|+1}, \widehat{\theta}^1_{1|-1}, \widehat{\theta}^1_{2|-1})$. Show your calculations.

4. (6 points) The log-likelihood of the labeled and unlabeled data $S = (S_L, S_U)$ under parameter estimates $\widehat{\boldsymbol{\theta}}^t$ is given by

$$
\ln p(S; \widehat{\boldsymbol{\theta}}^t) \;=\; \sum_{i=1}^{m_L} \ln p(\mathbf{x}_i, y_i; \widehat{\boldsymbol{\theta}}^t) + \sum_{i=m_L+1}^{m_L+m_U} \underbrace{\ln \left( \sum_{y_i \in \{\pm 1\}} p(\mathbf{x}_i, y_i; \widehat{\boldsymbol{\theta}}^t) \right)}_{\ln p(\mathbf{x}_i; \widehat{\boldsymbol{\theta}}^t)} .
$$

   Write an expression for the log-likelihood of the given training data as a function of the 5 parameters $\widehat{\theta}^t_{+1}, \widehat{\theta}^t_{1|+1}, \widehat{\theta}^t_{2|+1}, \widehat{\theta}^t_{1|-1}, \widehat{\theta}^t_{2|-1}$ for any $t$.

5. (6 points) Using the expression derived in the fourth part above, calculate the log-likelihood of the given training data under both the initial parameter estimates $\widehat{\boldsymbol{\theta}}^0$ computed in the first part above and the updated parameter estimates $\widehat{\boldsymbol{\theta}}^1$ computed in the third part above. After one step of EM, has the log-likelihood of the data increased or decreased?

   *(Hint: You might like to write a small script which given the 5 parameters as input, returns the log-likelihood value, according to the expression you derived in the fourth part above, as output. You can then plug in the values of the parameter estimates obtained above for $t = 0$ and $t = 1$ into your script to obtain the corresponding log-likelihoods.)*