Inference and Representation, Fall 2017

Structured Prediction for Part-of-Speech Tagging, MLE and Max-Ent, Max-Sum BP

Zhuoru Lin
zlin@nyu.edu

Disclaimer: I adhered to NYU honor code in this assignment.

1. **Part-of-speech tagging using SSVM**
   (a)
   (b)
   (c)
   (d)

2. **MaxSum**
   We can leverage the fglib library we used before. First we define the factor graph by the following codes. Notice how we can set the node potential by defining a prior factor that contains only one nodes. The edge potential is set by the factor that contains two nodes. The result of MaxSum algorithm returns is (0,1,1,1,1,1)

```
fg = graphs.FactorGraph()

###### Construct Graph
x1 = nodes.VNode('X1', rv.Discrete)
x2 = nodes.VNode('X2', rv.Discrete)
x3 = nodes.VNode('X3', rv.Discrete)
x4 = nodes.VNode('X4', rv.Discrete)
x5 = nodes.VNode('X5', rv.Discrete)
x6 = nodes.VNode('X6', rv.Discrete)
# Priors
f1 = nodes.FNode('f1', rv.Discrete([2,1], x1))
f2 = nodes.FNode('f1', rv.Discrete([2,1], x1))
f3 = nodes.FNode('f1', rv.Discrete([2,1], x1))
f4 = nodes.FNode('f1', rv.Discrete([2,1], x1))
f5 = nodes.FNode('f1', rv.Discrete([2,1], x1))
f6 = nodes.FNode('f1', rv.Discrete([2,1], x1))
# Factors
f12 = nodes.FNode('f12', rv.Discrete([[1,2],[3,4]], x1, x2))
f13 = nodes.FNode('f13', rv.Discrete([[1,2],[3,4]], x1, x3))
f14 = nodes.FNode('f14', rv.Discrete([[1,2],[3,4]], x1, x4))
f25 = nodes.FNode('f25', rv.Discrete([[1,2],[3,4]], x2, x5))
f26 = nodes.FNode('f26', rv.Discrete([[1,2],[3,4]], x2, x6))
# Register nodes
fg.set_nodes([x1,x2,x3,x4,x5,x6])
fg.set_nodes([f1,f2,f3,f4,f5,f6,f12,f13,f14,f25,f26])
# Edges
fg.set_edge(f1, x1)
fg.set_edge(f2, x2)
fg.set_edge(f3, x3)
fg.set_edge(f4, x4)
fg.set_edge(f5, x5)
fg.set_edge(f6, x6)
fg.set_edge(x1, f12)
fg.set_edge(f12, x2)
fg.set_edge(x1, f13)
fg.set_edge(f13, x3)
fg.set_edge(x1, f14)
fg.set_edge(f14, x4)
fg.set_edge(x2, f25)
fg.set_edge(f25, x5)
fg.set_edge(x2, f26)
fg.set_edge(f26, x6)
```

3. **Exponential families**
The log-likelihood of the data is:

$$\log[p(x,\theta)] = \sum_{n=1}^{L}(\langle\theta, f(x^{(n)})\rangle - \log[Z(\theta)]) \tag{1}$$

The maximum likelihood estimation θ_{ML} must satisfied:

$$\frac{\partial p}{\partial \theta}(\theta_{ML}) = 0 \tag{2}$$

$$\sum_{n=1}^{L} f(x^{(n)}) - L\sum_{x}\frac{\exp[<\theta_{ML}, f(x^n)>]}{Z(\theta_{ML})}f(x^n) = 0 \tag{3}$$

$$\frac{1}{L}\sum_{n=1}^{L} f(x^{(n)}) = \sum_{x} p(x|\theta_{ML})f(x^n) \tag{4}$$

Technically, we also need to show that this is in fact an maximum by showing the secon-derivative. This is ignored here.

4. **Maximun entropy distribution** The maximum entropy distribution can be formulated as an functinal optimization problem as:

$$\begin{aligned} \arg\max_p \int_x \log(p(x))p(x)dx \\ \text{with} \int_x p(x)dx = 1 \\ \text{and} \quad \int p(x)f_k(x) = \alpha_k \text{ for all k} \end{aligned}$$

The Lagrangian is:

$$\mathcal{L} = \int_x \log(p(x))p(x)dx - \sum_k (\theta_k \int p(x)f_k(x) - \alpha_k) - (\lambda \int p(x)dx - 1). \tag{5}$$

The Lagrange multiplier states that the optimum acheives zero derivatives of Lagragian. Taking the functional derivative, we have:

$$\log(p(x)) + 1 - \lambda - \sum_k \theta_k f_k(x) = 0 \tag{6}$$

Therefore we must have:

$$\begin{aligned} p(x) &= \exp(\sum_k \theta_k f_k(x) + \lambda - 1) && (7) \\ &= \frac{1}{Z}\exp(\langle \theta, f(x) \rangle) && (8) \end{aligned}$$

with $Z = \exp(1 - \lambda)$