Natural Language Processing
Lecture 4

Lecture by Dr. Kfir Bar
Typeset by Steven Karas

2019-04-02
Last edited 20:58:30 2019-04-02

Disclaimer These notes are based on the lectures for the course Natural Language Processing, taught by Dr. Kfir Bar at IDC Herzliyah in the spring semester of 2018/2019. Sections may be based on the lecture slides prepared by Dr. Kfir Bar.

1 Agenda

- POS tagging
- Intro to machine learning

2 Part of speech tagging

Given a sentence, we want to tag the part of speech for each token.

2.1 Named Entity Recognition

Marks named entities, such as people, places, and organizations. Some will use geopolitical entities to resolve the ambiguities between places and organizations (e.g. Israel).

Unfortunately, HMM lacks necessary features to correctly handle this task.

As an extension of this, structured information retrieval attempts to extract structured data from unstructured text. For example, "Trump met with Kim Jong Un on Tuesday".

2.1.1 IOB notation

- Inside - tags that are inside a sequence
- Other - any other tags
- Begin - the first tag in a sequence

2.2 Shallow parsing/chunking

More complicated than part of speech tagging, this also splits text into phrases.

2.3 Hidden Markov Models

Consider observations into the model, such that given a sequence of observations o we model how they map to a sequence of states s:

$$o = (o_1, \ldots, o_T)$$
$$s = (s_1, \ldots, s_T)$$

where the probability of seeing any particular state given an observation:

$$\Pr[s \mid o] = \frac{\Pr[o \mid s] \Pr[s]}{\Pr[o]} = \frac{\Pr[o \mid s] \Pr[s]}{\sum_{s'} \Pr[s', o]}$$

Note that the denominator is constant with regards to s, so we can ignore it. To maximize the probability for sequences observed in a corpus:

$$\max_{A,B} \frac{\Pr[o \mid s; A, B] \Pr[s; A, B]}{\sum_{s'} \Pr[s', o; A, B]}$$

In order to find the most likely state $\hat{s}$ given an observation o:

$$\begin{aligned}\hat{s} &= \arg\max_s \Pr[s \mid o] \\ &= \arg\max_s \Pr[o \mid s] \Pr[s] \\ &= \arg\max_s \prod_{t=1}^{T} B_{s_t o_t} \cdot A_{s_{t-1} s_t}\end{aligned}$$

2.3.1 HMM Inference

$$\Pr[s, o] = \Pr[o \mid s; A, B] \Pr[s; A, B]$$

$$= \prod_{t=1}^{T} \Pr[s_t \mid s_{t-1}] \Pr[o_t \mid s_t] = \prod_{t=1}^{T} A_{s_{t-1} s_t} B_{s_t o_t}$$

2.3.2 Viterbi decoding

The naive approach of finding the state sequence that maximizes the probabilities of the given observations takes exponential time. We use a dynamic programming approach instead: [1]

$$\begin{aligned}\delta_j(t) &= \max_{s_1,\ldots,s_{t-1}} \Pr[s_1 \ldots s_{t-1}, o_1 \ldots o_{t-1}, s_t = j, o_t] \\ \delta_j(t+1) &= \max_i \delta_i(t) a_{ij} b_{j o_{t+1}} \\ \delta_j(1) &= \pi_j b_{j o_1} \\ \psi_j(t+1) &= \arg\max_i \delta_i(t) a_{ij} b_{j o_{t+1}}\end{aligned}$$

Viterbi decoding

```
Viterbi(O[1, T], M=(A,B,π)) // O = observations, M = model
N = # of states
// Initialization Step
for n = 1 to N:
  δ(n, 1) = B[n, O[1]] * π[n]
  ψ(n, 1) = 0
// Iteration Step
for t = 2 to T:
  for n = 1 to N:
    δ(n, t) = B[n, O[t]] * max_{j=1,...,T} (δ(j, t - 1) * A[j, n])
    ψ(n, t) = index of j that gave the max above
// Sequence recovery
Seq(T) = n that maximizes δ(n,T)
for t = T - 1 downto 1:
  Seq(t) = ψ(Seq(t+1), t+1)
```

2.3.3 Training

Supervised learning requires tagged examples, but is just MLE.

Unsupervised use of the model is possible, by using a lexicon and has several approaches (which approximate clustering or MLE).

[1] The slides use x in place of s'.

2.3.4 Generalization

HMM inference is defined as:

$$\Pr[s, o] = \Pr[s] \Pr[o \mid s] = \prod_{t=1}^{T} \Pr[s_t \mid s_{t-1}] \Pr[o_t \mid s_t]$$

We can rewrite this as:

$$\begin{aligned}\Pr[s, o] &= \prod_{t=1}^{T} \exp\{\log \Pr[s_t \mid s_{t-1}] + \log \Pr[o_t \mid s_t]\} \\ &= \prod_{t=1}^{T} \exp \left\{ \begin{array}{lll} \sum\limits_{i,j \in S} & \log \Pr[i \mid j] & 1_{s_t=i} 1_{s_{t-1}=j} \\ + \sum\limits_{i \in S, o \in O} & \log \Pr[o \mid i] & 1_{o_t=o} 1_{s_t=i} \end{array} \right\}\end{aligned}$$

And then abstract the parameters and replace them with generic features where $\theta_{i,j} = \log \Pr[i \mid j]$ and $\mu_{o,i} = \log \Pr[o \mid i]$:

$$\begin{aligned}\Pr[s, o] &= \prod_{t=1}^{T} \exp \left\{ \begin{array}{lll} \sum\limits_{i,j \in S} & \theta & 1_{s_t=i} 1_{s_{t-1}=j} \\ + \sum\limits_{i \in S, o \in O} & \mu_{o,i} & 1_{o_t=o} 1_{s_t=i} \end{array} \right\} \\ &= \prod_{t=1}^{T} \exp \left\{ \sum_k \theta_k f_k(s_t, s_{t-1}, o_t) \right\}\end{aligned}$$

Where f is the feature vector. This is commonly called a log linear model.

Feature Functions There are many different features we can add. Here are some examples:

- Starts with capital letter
- All caps
- previous POS

2.4 Maximum Entropy Classifier

Also known as multinomial logistic regression or softmax classifier. Given a data point represented as a n-dimensional vector, we want to predict its class out of a closed set C.

$$\hat{c} = \arg\max_{c \in C} \Pr[c \mid x] = \frac{\exp\left(\sum\limits_{i=0}^{n} \theta_{ci} f_i\right)}{\sum_{c' \in C} \exp\left(\sum\limits_{i=0}^{n} \theta_{c'i} f_i\right)}$$

To train the model parameters θ, we find the model that maximizes:

$$\begin{aligned}\prod_{i=1}^{m} &= \Pr[x^{(i)} \mid c^{(i)}] \\ \log \prod_{i=1}^{m} \Pr[x^{(i)} \mid c^{(i)}] &= \sum_{i=1}^{m} \log \Pr[x^{(i)} \mid c^{(i)}]\end{aligned}$$

And then apply regularization to avoid overfitting:

$$\sum_{i=1}^{m} \sum_{j=1}^{n_j} \log \Pr[x_j^{(i)} \mid c_j^{(i)}] - \lambda \sum_{i,j} \theta_{ij}^2$$

2.5 Maximum Entropy Markov Model

We can mix the HMM with the ME approach to get:

$$\hat{s} = \arg\max_s \Pr[s \mid o] = \arg\max_s \prod_{t=1}^{T} \Pr[s_t \mid s_{t-1}, o_t]$$

Which is trained by:

$$\Pr[s_t \mid s_{t-1}, o_t] = \frac{\exp\{\theta^T \cdot f(s_{1\ldots t-1}, s_t, o, t)\}}{\sum_{s'} \exp\{\theta^T \cdot f(s_{1\ldots t-1}, s', o, t)}$$

I didn't have enough time to scribe the rest of the slides.

2.6 Model evaluation

- Leave out test set
- k-fold cross validation[2]
- leave-one-out-cross-validation (LOOCV)

POS tagging usually uses the Penn Wall Street Journal treebank.

3 Next Lecture

A makeup lecture will be held on Friday, April 5th at 845am. I will not be attending. The lecture next week will not be held due to national elections. The lecture the two weeks after that will not be held due to holidays.

References

[1] Yoav Goldberg. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726, 2015.

[2] Daniel Jurafsky and James H. Martin. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., 2009.

[3] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[2] This doesn't work well when the training process takes a very long time, because it multiplies the training process by k