CSE 517 HW2

Zhanlin Liu

Febuary 15, 2019

1 Bigram HMM

1.1 OOV cases

Since we are analyzing the twitter data, there are several cases require more attentions. For example, the existence of "#" might indicate the appearance of hash tags. In addition, the existence of "@" might indicate refer to a person. "http" might represent a website link, etc.. Based on these special properties, the low frequent words are classified in these classes. To infer how each rule would improve the result, dev set is used to measure the accuracy for each scenario. In this case, the linear interpolation coefficients are set as 0.4 and 0.6 as two constants.

Table 1: OOV rules for accuracy.

Rule	train	dev
"#" and others	0.824	0.817
"#", "@", "http", and others	0.837	0.830
"#", "@", "http", number, and others	0.862	0.837

Performance on the test set has accuracy of 84.5%.

1.2 Error analysis

Based on the confusion matrix and the wrong combo list, most of the errors exist in state "∧", "," and "!". Especially "∧", 869 cases of "∧" are predicted as N. This might due to the reason that there is not enough emission words for ∧. The confusion matrix is provided in the Jupyter notebook.

2 Trigram HMM

2.1 Joint probability model

$$\begin{aligned} p(s,y) &= p(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_{n+1}) \\ &= q(STOP|y_n, y_{n-1}) \prod_{i=1}^{n} q(y_i|y_{i-1}, y_{i-2}) e(x_i|y_i) \end{aligned} \tag{1}$$

$$\begin{aligned} \pi(i, y_i, y_{i-1}) &= \max_{y_1 \cdots, y_{i-1}} p(x_1 \cdots x_i, y_1 \cdots y_i) \\ &= \max_{y_{i-2}, y_{i-1}} e(x_i|y_i) q(y_i|y_{i-2} y_{i-1}) \max_{y_1 \cdots, y_{i-2}} p(x_1 \cdots x_i, y_1 \cdots y_{i-2}) \\ &= \max_{y_{i-2}, y_{i-1}} e(x_i|y_i) q(y_i|y_{i-2} y_{i-1}) \pi(i, y_{i-1}, y_{i-2}) \end{aligned} \tag{2}$$

2.2 Viterbi algorithm

2.3 Model Performance

The codes are provided in the Jupyternotebook.

Rule and λ	train	dev
"#" and others (0.4, 0.3, 0.3)	0.867	0.823
"#", "@", "http", and others (0.4, 0.3, 0.3)	0.888	0.842
"#", "@", "http", number, and others (0.4, 0.3, 0.3)	0.900	0.845
"#" and others (0.7, 0.2, 0.1)	0.872	0.832
"#", "@", "http", and others (0.7, 0.2, 0.1)	0.891	0.843
"#", "@", "http", number, and others (0.7, 0.2, 0.1)	0.901	0.846

Choose the last parameter combination, the accuracy for the test set is around 85.2%. It did not have significant a improvement comparing with the bigram model.

3 Unsupervised training with EM

3.1 Drive formula

$$\text{(expected) } count(y_{i-1}y_i) = \sum_i p(y_{i-1}y_i|x_1 \cdots x_n x_n)$$

Maximum likelihood parameters

$$q_{ML}(y_{i-1}y_i) = \frac{c(y_{i-2}y_{i-1}y_i}{y_{i-2}}, e_{ML(x|y)} = \frac{c(y,x)}{c(y)}$$

4 PCFG Language Models

I do not think $P_{tree}(x)$ is superior than $P_{seq}(x)$ with the following reasons:

- The rules of PCFG model is trained based on general document. Therefore, there might exist some biases on a particular document of analysis. Even though we can construct specific rules for the PCFG model for the specific kind of document. It might be unrealistic in practic. However, training a sequence model for specific document is light and realistic in practice.
- Another reason is PCFG focuses more on grammatical structures. It might prefer selecting sentences with a better grammatical structure. However, sequence model can detect the hidden meanings of the sentences.

5 Playing with Off-the-shelf Parsers

The parser used in this section is "http://nlp.stanford.edu:8080/parser/".

5.1 Wrong examples

- "I cleaned a bug in the office." This is incorrect, since this sentence tries to express "I"- the person in the office instead of the bug in the office. Parser used: "http://nlp.stanford.edu:8080/parser/"
- "I watched stars falling with my friends." This is incorrect since my friends are with me not falling with the stars.

5.2 Dependency examples

- "I watched a weather report saying there will be a storm in the morning." In the morning depends on I instead of the storm.
- "I dreamed I have graduated in my dorm." In my dorm should depend on dreamed instead of graduated.

6 Variations to CockeYoungerKasami (CKY) Algorithm

Input: A sentence $s = x_1 x_2 \dots x_n$, a $PCFG = (N, \Sigma, S, R, \epsilon)$
Initialization, for all $i \in \{1, 2, \dots, n\}$, for all $X \in N$:

$$\pi(i, i, X) = \begin{cases} q(X \to x_i, X \to x_i \in R \\ 0, otherwise \end{cases} \tag{3}$$

Algorithm:

For $l = 1, 2, \dots, n-1$:
For $l = 1, 2, \dots, n-l$:
Set $j = i + l$
For all $X \in N$ Calculate

$$\pi(i, j, X) = \max_{X \to VYZ \in R, s_1 \in \{i, \dots, j-2\}, s_2 \in \{i+1, \dots, j-1\}} \tag{4}$$
$$(q(X \to VYZ)\pi(i, s_1, Y)\pi(s_1 + l, s_2)\pi(s_2 + l, j, Z))$$
$$bp(i, j, X) = \max_{X \to VYZ \in R, s_1 \in \{i, \dots, j-2\}, s_2 \in \{i+1, \dots, j-1\}}$$
$$(q(X \to VYZ)\pi(i, s_1, Y)\pi(s_1 + l, s_2)\pi(s_2 + l, j, Z))$$

Return $\pi(1, n, S) = \max_{t \in T(\cdot)} P(t)$, and pack pointers bq which allow recovery of $\max_{t \in T(\cdot)} P(t)$.