ACS Intro to Natural Language Syntax and Parsing

Lecture 3
Part I: Language Modelling and Smoothing
Part II: Viterbi for Taggers

Stephen Clark

Natural Language and Information Processing (NLIP) Group

sc609@cam.ac.uk

Language Modelling

- A **language model** is a probability distribution over strings
- A core component of many language processing systems
  - speech recognition, hand writing recognition, machine translation, ...
- Many of the estimation techniques carry over to other NLP problems
  - POS tagging, parsing, ...

Classic N-gram Language Models

$$
\begin{aligned}
P(w_1, \ldots, w_n) &= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \ldots P(w_n|w_1, \ldots, w_{n-1}) \\
&\approx P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \ldots P(w_n|w_{n-2}, w_{n-1})
\end{aligned}
$$

- Chain rule followed by independence assumptions
  - here the independence assumptions give a **trigram** language model
- Notice chain rule is *exact*
- Note also that we don't have to apply the chain rule using this ordering of the words

Statistical Methods in NLP: Chomsky

It must be recognised that the notion of a 'probability of a sentence' is an entirely useless one, under any interpretation of this term (Chomsky, 1969)

[taken from Chapter 1 of Young and Bloothooft, eds, *Corpus-Based Methods in Language and Speech Processing*]

[Colorless green ideas sleep furiously example]

It's all about *Prediction*

Statistics is the science of learning from observations and experience (Ney, 1997)

By chance *we mean something like a guess. Why do we make guesses? We make guesses when we wish to make a judgement but have incomplete information or uncertain knowledge. We want to make a guess as to what things are, or what things are likely to happen. Often we wish to make a guess because we have to make a decision ... Sometimes we make guesses because we wish, with our limited knowledge, to say as much as we can about some situation. Really, any generalization is in the nature of a guess. Any physical theory is a kind of guesswork. There are good guesses and there are bad guesses. The theory of probability is a system for making better guesses. The language of probability allows us to speak quantitatively about some situation which may be highly variable, but which does have some consistent average behavior.* (Guess who?)

[taken from Chapter 1 of Young and Bloothooft, eds, *Corpus-Based Methods in Language and Speech Processing*]

Overfitting

- Maximum likelihood estimation (relative frequency estimation in our cases) can suffer from *overfitting*
- Because we are choosing parameters to make the data as probable as possible, the resulting model can look “too much” like the data
- Classic example of this occurs in language modelling with a word we haven’t seen before

The Problem with Zeros

$$\hat{P}(w_3|w_2, w_1) = \frac{f(w_1, w_2, w_3)}{f(w_1, w_2)}$$

- If w_1 followed by w_2 is unseen then estimate is undefined
- If w_1 followed by w_2 followed by w_3 is unseen then estimate is zero
- Zeros propogate through the product to give a zero probability for the whole string
- *Smoothing* techniques are designed to solve this problem (called *smoothing* because the resulting distributions tend to be more uniform)

Add-one Smoothing

$$\hat{P}(w_i|w_{i-1}) = \frac{1 + f(w_{i-1}, w_i)}{\Sigma_{w_i} 1 + f(w_{i-1}, w_i)}$$
$$= \frac{1 + f(w_{i-1}, w_i)}{|V| + \Sigma_{w_i} f(w_{i-1}, w_i)}$$

- Simple technique not widely used anymore

Linear Interpolation

- Consider *burnish the* and *burnish thou* where:

$$f(\textit{burnish, the}) = f(\textit{burnish, thou}) = 0$$

- add-1 smoothing would assign the same probability to $P(\textit{the}|\textit{burnish})$ and $P(\textit{thou}|\textit{burnish})$
- But *burnish the* intuitively more likely (because *the* more likely than *thou*)

Linear Interpolation

$$\tilde{P}(w_i|w_{i-1}) = \lambda \hat{P}(w_i|w_{i-1}) + (1-\lambda)\hat{P}(w_i)$$

- $\tilde{P}$ is the interpolated model and $\hat{P}$ is the maximum likelihood (relative frequency) estimate

$$\tilde{P}(w_i|w_{i-1}, w_{i-2}) = \lambda_1 \hat{P}(w_i|w_{i-1}, w_{i-2})$$
$$+(1-\lambda_1)(\lambda_2 \hat{P}(w_i|w_{i-1}) + (1-\lambda_2)\hat{P}(w_i))$$

Backoff

$$
\begin{aligned}
\tilde{P}(w_i|w_{i-1}, w_{i-2}) &= \alpha_1 \hat{P}(w_i|w_{i-1}, w_{i-2}) && \text{if } f(w_{i-2}, w_{i-1}, w_i) > 0 \\
&= \alpha_2 \hat{P}(w_i|w_{i-1}) && \text{if } f(w_{i-2}, w_{i-1}, w_i) = 0 \\
& && \text{and } f(w_{i-1}, w_i) > 0 \\
&= \alpha_3 \hat{P}(w_i) && \text{otherwise}
\end{aligned}
$$

- where the αs are required to ensure a proper distribution
- although see *Large Language Models in Machine Translation*, Brants et al., based on **2 trillion tokens** which uses "stupid backoff" and ignores the αs

[see google n-gram corpus]

Back to Tagging ...

- Tag sequence probabilities can be smoothed (or backed off):

$$\tilde{P}(t_i|t_{i-1}, t_{i-2}) = \lambda_1 \hat{P}(t_i|t_{i-1}, t_{i-2})$$
$$+(1-\lambda_1)(\lambda_2 \hat{P}(t_i|t_{i-1}) + (1-\lambda_2)\hat{P}(t_i))$$

- A simple solution for unknown words is to replace them with UNK:

$$P(w_i|t_i) = P(\mathsf{UNK}|t_i)$$

where any word in the training data occurring less than, say, 5 times is replaced with UNK

Better Handling of Unknown Words

- Lots of clues as to what the tag of an unknown word might be:
  - proper nouns (NNP) likely to be unknown
  - if the word ends in *ing*, likely to be VBG
  - . . .

$$P(w|t) = \frac{1}{Z} P(\text{unknown word}|t) P(\text{capitalized}|t) P(\text{endings}|t)$$

- but now we're starting to see the weaknesses of generative models for taggers
- *Conditional* models can deal with these features directly
  - see Ratnaparkhi's tagger as an example

Tagging: Why is there a search problem?

$$T^* = \arg\max_T P(T|W) = \arg\max_T P(W|T)P(T)$$

- Number of tag sequences for a sentence of length n is $O(T^n)$ where T is the size of the tagset
- OK, but why is there a non-trivial search problem?
  - e.g. for a unigram model we can just take the most probable tag for each word, an algorithm which runs in $O(nT)$ time

Tagging: Why is there a non-trivial search problem?

$$T^* = \arg\max_T P(T|W) = \arg\max_T P(W|T)P(T)$$

- But what about a bigram model?
- Intuition: suppose I have two competing tags for word w_i, t_i^1 and t_i^2
- Compare:

$$\text{Score}(t_i^1) = P(t_i^1|t_{i-1})P(w_i|t_i^1)$$
$$\text{Score}(t_i^2) = P(t_i^2|t_{i-1})P(w_i|t_i^2)$$

Suppose $\text{Score}(t_i^1) > \text{Score}(t_i^2)$; can we be sure t_i^1 is part of the highest scoring tag *sequence*?

Viterbi Algorithm

- Dynamic Programming (DP) algorithm, so requires the "optimal sub-problem property"
  - i.e. optimal solution to the complete problem can be defined in terms of optimal solutions to sub-problems
- So what are the sub-problems in this case?
  - intuition: suppose we want the optimal tag sequence ending at w_n, and we know the optimal sub-sequence ending at w_{n-1}, for all possible tags at w_{n-1}

Viterbi Algorithm (for a bigram tagger)

$$\delta_{t_j}(n+1) = \max_{t_i} \delta_{t_i}(n) P(t_j|t_i) P(w_{n+1}|t_j)$$

where $\delta_{t_j}(n+1)$ is the probability of the most probable tag sequence ending in tag t_j at position $n+1$

- Recursion bottoms out at position 1 in the sentence
- Most probable tag sequence can be obtained by following the recursion from the right backwards
- Time complexity is $O(T^2 n)$ where T is the size of the tagset

[See Chs. 9 and 10 of Manning and Schutze for a more rigorous presentation]

Practicalities

- Choice of tags to be assigned to a particular word usually governed by a “tag dictionary”
- Accuracy measured by taking a manually created “gold-standard” for a set of held-out test sentences
- Accuracy of POS taggers on newspaper data is over 97%, close to the upper bound represented by human agreement (and existence of noise in the data)
- Linear time process (in length of sentence) means tagging can be performed very fast, e.g. hundreds of thousands of words per second

Reading for Today's Lecture

- Jurafsky and Martin, Speech and Language Processing, Chapter on N-grams
- Manning and Schutze, Foundations of Statistical Natural Language Processing, Chapter on Statistical Inference: n-gram models over sparse data
- An Empirical Study of Smoothing Techniques for Language Modeling, Chen and Goodman, TR-10-98, 1998 (*lots of technical and empirical details*)
- Chapter 1 of Corpus-Based Methods in Language and Speech Processing, ed Steve Young and Gerrit Bloothooft, Kluwer 1997 (*if available*)
- Chapters 9 and 10 of Manning and Schutze