**Language modelling**

1. **Noisy channel model**:

Applications to machine translation, spelling correction, etc. 看一下计算题例子P42

1. **Estimating probabilities by counting**: P(w)
2. **n-gram language models:**

**n-gram with approximation:**

(look back only m-1 words)

1. **why performs smoothing?**

Solve out-of-vocabulary problem.

1. **Back-off and interpolation**.
2. Back-off:

Simple solution: **if =0, use .**

However, now we do not have a true probability distribution and overestimates some unlikely n-grams, so, we must discount the counts.

Where is the probability mass of seen examples and

1. Interpolation

We simply interpolate the unigram probability into the bigram probability as follows:

Generalizes to higher n-grams as:

1. Perplexity to evaluate language models.

The smaller the PP value, the better the language model.

**Syntax 1**

1. **Part of Speech Tagging: Hidden Markov Model**
2. **Three fundamental problems for HMM**
3. What is the probability of a sequence given an observation and a model?
4. What is P(DET N VBZ DET N| the cat chases the mouse, μ)
5. What is the probability of an observation given the model?
6. What is P(the cat chases the mouse| μ)
7. What is the model that maximizes the likelihood of the observed data and the known sequences?
8. what μ maximizes P(the cat chases the mouse, DET N VBZ DET N|μ)
9. what μ maximizes P(the cat chases the mouse| μ))
10. **Viterbi algorithm**

**Set**

**Set**

**For** i **from** 1 **to** T

**For**

**Set**

**Append** t **to**

**Return**

**maximized**

**Forward algorithm also! (changing the maximum into a sum)**

1. **Supervised learning of HMMs**

p(w|s)=

**Syntax 2**

1. **Parsing: ambiguity**

Example: the astronomers saw the stars with telescopes.

How to deal with: get the highest score of the parsing tree using probability.

1. **Context-free grammars**

A context free grammar G=(N, Σ, P, S) consists of:

* A set of non-terminal symbols N

e.g. “N”, “VP”, “S”

* A set of terminal symbols Σ

e.g. “cat”, “astronomer”

* A set of productions P

e.g. S→NP VP

* A start symbol “S”
* (PCFG) a probability function “D”

Note that (∀A)

1. **CYK algorithm (week6-P21)**

Avoid exponential costs by recursive implementation by simply keeping the values of t, i, j in a table and looing them up when they are needed.

1. **Chomsky Normal Form**

A→BC. A→a

1. **Problems of PCFGs**

**Problems:**

* Lexical dependencies

In a CFG the expansion of one non-terminal is independent of any other non-terminal.

However, in Francis et.al (1999), subjects are 91% pronouns and 9% lexical noun phrases, direct objects are 34% pronouns and 66% lexical noun phrases. Thus, we would need to distinguish between NP in subject and direct object positions

* Lexical attachment

Lexical attachment is very word dependent. For example, “he hit the man with a hat”. PCFGs cannot model lexical dependencies.

* Parse ambiguity not distinguished

Still has ambiguity.

**Solutions:**

* Lexicalized PCFGs

Further differentiate non-terminals by associating them with lexical items; leads to increased sparsity.

* Dependency grammars

Model links between individual words in the parse; complex to combine with standard CFG.