INF4820

POS Tagging
Hidden Markov Models

Erik Velldal

University of Oslo

Sep. 29, 2009

Today's Agenda

N-gram models

- Wrap up the n-gram LM presentation from last week's lecture.
- The sparse data problem + smoothing

Parts-Of-Speech

- Lexical categories
- POS Tagging
- Stochastic and symbolic approaches

Hidden Markov Models

- Start introducing HMMs for stochastic POS tagging

The Sparse Data Problem

- MLE works fine for events that occur with a high frequency in the training data.
- For unseen or low-frequency events, however, the MLE estimates will not generalize well to new data.

The Sparse Data Problem

- MLE works fine for events that occur with a high frequency in the training data.
- For unseen or low-frequency events, however, the MLE estimates will not generalize well to new data.
  - If n-gram x occurs twice, and n-gram y occurs once, is x really twice as likely as y?
  - Should unobserved n-grams have zero probability?
  - If a sequence contains an n-gram with a zero count, the probability of the entire sequence is zero.
  - What about unknown words?

The Sparse Data Problem (cont'd)

- Why can't we just include some more data and stop worrying?
- Chomsky: language use is a creative process.
  - Natural language continuously sees the addition of new words and new combinations of words.

The Sparse Data Problem (cont'd)

- Why can't we just include some more data and stop worrying?
- Chomsky: language use is a creative process.
  - Natural language continuously sees the addition of new words and new combinations of words.
- The general tendency described by Zipf's law is found to often fit well with empirical counts from corpus data.
  - A small number of events occur with high frequency, while a large number of events occur with a low frequency.
  - Long tail of rare events.

Alleviating the Sparse Data Problem

- Make provisions for out-of-vocabulary words (OOVs).
  - Include a designated token $< unk >$
  - Open vs closed vocabulary

Alleviating the Sparse Data Problem

- Make provisions for out-of-vocabulary words (OOVs).
  - Include a designated token $< unk >$
  - Open vs closed vocabulary
- Make sure all n-grams receive a non-zero count. Smoothing or discounting.
- General idea: take some of the probability mass of frequent events, and redistribute it to less frequent or unseen events.
- Makes the distribution less "spiked".
- Simplest approach: Add-One smoothing.

Add-One smoothing

- For all n-grams (including those with zero counts) add one to their counts in the training data.
- MLE probability: $P_{MLE}(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})}$
- Add-one probability: $P_{+1}(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i})+1}{C(w_{i-n+1}^{i-1})+V}$

Add-One smoothing

- For all n-grams (including those with zero counts) add one to their counts in the training data.
- MLE probability: $P_{MLE}(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})}$
- Add-one probability: $P_{+1}(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i})+1}{C(w_{i-n+1}^{i-1})+V}$
- Problems
  - Too much probability mass is shifted towards unseen n-grams.
  - Underestimates frequent events while overestimating rare events.
  - Uniform smoothing strategy of all n-grams, regardless of their counts.

Other Smoothing Techniques

Witten-Bell Discounting

- Redistributes probability mass depending on the context of words.
- For an unseen n-gram w_{i-n+1}^{i}, the probability $P_{WB}(w_i|w_{i-n+1}^{i-1})$ is higher if w_{i-n+1}^{i-1} has occured with many different words w_i'.

Other Smoothing Techniques

Witten-Bell Discounting

- Redistributes probability mass depending on the context of words.
- For an unseen n-gram w_{i-n+1}^{i}, the probability $P_{WB}(w_i|w_{i-n+1}^{i-1})$ is higher if w_{i-n+1}^{i-1} has occured with many different words w_i'.

Katz' Back-Off Smoothing

- If the count for the current n-gram is lower than some threshold m, revert to a shorter a n-gram context. Simplified version:

$$P_{BO}(w_i|w_{i-n+1}^{i-1}) = \begin{cases} P(w_i|w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) > m \\ P(w_i|w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases}$$

Other Smoothing Techniques (cont'd)

Deleted Interpolation

- A weighted sum of different models. S.c. *mixture model*.
- Similar to back-off, but we always include the predictions of the lower-order models regardless of the observed count.
- Called "deleted" because all the interpolated functions use a subset of the conditioning information of the most discriminating model (M&S, 1999). E.g. for a trigram LM we would have

$$
\begin{aligned}
P_{DI}(w_i|w_{i-2}, w_{i-1}) = & \lambda_1 P_1(w_i) + \\
& \lambda_2 P_2(w_i|w_{i-1}) + \\
& \lambda_3 P_3(w_i|w_{i-2}, w_{i-1})
\end{aligned}
$$

- For P_{DI} to be a proper distribution we require that $\sum_j \lambda_j = 1$.
- The λ-weights can be optimized using held-out data.

Other Smoothing Techniques (cont'd)

- And there are still many others; Good-Turing Discounting, Kneser-Ney Smoothing...
- Skip language models

Other Smoothing Techniques (cont'd)

- And there are still many others; Good-Turing Discounting, Kneser-Ney Smoothing. . .
- Skip language models
- Class-based language models
  - n-gram statistics over more general categories based on distributional properties or pre-defined categories such as e.g. types of proper nouns or lexical word class.

Markov Models (n-gram recap)

- We've already seen an example of ("visible") Markov Models: n-gram language models.
- Recall, a sequence of discrete random variables $(X_1, \ldots, X_k)$ is called a Markov chain if it has the following properties (for some $n \ll k$):
  - Limited Horizon / Memory:

$$P(X_t = o_k | X_1, \ldots, X_{t-1}) = P(X_t | X_{t-n+1}^{t-1})$$

  - Time Invariant / Stationary:

$$P(X_t = o_k | X_{t-n+1}^{t-1}) = P(X_5 = o_k | X_{5-n+1}^{4})$$

- Similar to a weighted FSA. Transitions associated with probabilities.

Hidden Markov Models

- "Visible" Markov Models are sufficient when dealing with sequences of observable variables.
- However, sometimes we want to model an additional layer of underlying hidden / source variables.

Hidden Markov Models

- "Visible" Markov Models are sufficient when dealing with sequences of observable variables.
- However, sometimes we want to model an additional layer of underlying hidden / source variables.
- A sequence of (unobserved) weather conditions for an (observed) sequence of holiday activities:

`museum`	`beach`	`beach`	`beach`	`museum`
RAINY	SUNNY	SUNNY	SUNNY	RAINY

Hidden Markov Models

- "Visible" Markov Models are sufficient when dealing with sequences of observable variables.
- However, sometimes we want to model an additional layer of underlying hidden / source variables.
- A sequence of (unobserved) weather conditions for an (observed) sequence of holiday activities:

`museum`	`beach`	`beach`	`beach`	`museum`
RAINY	SUNNY	SUNNY	SUNNY	RAINY

- A sequence of (unobserved) part-of-speech tags for an (observed) sequence of word forms:

This	is	a	short	sentence	.
DT	VBZ	DT	JJ	NN	.

Parts of Speech

- AKA: parts-of-speech, POS, lexical categories, word classes, morphological classes, lexical tags...
- Examples:

Tag	POS	Example
N	noun	chair, bandwidth, pacing
V	verb	study, debate, munch
ADJ	adjective	purple, tall, ridiculous
ADV	adverb	unfortunately, slowly
P	preposition	of, by, to
PRO	pronoun	I, me, mine
DET	determiner	the, a, that, those

- POS Tagging = The task of automatically assigning part-of-speech markers to words.

When is POS information useful?

First step in very many tasks

- Parsing / Chunking
- Machine Translation (MT)
  - (No.) *sky* $\rightarrow$ (En.) *cloud*, *shy*, *avoid*...?
- Lemmatization
- Word Sense Disambiguation (WSD)
- Information Extraction (IE)
- Helps producing the correct pronunciation in speech synthesis:
  - **IN**sult *vs* in**SULT**
- Build more accurate n-gram models...

Open vs Closed Classes

- Open Word Classes:
  - New words created all the time.
- Closed Word Classes:
  - Smaller classes with fixed membership.
  - Usually function words

Open vs Closed Classes

- Open Word Classes:
  - New words created all the time.
- Closed Word Classes:
  - Smaller classes with fixed membership.
  - Usually function words
- Let's rush through some examples just to refresh our memory...

Open Class Words

Nouns

- Typically denoting people, places, things, concepts, phenomena...
- Proper nouns (Oslo, Peter Sellers)
- Common nouns (the rest)
  - Count nouns: Countable, plural forms (chicken/chickens, one chicken, two chickens)
  - Mass nouns: Uncountable (snow, altruism, *two snows)

Open Class Words

Nouns

- Typically denoting people, places, things, concepts, phenomena...
- Proper nouns (Oslo, Peter Sellers)
- Common nouns (the rest)
  - Count nouns: Countable, plural forms (chicken/chickens, one chicken, two chickens)
  - Mass nouns: Uncountable (snow, altruism, *two snows)

Adjectives

- Typically descriptive of a noun, denoting properties, characteristics, qualities, etc.
- Can be compared for degree (*small* – *smaller* –*smallest*)

Open Class Words (cont'd)

Verbs

- Typically denoting actions, processes, etc.
- Morphological affixes for person, tense, and aspect (*eat*/*eats*/*eaten*)
  - Auxiliaries: Closed-class subclass

Open Class Words (cont'd)

Verbs

- Typically denoting actions, processes, etc.
- Morphological affixes for person, tense, and aspect (*eat*/*eats*/*eaten*)
  - Auxiliaries: Closed-class subclass

Adverbs

- Very heterogeneous lexical class
- Modifying verbs, verb phrases, or other adverbs.
  - Many possible subclasses:
  - Directional/locative adverbs (*here*, *home*, *downhill*)
  - Degree adverbs (*extremely*, *very*, *somewhat*)
  - Manner adverbs (*slowly*, *delicately*)
  - Temporal adverbs. . .

Closed Class Words

- Prepositions: *on*, *under*, *from*, *at*, *near*, *over*, ...
- Particles: *up*, *down*, *on*, *off*, *by*, ...
- Determiners: *a*, *an*, *the*, *that*, ...
- Pronouns: *she*, *who*, *I*, *others*, ...
- Conjunctions: *and*, *but*, *or*, *when*, ...
- Auxiliary verbs: *can*, *may*, *should*, *must*, ...
- Numerals: *one*, *two*, *first*, *third*, ...
- Interjections, negatives, politeness makers, greetings, existential there...

(Examples from J&M 2009)

Why is it hard?

Ambiguity

- Each word can have many possible POS.
  (More high-frequent words are often more ambiguous (economical))
- POS tagging is therefore a disambiguation task: Determine the POS for a particular occurrence of a word in context.

Why is it hard?

Ambiguity

- Each word can have many possible POS.
  (More high-frequent words are often more ambiguous (economical))
- POS tagging is therefore a disambiguation task: Determine the POS for a particular occurrence of a word in context.

A side note

- Various standardized tag sets with varying degree of coarseness.
- E.g. Brown, Penn TreeBank tag set, C5.
- Note that, the tags are usually a bit more specific than the word classes we've discussed above, e.g.denoting a *plural form common noun*, a *third-person singular present-tense verb*, etc..

Example (based on output from the Oslo-Bergen Tagger)

```
"<Beinet>"
        "bein" subst
        "beine" verb
        "beinet" adj
"<var>"
        "var" adj
        "var" subst
        "vare" verb
        "være" verb
"<rett>"
        "rett" adj
        "rett" subst
        "rette" verb
"<.>"
        "$." <punkt>
```

Example (based on output from the Oslo-Bergen Tagger)

```
"<Beinet>"
        "bein" subst
        "beine" verb
        "beinet" adj
"<var>"
        "var" adj
        "var" subst
        "vare" verb
        "være" verb
"<rett>"
        "rett" adj
        "rett" subst
        "rette" verb
"<.>"
        "$." <punkt>
```

Two Main Approaches

Rule-based ("symbolic")

- POS assignment and disambiguation based on manually crafted rules.

Two Main Approaches

Rule-based (“symbolic”)

- POS assignment and disambiguation based on manually crafted rules.

Stochastic (empirical / data-driven)

- Probabilistic sequence models
- Data-driven taggers are often based on the HMM approach.
- Trained on previously tagged data.

Two Main Approaches

Rule-based (“symbolic”)

- POS assignment and disambiguation based on manually crafted rules.

Stochastic (empirical / data-driven)

- Probabilistic sequence models
- Data-driven taggers are often based on the HMM approach.
- Trained on previously tagged data.

Common Overall Goal

- Use context to disambiguate candidate tags.

Rule-Based Tagging

Two Main Stages

- Look-up
  - Morphological analysis + dictionary look-up to assign all possible POS tags.
- Elimination
  - Apply hand written rules (possibly on the order of thousands) to remove inconsistent tags.

Rule-Based Tagging

Two Main Stages

- Look-up
  - Morphological analysis + dictionary look-up to assign all possible POS tags.
- Elimination
  - Apply hand written rules (possibly on the order of thousands) to remove inconsistent tags.

The Oslo-Bergen Tagger

- Example of a rule-based tagger for Norwegian.
- Defined by thousands of rules written in the Constraint Grammar format (Reg-Exp-like), with a Common Lisp interpreter.
- Categories based on *Norsk Referansegrammatikk*.

HMM Tagging as Bayesian Classification

- Given a sequence of words $w_1, \dots, w_n$, we want to find the most probable sequence of tags $t_1, \dots, t_n$.
- Applying Bayes' Rule, we can state our search problem as

$$\begin{aligned}\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n | w_1^n) &= \arg\max_{t_1^n} \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)} \\ &= \arg\max_{t_1^n} P(w_1^n | t_1^n) P(t_1^n)\end{aligned}$$

HMM Tagging as Bayesian Classification

- Given a sequence of words $w_1, \ldots, w_n$, we want to find the most probable sequence of tags $t_1, \ldots, t_n$.
- Applying Bayes' Rule, we can state our search problem as

$$\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n|w_1^n) = \arg\max_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$
$$= \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n)$$

- We'll make a few simplifying assumptions before rewriting further.
- Assume the Markov property for $P(t_1^n)$ (For simplicity we will use an bigram model here, but we can just as well use a higher-order n-gram model):

$$P(t_1^n) = P(t_1)P(t_2|t_1)P(t_3|t_1, t_2) \ldots P(t_n|t_1^{n-1})$$
$$\approx \prod_i P(t_i|t_{i-1})$$

HMM tagging as Bayesian Classification (cont'd)

- Make two more simplifying assumptions regarding $P(w_1^n|t_1^n)$.
  - Each word is conditionally independent of the other words given the tags:

$$
\begin{aligned}
P(w_1^n|t_1^n) &= P(w_1|t_1^n)P(w_2|w_1, t_1^n)\ldots P(w_n|w_1^{n-1}, t_1^n) \\
&\approx \prod_i P(w_i|t_1^n)
\end{aligned}
$$

HMM tagging as Bayesian Classification (cont'd)

- Make two more simplifying assumptions regarding $P(w_1^n|t_1^n)$.
  - Each word is conditionally independent of the other words given the tags:

$$\begin{aligned} P(w_1^n|t_1^n) &= P(w_1|t_1^n)P(w_2|w_1,t_1^n)\ldots P(w_n|w_1^{n-1},t_1^n) \\ &\approx \prod_i P(w_i|t_1^n) \end{aligned}$$

  - Each word is conditionally independent of all tags but its own:

$$\prod_i P(w_i|t_1^n) \approx \prod_i P(w_i|t_i)$$

HMM tagging as Bayesian Classification (cont'd)

- Make two more simplifying assumptions regarding $P(w_1^n|t_1^n)$.
  - Each word is conditionally independent of the other words given the tags:

$$\begin{aligned} P(w_1^n|t_1^n) &= P(w_1|t_1^n)P(w_2|w_1,t_1^n)\ldots P(w_n|w_1^{n-1},t_1^n) \\ &\approx \prod_i P(w_i|t_1^n) \end{aligned}$$

  - Each word is conditionally independent of all tags but its own:

$$\prod_i P(w_i|t_1^n) \approx \prod_i P(w_i|t_i)$$

- We can now finally formulate the search problem as:

$$\hat{t}_1^t = \arg\max_{t_1^n} P(t_1^n|w_1^n) \approx \arg\max_{t_1^n} \prod_i P(w_i|t_i)P(t_i|t_{i-1})$$

Estimation

Tag Transition Probabilities

Based on a training corpus of previously tagged text, the MLE can be computed from the counts of observed tags:

$$P(t_i|t_{t-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

Estimation

Tag Transition Probabilities

Based on a training corpus of previously tagged text, the MLE can be computed from the counts of observed tags:

$$P(t_i|t_{t-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

Word Likelihoods

Computed from relative frequencies in the same way: $P(w_i|t_j) = \frac{C(t_i, w_j)}{C(t_i)}$

Estimation

Tag Transition Probabilities

Based on a training corpus of previously tagged text, the MLE can be computed from the counts of observed tags:

$$P(t_i|t_{t-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})}$$

Word Likelihoods

Computed from relative frequencies in the same way: $P(w_i|t_j) = \frac{C(t_i,w_j)}{C(t_i)}$

Sparse Data Problem

The issues related to MLE / smoothing that we discussed for n-gram models also applies here...

Topics for the Next Lecture...

- Formal specification of an HMM; $< Q, A, O, B, q_0, q_F >$
- Dynamic Programming
  - The Forward algorithm for computing the HMM probability of an observed sequence of words.
  - The Viterbi algorithm for computing the HMM probability of an unobserved sequence of tags.