Ling/CSE 472: Introduction to Computational Linguistics

4/29/15

N-grams

Overview

- What are N-grams? (high-level)
- When are they useful?
- Evaluation
- Simple (un-smoothed) N-grams
- Training and test sets
- Unknown words
- N-grams and linguistic knowledge
- Reading questions
- Next time: smoothing, back-off, interpolation

What are N-grams?

- A way of modeling the probability of a string of words.
- ... or the probability of the N+1st word being *w* given words 1-N.

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

- We'll come back to this in more detail in a moment

When are they useful?

- When would you want to know the relative probability of two strings?
- ... the relative probability of two words given the previous N words?
- ... the absolute probability of a string?

When are they useful?

- When would you want to know the relative probability of two strings?
  - Choosing among ASR candidates
  - Choosing among MT candidates
- ... the relative probability of two words given the previous N words?
  - Choosing among spell correction candidates
  - Predictive text (T9, AAC): Does your cell phone store an n-gram model?
- ... the absolute probability of a string? (kind of)
  - Spell checking: relative to a threshold
  - Language ID, authorship ID: relative to probability from some other model

Evaluating N-gram models

- What kinds of extrinsic evaluation are possible?
- What kinds of intrinsic evaluation are possible?
- What different kinds of models could you compare?

Evaluating N-gram models

- What kinds of extrinsic evaluation are possible?
    - ASR, MT, ...
- What kinds of intrinsic evaluation are possible?
    - Perplexity: Given an n-gram model trained on some training set, how well does it predict the test set? (i.e., what probability does it assign to the test set?)
- What different kinds of models could you compare?
    - Different: training data, smoothing/back-off techniques, higher-level tokens

Training and test sets

- Training and test sets must be distinct, otherwise probabilities will be artificially high

- This is just a special case of a more general reason: The purpose of test sets is to see if the method generalizes to unseen data

- Training and test sets are typically drawn from the same or comparable corpora

  - Why?

  - What if they aren’t?

Simple (un-smoothed) N-grams

- What we'd really like to calculate:

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1})$$

$$= \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

- But we'd never find a corpus big enough.

- Why not?

So: An approximation

- Markov assumption: The probability of a given word only depends on the previous word

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

- Is this assumption valid?

- (NB: That's a bigram model ... for a trigram model, we look at the previous two words, etc.)

Maximum Likelihood Estimates for bigram counts

- Bigram probability for a word *y* given a previous word *x*:

- Out of all the times you saw *x*, in what percentage was it followed by *y*?

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

Probability v. Frequency

- Probability: How likely something is to happen
- Frequency: How frequently something has happened in a set of observations
- Probability clearly influences frequency
- Frequency can be used to estimate probability (what else can?)
    - ... but they are not the same thing
- If a bigram never appears in a training corpus,
    - What is its observed frequency?
    - What is its probability?

Example

- What are the bigrams in the following mini corpus? What are their MLEs?

<s> How much wood would a wood chuck chuck if a wood chuck could chuck wood? </s> <s> As much wood as a wood chuck could if a wood chuck could chuck wood. </s>

- What probability does that bigram model assign to the following sentences?

<s> How much wood. </s>
<s> How much wood? </s>
<s> As much wood chuck chuck chuck wood. </s>
<s> How would a wood chuck chuck ? </s>

bigrams

- <s> How = 1/2
- How much = 1
- much wood = 1
- wood would = 1/8
- would a = 1
- a wood = 1
- wood chuck = 1/2
- chuck chuck = 1/7
- chuck if = 1/7
- if a = 1
- chuck could = 3/7
- could chuck = 2/3
- chuck wood = 2/7
- wood ? = 1/8
- ? </s> = 1
- <s> As = 1/2
- As much = 1/2
- wood as = 1/8
- as a = 1/2
- could if = 1/3
- wood . = 1/8
- . </s> = 1

Sentences

<s> How much wood. </s>
<s> How much wood? </s>
<s> As much wood chuck chuck chuck wood. </s>
<s> How would a wood chuck chuck ? </s>

1. 1/2 * 1 * 1 * 1/8 * 1 = 1/16
2. 1/2 * 1 * 1 * 1/8 * 1 = 1/16
3. 1/2 * 1/2 * 1 * 1/2 * 1/7 * 1/7 * 2/7 * 1 = 1/1372
4. 1/2 * 0 ... = 0

Counting things in a corpus

- Type/token distinction

- But what counts as a token? What are some cases where this is not obvious?

- And what counts as the same type? What are some cases where this is not obvious?

- Is there a single right answer?

Counting things in a corpus

- Type/token distinction
- But what counts as a token? What are some cases where this is not obvious?
    - Contracted forms, punctuation, hyphenated forms, words with spaces (*New York*), ...
- And what counts as the same type? What are some cases where this is not obvious?
    - Caps/non-caps, word-form/lemma, homographs, ...
- Is there a single right answer?
    - No: It depends on the application context

Unknown words

- What would a n-gram model trained as described so far say about the probability of a sentence with an unknown word in it?

- What could be done about that?

Unknown words

- What would a n-gram model trained as described so far say about the probability of a sentence with an unknown word in it? -- 0

- What could be done about that?

- Choose a vocabulary smaller than the actual V, and replace all other words with <UNK> -- Any words you might want to be sure to include in V?

- Or: Replace the first occurrence of every word in the training set with <UNK>

- Then: Estimate probabilities of <UNK> just like any other word

N-grams and linguistic knowledge

- Is an n-gram model a grammar?
- What kinds of information about a language does it capture?
- What kinds of information about a language does it miss?

N-grams and linguistic knowledge

- N-gram models are supposed to be “language-independent” in that they don’t require specific knowledge about a language to create --- just text.

- Would you expect them to work equally well across languages? Why or why not?

Overview

- What are N-grams? (high-level)
- When are they useful?
- Evaluation
- Simple (un-smoothed) N-grams
- Training and test sets
- Unknown words
- N-grams and linguistic knowledge
- Reading questions
- Next time: smoothing, back-off, interpolation

Reading questions

- When the book talks about picking "random" n-grams on page 93, how does that work? It says n-grams for the Shakespeare set were generated randomly and according to their probability, which is confusing. Does this mean that if some number of words had the same probability then one of them would be chosen at random? Otherwise, wouldn't it just be that whatever next word had the highest probability based on the previous word/s would be chosen (not at random), and only the first word in a sequence would be truly randomly chosen?

Reading questions

- The keyboard on my Android phone will present 3 words as suggestions after anything you type. I tested this out after the reading and found it will often get itself into a loop. Typing 'the' and selecting 'following' will then suggest 'the' as the next word, and then "following', "the", etc forever giving: "the following the following the following". It seems clear then that this is only using the previous word to give a suggestion, not a trigram or quadrigram model. I understand from the reading that tri and quadrigram models rely much more strongly on the input test data and can hard to use effectively without a large training set of text from the same 'genre'; however I wonder if any sort of real time analysis of the output being created can't be done to account for cases like this - to lower the probability of some phrase or pair of words after it is used consecutively even one time. The reading didn't seem to mention that so I wonder why it wouldn't be done.

Reading questions

- I know that in some applications that use n-gram models, like autocorrect, the models can incorporate new information into their internal model. For example, if I constantly ignore autocorrect's suggestion for a particular name, it will eventually stop trying to correct it. How do n-gram models incorporate new information like this?

Reading questions

- Two questions about trigrams. First, what do the count/probability tables (figures 4.1 and 4.2) for trigrams look like? Where does the third word get added in? Second, when writing out the notation for trigrams, which order do the first two words go in. For example, is the notation for finding Bob after I saw P(Bob|<I><saw>) or P(Bob|<saw><I>)?

- I'm confused about how the bigram model is generalized into something that predicts the "probability of a complete word sequence". My understanding of the chain rule of probability is not good either, which might be why I'm having trouble with this. It seems like the idea of perplexity also builds on this, so I guess I would benefit from an overview on how these ideas work.

Reading questions

- Regarding training corpora, does an increase in the size of the training set correlate with more natural sentence generation in N-gram models where N is small? Is there a limit as to how 'good' an N-gram model can get given an infinitely large training set?

- Section 4.3.2 mentions placing "unknown words" into the training data, but what does this do for N-Gram?

- Will n gram probabilities tend to be very skewed for rare words? For example, if "bivouac" occurred only once in a training set, in the context "... a bivouac that...", and we are measuring bigram probabilities, then P(that|bivouac) = 1 and P([anything else] | bivouac) = 0. Could we give an n-gram prediction a "certainty" related to the number of samples we were able to use to calculate the given probability?

Reading questions

- I am confused by the idea of Perplexity. Is it just a way of determining which model is a better model through computing the probability of each sentence in the test set? If so, why is it that minimizing perplexity is equivalent to maximizing the test set probability according to the language model?

- The text says that perplexity is not comparable between different corpora. If then we use the same n-gram model proposed in the book on our various corpora, we don't have perplexities of models we can compare to each other. Wouldn't that then make perplexity a meaningless measure?

- In which case an intrinsic improvement in perplexity does not guarantee and extrinsic improvement, and why? Also, why the improvement in perplexity can be confirmed by extrinsic evaluation?

Reading questions

- If perplexity is the weighted average branching factor of a language, why would the unigram perplexity for a test set of 1.5 million words be 962. I thought that unigrams could have any other unigram as a possible branch since they are independent. Shouldn't the perplexity be closer to 1.5 million or even 38 million (since this was the number of words in the test set)? Am I misinterpreting the concept of perplexity or is this a discrepancy that occurs as a difference between the contents of the test and training data and/or that the fact words may repeat within data?

- How does the entropy of English compare to other written or spoken languages? How much is this a function of the number of characters in the alphabet, and how much of it has to do with effectively conveying information without duplication?