CS447: Natural Language Processing
http://courses.engr.illinois.edu/cs447

Lecture 3: Language models

Julia Hockenmaier
juliahmr@illinois.edu
3324 Siebel Center

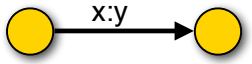
Last lecture's key concepts

Morphology (word structure): stems, affixes
Derivational vs. inflectional morphology
Compounding
Stem changes
Morphological analysis and generation

Finite-state automata
Finite-state transducers
Composing finite-state transducers

Finite-state transducers

- FSTs define a **relation** between two regular languages.
- Each state transition maps (**transduces**) a character from the input language to a character (or a sequence of characters) in the output language



- By using the **empty character** (ε), characters can be **deleted** (x:ε) or **inserted**(ε:y)

- FSTs can be composed (**cascaded**), allowing us to define **intermediate representations**.

Today's lecture

How can we distinguish word salad, spelling errors and grammatical sentences?

Language models define probability distributions over the strings in a language.
N-gram models are the simplest and most common kind of language model.

We'll look at how they're defined, how to estimate (learn) them, and what their shortcomings are.

We'll also review some very basic probability theory.

Why do we need language models?

Many NLP tasks return output in natural language:
- Machine translation
- Speech recognition
- Natural language generation
- Spell-checking

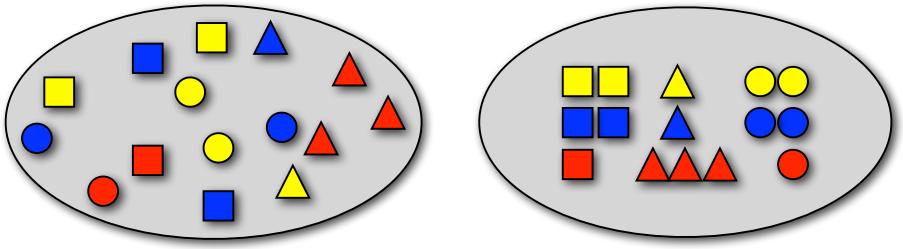
Language models define probability distributions over (natural language) strings or sentences.

We can use them to score/rank possible sentences:
If $P_{LM}(A) > P_{LM}(B)$, choose sentence A over B

Reminder: Basic Probability Theory

Sampling with replacement

Pick a random shape, then put it back in the bag.



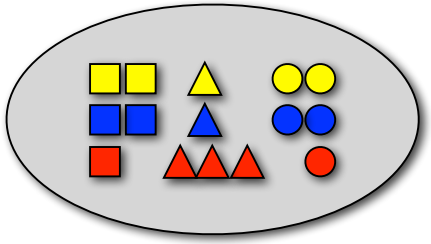
$P(\blacksquare) = 2/15$
$P(\text{blue}) = 5/15$
$P(\text{blue} \mid \square) = 2/5$

$P(\blacksquare) = 1/15$
$P(\text{red}) = 5/15$
$P(\square) = 5/15$

$P(\blacksquare \text{ or } \blacktriangle) = 2/15$
$P(\triangle \mid \text{red}) = 3/5$

Sampling with replacement

Pick a random shape, then put it back in the bag.
What sequence of shapes will you draw?



$P(\blacksquare\blacktriangle\blacktriangle\blacksquare) = 1/15 \times 1/15 \times 1/15 \times 2/15 = 2/50625$

$P(\blacktriangle\bullet\bullet\blacktriangle) = 3/15 \times 2/15 \times 2/15 \times 3/15 = 36/50625$

$P(\blacksquare) = 2/15$
$P(\text{blue}) = 5/15$
$P(\text{blue} \mid \square) = 2/5$

$P(\blacksquare) = 1/15$
$P(\text{red}) = 5/15$
$P(\square) = 5/15$

$P(\blacksquare \text{ or } \blacktriangle) = 2/15$
$P(\triangle \mid \text{red}) = 3/5$

Sampling with replacement

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

$P(\text{of}) = 3/66$	$P(\text{her}) = 2/66$
$P(\text{Alice}) = 2/66$	$P(\text{sister}) = 2/66$
$P(\text{was}) = 2/66$	$P(\text{,}) = 4/66$
$P(\text{to}) = 2/66$	$P(\text{'}) = 4/66$

Sampling with replacement

beginning by, very Alice but was and? reading no tired of to into sitting sister the, bank, and thought of without her nothing: having conversations Alice once do or on she it get the book her had peeped was conversation it pictures or sister in, 'what is the use had twice of a book''pictures or' to

$P(\text{of}) = 3/66$	$P(\text{her}) = 2/66$
$P(\text{Alice}) = 2/66$	$P(\text{sister}) = 2/66$
$P(\text{was}) = 2/66$	$P(\text{,}) = 4/66$
$P(\text{to}) = 2/66$	$P(\text{'}) = 4/66$

In this model, $P(\textit{English sentence}) = P(\textit{word salad})$

Probability theory: terminology

Trial:
Picking a shape, predicting a word

Sample space Ω:
The set of all possible outcomes
(all shapes; all words in *Alice in Wonderland*)

Event $\omega \subseteq \Omega$:
An actual outcome (a subset of Ω)
(predicting '*the*', picking a triangle)

The probability of events

Kolmogorov axioms:
1) Each event has a probability between 0 and 1.
2) The null event has probability 0.
   The probability that any event happens is 1.
3) The probability of all disjoint events sums to 1.

$$0 \leq P(\omega \subseteq \Omega) \leq 1$$
$$P(\emptyset) = 0 \text{ and } P(\Omega) = 1$$
$$\sum_{\omega_i \subseteq \Omega} P(\omega_i) = 1 \quad \text{if } \forall j \neq i : \omega_i \cap \omega_j = \emptyset \text{ and } \bigcup_i \omega_i = \Omega$$

Discrete probability distributions: single trials

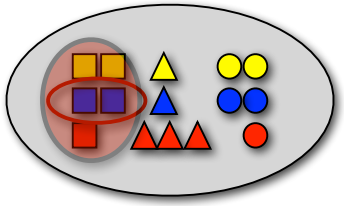
Bernoulli distribution (two possible outcomes)
The probability of success (=head,yes)
- The probability of *head* is p.
- The probability of *tail* is $1-p$.

Categorical distribution (*N* possible outcomes)
The probability of category/outcome c_i is p_i
$(0 \leq p_i \leq 1 \; \sum_i p_i = 1)$

Joint and Conditional Probability

The conditional probability of X given Y, $P(X \mid Y)$, is defined in terms of the probability of Y, $P(Y)$, and the joint probability of X and Y, $P(X,Y)$:

$$P(X|Y) = \frac{P(X,Y)}{P(Y)}$$



$P(\text{blue} \mid \blacksquare) = 2/5$

Conditioning on the previous word

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

$P(w_{i+1} = \textbf{of} \mid w_i = \textbf{tired}) = 1$
$P(w_{i+1} = \textbf{of} \mid w_i = \textbf{use}) = 1$
$P(w_{i+1} = \textbf{sister} \mid w_i = \textbf{her}) = 1$
$P(w_{i+1} = \textbf{beginning} \mid w_i = \textbf{was}) = 1/2$
$P(w_{i+1} = \textbf{reading} \mid w_i = \textbf{was}) = 1/2$

$P(w_{i+1} = \textbf{bank} \mid w_i = \textbf{the}) = 1/3$
$P(w_{i+1} = \textbf{book} \mid w_i = \textbf{the}) = 1/3$
$P(w_{i+1} = \textbf{use} \mid w_i = \textbf{the}) = 1/3$

Conditioning on the previous word

English

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

Word Salad

beginning by, very Alice but was and? reading no tired of to into sitting sister the, bank, and thought of without her nothing: having conversations Alice once do or on she it get the book her had peeped was conversation it pictures or sister in, 'what is the use had twice of a book''pictures or' to

Now, P(English) ≫ P(word salad)

$P(w_{i+1} = \textbf{of} \mid w_i = \textbf{tired}) = 1$
$P(w_{i+1} = \textbf{of} \mid w_i = \textbf{use}) = 1$
$P(w_{i+1} = \textbf{sister} \mid w_i = \textbf{her}) = 1$
$P(w_{i+1} = \textbf{beginning} \mid w_i = \textbf{was}) = 1/2$
$P(w_{i+1} = \textbf{reading} \mid w_i = \textbf{was}) = 1/2$

$P(w_{i+1} = \textbf{bank} \mid w_i = \textbf{the}) = 1/3$
$P(w_{i+1} = \textbf{book} \mid w_i = \textbf{the}) = 1/3$
$P(w_{i+1} = \textbf{use} \mid w_i = \textbf{the}) = 1/3$

The chain rule

The joint probability $P(X,Y)$ can also be expressed in terms of the conditional probability $P(X \mid Y)$

$$P(X,Y) = P(X|Y)P(Y)$$

This leads to the so-called chain rule:

$$\begin{aligned} P(X_1, X_2, \ldots, X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)....P(X_n|X_1, ...X_{n-1}) \\ &= P(X_1)\prod_{i=2}^{n} P(X_i|X_1 \ldots X_{i-1}) \end{aligned}$$

Independence

Two random variables X and Y are independent if

$$P(X,Y) = P(X)P(Y)$$

If X and Y are independent, then $P(X \mid Y) = P(X)$:

$$\begin{aligned} P(X|Y) &= \frac{P(X,Y)}{P(Y)} \\ &= \frac{P(X)P(Y)}{P(Y)} \quad (X,\ Y \text{ independent}) \\ &= P(X) \end{aligned}$$

Probability models

Building a probability model consists of two steps:
1. Defining the model
2. Estimating the model's parameters
   (= training/learning)

Models (almost) always make
independence *assumptions.*

That is, even though X and Y are not actually independent, our model may treat them as independent.

This reduces the number of model parameters that we need to estimate (e.g. from n^2 to $2n$)

Language modeling with n-grams

Language modeling with N-grams

A language model over a vocabulary V assigns probabilities to strings drawn from V*.

Recall the chain rule:

$$P(w_1...w_i) \quad = \quad P(w_1)P(w_2|w_1)P(w_3|w_1w_2)...P(w_i|w_1...w_{i-1})$$

An n-gram language model assumes each word depends only on the last n–1 words:

$$P_{ngram}(w_1...w_i) \quad := \quad P(w_1)P(w_2|w_1)...P(\underbrace{w_i}_{nth\ word}|\underbrace{w_{i-n-1}...w_{i-1}}_{prev.\ n-1\ words})$$

N-gram models

Unigram model	$P(w_1)P(w_2)...P(w_i)$
Bigram model	$P(w_1)P(w_2|w_1)...P(w_i|w_{i-1})$
Trigram model	$P(w_1)P(w_2|w_1)...P(w_i|w_{i-2}w_{i-1})$
N-gram model	$P(w_1)P(w_2|w_1)...P(w_i|w_{i-n-1}...w_{i-1})$

N-gram models assume each word (event) depends only on the previous n–1 words (events).
Such independence assumptions are called Markov assumptions (of order n–1).

$$P(w_i|w_1...w_{i-1}) :\approx P(w_i|w_{i-n-1}...w_{i-1})$$

Estimating N-gram models

1. Bracket each sentence by special start and end symbols:

   <s> Alice was beginning to get very tired… </s>
   (We only assign probabilities to strings `<s>...</s>`)

2. Count the frequency of each n-gram….
   C(`<s> Alice`) = 1, C(`Alice was`) = 1,….

3. .... and normalize these frequencies to get the probability:

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

This is called a relative frequency estimate of $P(w_n \mid w_{n-1})$

Start and End symbols <s>… <\s>

Why do we need a **start-of-sentence** symbol?

This is just a mathematical convenience, since it allows us to write e.g. $P(w_1 \mid \langle s \rangle)$ for the probability of the first word in analogy to $P(w_{i+1} \mid w_i)$ for any other word.

Why do we need an **end-of-sentence** symbol?

This is necessary if we want to compare the probability of strings of different lengths (and actually define a probability distribution over V*).

We include <\s> in the vocabulary V, require that each string ends in <\s> and that <\s> can only appear at the end of sentences, and estimate $P(w_{i+1} = \langle /s \rangle \mid w_i)$.

Parameter estimation (training)

Parameters: the actual probabilities

$P(w_i = \text{'the'} \mid w_{i-1} = \text{'on'}) = ???$

We need (a large amount of) text as training data to estimate the parameters of a language model.

The most basic estimation technique:
relative frequency estimation (= counts)

$P(w_i = \text{'the'} \mid w_{i-1} = \text{'on'}) = C(\text{'on the'}) / C(\text{'on'})$

Also called Maximum Likelihood Estimation (MLE)

MLE assigns *all* probability mass to events that occur in the training corpus.

How do we use language models?

Independently of any application, we can use a language model as a random sentence generator
(i.e we sample sentences according to their language model probability)

Systems for applications such as machine translation, speech recognition, spell-checking, generation, often produce multiple candidate sentences as output.

- We prefer output sentences S_{Out} that have a higher probability
- We can use a language model $P(S_{Out})$ to score and rank these different candidate output sentences, e.g. as follows:

$\text{argmax}_{SOut}\, P(S_{Out} \mid \text{Input}) = \text{argmax}_{SOut}\, P(\text{Input} \mid S_{Out}) P(S_{Out})$

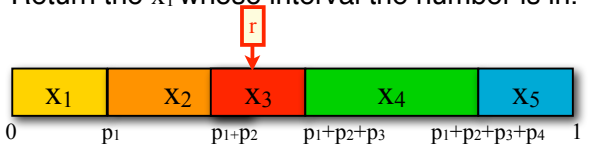
Using n-gram models to generate language

Generating from a distribution

How do you generate text from an *n*-gram model?

That is, how do you sample from a distribution $P(X \mid Y=y)$?

- Assume X has N possible outcomes (values): $\{x_1, \ldots, x_N\}$ and $P(X=x_i \mid Y=y) = p_i$
- Divide the interval [0,1] into N smaller intervals according to the probabilities of the outcomes
- Generate a random number r between 0 and 1.
- Return the x_1 whose interval the number is in.



Generating the Wall Street Journal

unigram: Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

bigram: Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

trigram: They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

Generating Shakespeare

Unigram	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
Bigram	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
Quadrigram	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Intrinsic vs Extrinsic Evaluation

How do we know whether one language model is better than another?

There are two ways to evaluate models:

- intrinsic evaluation captures how well the model captures what it is supposed to capture (e.g. probabilities)
- extrinsic (task-based) evaluation captures how useful the model is in a particular task.

Both cases require an evaluation metric that allows us to measure and compare the performance of different models.

How do we evaluate models?

Define an evaluation metric (scoring function).

We will want to measure how similar the predictions of the model are to real text.

Train the model on a *'seen'* training set

Perhaps: tune some parameters based on held-out data (disjoint from the training data, meant to emulate unseen data)

Test the model on an *unseen* test set

(usually from the same source (e.g. WSJ) as the training data)

Test data must be disjoint from training and held-out data

Compare models by their scores (more on this next week).

Intrinsic Evaluation of Language Models: Perplexity

Perplexity

Perplexity is the inverse of the probability of the test set (as assigned by the language model), normalized by the number of word tokens in the test set.

Minimizing perplexity = maximizing probability!

Language model LM_1 is better than LM_2
if LM_1 assigns lower perplexity (= higher probability)
to the test corpus $w_1 \ldots w_N$

NB: the perplexity of LM_1 and LM_2 can only be directly compared if both models use the same vocabulary.

Perplexity

The inverse of the probability of the test set, normalized by the number of tokens in the test set.

Assume the test corpus has N tokens, $w_1 \ldots w_N$
If the LM assigns probability $P(w_1, \ldots, w_{i-n})$ to the test corpus, its perplexity, $PP(w_1 \ldots w_N)$, is defined as:

$$PP(w_1 ... w_N) = P(w_1 ... w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 ... w_N)}}$$

A LM with lower perplexity is better because it assigns a higher probability to the unseen test corpus.

Perplexity $PP(w_1 \ldots w_n)$

Given a test corpus with N tokens, $w_1 \ldots w_N$,
and an n-gram model $P(w_i \mid w_{i-1}, \ldots, w_{i-n+1})$
we compute its perplexity $PP(w_1 \ldots w_N)$ as follows:

$$PP(w_1 ... w_N) = P(w_1 ... w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{P(w_1 ... w_N)}} = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 ... w_{i-1})}}$$ (Chain rule)

$$= def \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1}, ..., w_{i-n+1})}}$$ (N-gram model)

Practical issues

Since language model probabilities are very small, multiplying them together often yields to underflow.

It is often better to use logarithms instead, so replace

$$PP(w_1 \ldots w_N) =_{def} \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1}, \ldots, w_{i-n+1})}}$$

with

$$PP(w_1 \ldots w_N) =_{def} \exp\left(-\frac{1}{N}\sum_{i=1}^{N} \log P(w_i|w_{i-1}, \ldots, w_{i-n+1}\right)$$

Perplexity and LM order

Bigram LMs have lower perplexity than unigram LMs
Trigram LMs have lower perplexity than bigram LMs
…

Example from the textbook
(WSJ corpus, trained on 38M tokens, tested on 1.5 M tokens, vocabulary: 20K word types)

	Unigram	**Bigram**	**Trigram**
Perplexity	962	170	109

Extrinsic (Task-Based) Evaluation of LMs: Word Error Rate

Intrinsic vs. Extrinsic Evaluation

Perplexity tells us which LM assigns a higher probability to unseen text

This doesn't necessarily tell us which LM is better for our task (i.e. is better at scoring candidate sentences)

Task-based evaluation:
- Train model A, plug it into your system for performing task T
- Evaluate performance of system A *on task T*.
- Train model B, plug it in, evaluate system B on same task T.
- Compare scores of system A and system B on task T.

Word Error Rate (WER)

Originally developed for speech recognition.

How much does the predicted sequence of words differ from the actual sequence of words in the correct transcript?

$$\text{WER} = \frac{\text{Insertions} + \text{Deletions} + \text{Substitutions}}{\text{Actual words in transcript}}$$

Insertions: "eat lunch" → "eat **a** lunch"
Deletions: "see **a** movie" → "see movie"
Substitutions: "drink **ice** tea"→ "drink **nice** tea"

But….

… unseen test data will contain unseen words

Getting back to Shakespeare…

Generating Shakespeare

Unigram	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
Bigram	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
Quadrigram	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Shakespeare as corpus

The Shakespeare corpus consists of N=884,647 word **tokens** and a vocabulary of V=29,066 word **types**

Shakespeare produced 300,000 bigram types out of V^2= 844 million possible bigram types.

99.96% of possible bigrams don't occur in the corpus.

Our relative frequency estimate assigns non-zero probability to only 0.04% of the possible bigrams
That percentage is even lower for trigrams, 4-grams, etc.
4-grams *look* like Shakespeare because they *are* Shakespeare!

MLE doesn't capture unseen events

We estimated a model on 440K word tokens, but:

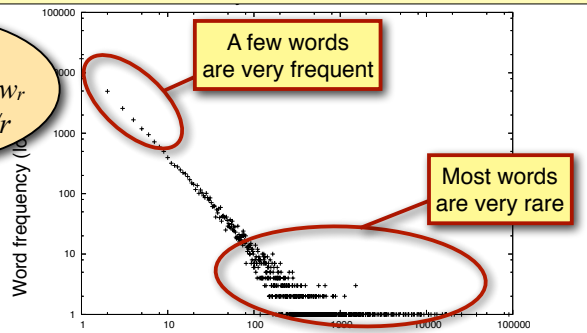
Only 30,000 word types occur in the training data
Any word that does not occur in the training data has zero probability!

Only 0.04% of all possible bigrams (over 30K word types) occur in the training data
Any bigram that does not occur in the training data has zero probability (even if we have seen both words in the bigram)

Zipf's law: the long tail

How many words occur once, twice, 100 times, 1000 times?

the *r*-th most common word w_r has $P(w_r) \propto 1/r$



In natural language:
- A small number of events (e.g. words) occur with high frequency
- A large number of events occur with very low frequency

So….

… we can't actually evaluate our MLE models on unseen test data (or system output)…

… because both are likely to contain words/n-grams that these models assign zero probability to.

We need language models that assign some probability mass to unseen words and n-grams.

We will get back to this on Friday.

To recap….

Today's key concepts

N-gram language models
Independence assumptions
Relative frequency (maximum likelihood) estimation
Evaluating language models: Perplexity, WER
Zipf's law

Today's reading:

Jurafsky and Martin, Chapter 4, sections 1-4 (2008 edition)
Chapter 3 (3rd Edition)

Friday's lecture: Handling unseen events!