Natural Language Understanding, Generation, and Machine Translation

Lecture 3: Conditional Language Models (with n-grams)

Alexandra Birch
19 January 2024 (week 1)

School of Informatics
University of Edinburgh

a.birch@ed.ac.uk

Based on slides by Adam Lopez.
Revision

Language models

\(n\)-gram Language models

Conditional language models

Modeling translation with \(n\)-grams

Parameter estimation

Decoding

Required, optional, and revision readings are listed on Opencourse.
Agenda for Today

**Last lecture:** should have given you some intuitions about how to model the problem of machine translation.

**This lecture:** see how to turn those intuitions into a probabilistic model that can be learned from data and used to translate new sentences.
Revision
Summer is hot, winter is
Predict the next word!

Summer is hot winter is ___
She is drinking a hot cup of ___
In the park I saw a giraffe.
In the park I saw a ______

Image captioning
A language model is a probabilistic model of strings
Example: Train a probabilistic model from CNN Business Headlines.

- Disneyland raises prices ahead of new Star Wars land opening
- Face-scanning technology at Orlando airport expands to all international travelers
- More than 1 million people subscribe to this electric toothbrush startup
- Heart drug recall expanded again
A language model is a probabilistic model of strings

Example: Train a probabilistic model from CNN Business Headlines.

- Disneyland raises prices ahead of new Star Wars land opening
- Face-scanning technology at Orlando airport expands to all international travelers
- More than 1 million people subscribe to this electric toothbrush startup
- Heart drug recall expanded again

Sample new headlines:

- Star Wars Episode IX Has New Lime Blazer

Source: http://aiweirdness.com/
A language model is a probabilistic model of strings

Example: Train a probabilistic model from CNN Business Headlines.

• Disneyland raises prices ahead of new Star Wars land opening
• Face-scanning technology at Orlando airport expands to all international travelers
• More than 1 million people subscribe to this electric toothbrush startup
• Heart drug recall expanded again

Sample new headlines:

• Star Wars Episode IX Has New Lime Blazer
• Coca-Cola is Scanning Your Messages for Big Chinese Tech

Source: http://aiweirdness.com/
A language model is a probabilistic model of strings

Example: Train a probabilistic model from CNN Business Headlines.

- Disneyland raises prices ahead of new Star Wars land opening
- Face-scanning technology at Orlando airport expands to all international travelers
- More than 1 million people subscribe to this electric toothbrush startup
- Heart drug recall expanded again

Sample new headlines:

- Star Wars Episode IX Has New Lime Blazer
- Coca-Cola is Scanning Your Messages for Big Chinese Tech
- Amazon is Recalling 1 Trillion Jobs

Source: http://aiweirdness.com/
Conditional language models have many uses

There are many, many applications where we want to predict words *conditioned on some input*:

- speech recognition: condition on speech signal
- machine translation: condition on text in another language
- text completion: condition on the first few words of a sentence
- optical character recognition: condition on an image of text
- image captioning: condition on an image
- grammar checking: condition on surrounding words
DISCLAIMER: Notation is not universally consistent!

In each lecture: notation will be consistent. Variables named.
If you find something confusing or inconsistent, PLEASE ASK!
Someone else also found it confusing or inconsistent.
Across lectures: notation will be similar, but not identical.
Expect notation to be **internally consistent** in an individual lecture, paper, or exam question, not globally consistent.

In general: there is no universally agreed upon notation for any of this stuff. Different fields and even subfields have different conventions, but even they tend to vary.
In each lecture: notation will be consistent. Variables named.

If you find something confusing or inconsistent, PLEASE ASK! Someone else also found it confusing or inconsistent.

Across lectures: notation will be similar, but not identical.

Expect notation to be **internally consistent** in an individual lecture, paper, or exam question, not globally consistent.

In general: there is no universally agreed upon notation for any of this stuff. Different fields and even subfields have different conventions, but even they tend to vary.

tl;dr: Notation is a kind of language, and there are many different dialects. I might code switch between dialects without noticing.
Given a finite vocabulary $V$, we want to define a probability distribution $P : V^* \rightarrow \mathbb{R}_+$. 

Revision questions:

- What is the sample space?
- What might be some useful random variables?
- What constraints must $P$ satisfy?
Given a finite vocabulary $V$, we want to define a probability distribution $P : V^* \rightarrow \mathbb{R}_+$. The *finite vocabulary* bit should worry you. We’ll come back to this, but not today!
Language modeling as probabilistic prediction

Given a finite vocabulary $V$, we want to define a probability distribution $P : V^* \rightarrow \mathbb{R}_+$. The *finite vocabulary* bit should worry you. We’ll come back to this, but not today!

Revision questions:

- What is the sample space?
- What might be some useful random variables?
- What constraints must $P$ satisfy?
How to derive an $n$-gram language model

Let $w$ be a sequence of words. Let $|w|$ be its length and let $w_i$ be its $i$th word. So, $w = w_1 \ldots w_{|w|}$.
How to derive an $n$-gram language model

Let $w$ be a sequence of words. Let $|w|$ be its length and let $w_i$ be its $i$th word. So, $w = w_1 \ldots w_{|w|}$.

Q: How do we define the probability $P(w) = P(w_1 \ldots w_{|w|})$?
How to derive an $n$-gram language model

Let $w$ be a sequence of words. Let $|w|$ be its length and let $w_i$ be its $i$th word. So, $w = w_1 \ldots w_{|w|}$.

Q: How do we define the probability $P(w) = P(w_1 \ldots w_{|w|})$?

Let $W_i$ be a random variable taking value of word at position $i$. Use the chain rule:

$$P(w_1 \ldots w_{|w|}) = P(W_1 = w_1) \times P(W_2 = w_2 | W_1 = w_1) \times \ldots \times P(W_{|w|} = w_{|w|} | W_1 = w_1, \ldots, W_{|w|-1} = w_{|w|-1})$$

Note: $\langle \text{STOP} \rangle$ is a symbol not in $V$. 
How to derive an $n$-gram language model

Let $w$ be a sequence of words. Let $|w|$ be its length and let $w_i$ be its $i$th word. So, $w = w_1 \ldots w_{|w|}$.

Q: How do we define the probability $P(w) = P(w_1 \ldots w_{|w|})$?

Let $W_i$ be a random variable taking value of word at position $i$.

Use the chain rule:

$$P(w_1 \ldots w_{|w|}) = P(W_1 = w_1) \times$$

$$P(W_2 = w_2 \mid W_1 = w_1) \times$$

$$\ldots$$

$$P(W_{|w|} = w_{|w|} \mid W_1 = w_1, \ldots, W_{k-1} = w_{|w|-1})$$

$$P(W_{|w|+1} = \langle \text{STOP} \rangle \mid W_1 = w_1, \ldots, W_k = w_{|w|})$$

Note: $\langle \text{STOP} \rangle$ is a symbol not in $V$. 


Written more concisely

\[ P(w_1 \ldots w_{|w|}) = P(w_1) \times \]
\[ P(w_2 \mid w_1) \times \]
\[ \ldots \]
\[ P(w_{|w|} \mid w_1, \ldots, w_{|w|-1}) \]
\[ = \prod_{i=1}^{|w|+1} P(w_i \mid w_1, \ldots, w_{|w|-1}) \]

Defines a \textit{joint distribution} over an \textit{infinite} sample space in terms of \textit{conditional distributions}, each over a \textit{finite} sample space (but with potentially infinite history!)
Written more concisely

\[ P(w_1 \ldots w_{|w|}) = P(w_1) \times P(w_2 \mid w_1) \times \ldots \times P(w_{|w|} \mid w_1, \ldots, w_{|w|-1}) \]

\[ = \prod_{i=1}^{|w|+1} P(w_i \mid w_1, \ldots, w_{|w|-1}) \]

Defines a joint distribution over an infinite sample space in terms of conditional distributions, each over a finite sample space (but with potentially infinite history!)
Written more concisely

\[ P(w_1 \ldots w_{|w|}) = P(w_1) \times P(w_2 \mid w_1) \times \ldots \]
\[ \cdots \]
\[ P(w_{|w|} \mid w_1, \ldots, w_{|w|-1}) \]
\[ P(\langle \text{STOP} \rangle \mid w_1, \ldots, w_{|w|}) \]
\[ = \prod_{i=1}^{\lvert w \rvert + 1} P(w_i \mid w_1, \ldots, w_{\lvert w \rvert - 1}) \]

Defines a \textit{joint distribution} over an \textit{infinite} sample space in terms of \textit{conditional distributions}, each over a \textit{finite} sample space (but with potentially infinite history!)
n-gram models make all terms finite with a Markov assumption

\[ P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \]
\[ P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \]

What is \( P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \)?
$P(w_i \mid w_{i-1}, \ldots, w_{i-n+1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1})$

What is $P(w_i \mid w_{i-n+1}, \ldots, w_{i-1})$?

Given $w_{i-n+1}, \ldots, w_{i-1}$, $P$ is a probability distribution, hence:

Probabilities must be non-negative

$P : V \rightarrow \mathbb{R}_+$

$n$-gram models make all terms finite with a Markov assumption.
n-gram models make all terms finite with a Markov assumption

\[ P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \]

What is \( P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \)?

Given \( w_{i-n+1}, \ldots, w_{i-1} \), \( P \) is a probability distribution, hence:

Probabilities must be non-negative

... and all sum to one

\[ \sum_{w \in V} P(w \mid w_{i-n+1}, \ldots, w_{i-1}) = 1 \]
*n*-gram models make all terms finite with a Markov assumption

\[ P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \]

What is \( P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \)?

Given \( w_{i-n+1}, \ldots, w_{i-1} \), \( P \) is a probability distribution, hence:

Probabilities must be non-negative

... and all sum to one

\[ \sum_{w \in V} P(w \mid w_{i-n+1}, \ldots, w_{i-1}) = 1 \]

Any function satisfying these constraints is a probability distribution! How would you define one?
\( P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \)

What is \( P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \)?

Given \( w_{i-n+1}, \ldots, w_{i-1} \), \( P \) is a probability distribution, hence:

\[
\text{Probabilities must be non-negative} \quad P : V \rightarrow \mathbb{R}_+ \\
\text{... and all sum to one} \quad \sum_{w \in V} P(w \mid w_{i-n+1}, \ldots, w_{i-1}) = 1
\]

Any function satisfying these constraints is a probability distribution! How would you define one?

Simple idea: since the number of \( P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \) terms is finite, let each one be a parameter (i.e. a real number) in a table indexed by \( w_{i-n+1}, \ldots, w_i \).
Estimate conditional probabilities from \( n \)-gram counts in the training data \( \mathcal{D} \):

\[
P(w_2 \mid w_1) = \frac{\text{Count}_\mathcal{D}(w_1w_2)}{\text{Count}_\mathcal{D}(w_1)} \quad P(w_3 \mid w_1, w_2) = \frac{\text{Count}_\mathcal{D}(w_1w_2w_3)}{\text{Count}_\mathcal{D}(w_1w_2)}
\]

Why does this work?
Suppose we have a bigram model. Let $\theta$ be the parameters of this model, indexed by bigrams, so that $P(w_2 \mid w_1) = \theta_{w_1w_2}$.

The likelihood of the training data $\mathcal{D}$, as a function of the model parameters (bigram probabilities) is then:

$$P(\mathcal{D} \mid \theta) = \prod_{w_1w_2 \in V^2} \theta_{w_1w_2}^{\text{Count}_\mathcal{D}(w_1w_2)}$$

The maximum likelihood estimate chooses $\hat{\theta}$ such that

$$\hat{\theta} = \arg \max_{\theta} P(\mathcal{D} \mid \theta)$$
Counting $n$-grams maximizes likelihood

Suppose the word $white$ appears ten times, followed seven times by $house$ and three times by $whale$. Maximum likelihood sets $P(\text{house} \mid \text{white}) = \frac{7}{10}$. 
Estimating $n$-gram probabilities accurately is hard

- The higher $n$ gets, the better the model, if you have enough data.
- But most higher-order $n$-grams will never be observed—are these sampling zeros or structural zeros?
- Requires smoothing and/or backoff to estimate probabilities of unseen $n$-grams.
- Good models need to be trained on billions of words.
- This entails lots of memory and clever data structures.
You can use an $n$-gram LM to predict the next word

If we have a sequence of words $w_1 \ldots w_k$, then we can use the language model to predict the next word $w_{k+1}$:

$$\hat{w}_{k+1} = \arg\max_{w_{k+1}} P(w_{k+1}|w_1 \ldots w_k)$$

This is useful for applications that process input in real time (word-by-word).
Conditional language models
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So...
How would you model translation with \( n \)-grams?

Så varför minskar inte vi våra utsläpp?

So why
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?

Let $x$ be the Swedish sentence, $y$ be English.

$x = x_1 ... x_{|x|}$

$y = y_1 ... y_{|y|}$

How can we define $P(y|x)$?

Note: probabilistic machine translation models originated with French-English translation, and in older papers you will often see $f$ (for French) instead of $x$, and $e$ (for English) instead of $y$. In ML, $x$ and $y$ typically denote input and output, respectively, and are more common in current literature.
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we...
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?
How would you model translation with $n$-grams?

Let $x$ be the Swedish sentence, $y$ be English.

$x = x_1...x_{|x|}$

$y = y_1...y_{|y|}$

How can we define $P(y \mid x)$?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?
How would you model translation with \( n \)-grams?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?

Let \( x \) be the Swedish sentence, \( y \) be English.

\[
x = x_1 \ldots x_{|x|}
\]

\[
y = y_1 \ldots y_{|y|}
\]

How can we define \( P(y \mid x) \)?

Note: probabilistic machine translation models originated with French-English translation, and in older papers you will often see \( f \) (for French) instead of \( x \), and \( e \) (for English) instead of \( y \). In ML, \( x \) and \( y \) typically denote input and output, respectively, and are more common in current literature.
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp? So why are we not reducing our emissions?

What if we model translation as one long sequence?

$$P(yx) = P(x_1...x_{|x|}|y_1...y_{|y|})$$
How would you model translation with $n$-grams?

Så varför minskar inte vi våra utsläpp? So why are we not reducing our emissions?

What if we model translation as one long sequence?

$$P(yx) = P(x_1...x_{|x|}y_1...y_{|y|})$$

Problem: the English sentence will usually be longer than $n!$
How would you model translation with $n$-grams?

Så So varför why minskar are inte we vi not våra reducing utsläpp our emissions?

What if we alternate source and target words?

$$P(yx) = P(x_1y_1...x_{|x|}y_{|x|+1}...y_{|y|})$$
How would you model translation with $n$-grams?

Så So varför why minskar are inte we vi not våra reducing utsläpp our emissions?

What if we alternate source and target words?

$$P(yx) = P(x_1y_1...x_{|x|}y_{|x|}y_{|x|+1}...y_{|y|})$$

Problem 1: The sentences are not usually the same length!

Problem 2: English and Swedish word orders are different!
Could we use word alignments to model translation?

**Key idea:** we want to model bigram translation probabilities, like $P(\text{So} | \text{Så})$, $P(\text{why} | \text{varför})$, $P(\text{are} | \text{våra})$, and so on...
Could we use word alignments to model translation?

Key idea: we want to model bigram translation probabilities, like $P(\text{So} \mid \text{Så})$, $P(\text{why} \mid \text{varför})$, $P(\text{are} \mid \text{våra})$, and so on...

But this changes our model! If $x$ is Swedish and $y$ is English, we must now also model $z$, the alignment.
Could we use word alignments to model translation?

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?

**Key idea:** we want to model bigram *translation probabilities*, like $P(\text{So} \mid \text{Så})$, $P(\text{why} \mid \text{varför})$, $P(\text{are} \mid \text{våra})$, and so on...

*But this changes our model!* If $x$ is Swedish and $y$ is English, we must now also model $z$, the alignment.

We get $P(y \mid x) = \sum_z P(y, z \mid x)$ from the laws of probability.
Decompose $P(y, z \mid x)$ using the chain rule:

$$P(y, z \mid x) = P(y \mid x, z)P(z \mid x)$$

$$= P(|y|, |z| \mid x)$$

$$\prod_{i=1}^{\mid y \mid} P(y_i \mid y_1, \ldots, y_{i-1}, x, z) \prod_{i=1}^{\mid z \mid} P(z_i \mid z_1, \ldots, z_{i-1}, x)$$

**Note:** the chain rule is *always true* under the laws of probability. But as the modeler, you get to choose the order of the variables (since any order is valid).

The first term chooses the length of $y$ and $z$. We need to make some independence assumptions to simplify the other two terms into something we can work with.
Modeling English conditioned on Swedish with bigrams

Så varför minskar inte våra utsläpp?
Så varför minskar inte vi våra utsläpp?

Step 1. Draw length of English, conditioned on Swedish.

Full model: $P(|y| \mid x)$
Så varför minskar inte vi våra utsläpp?

**Step 1.** Draw length of English, conditioned on Swedish.

**Step 2.** For each English position, draw a Swedish word uniformly at random. Let $|z| = |y|$ and let $z_i$ be position of aligned Swedish word for $y_i$.

Full model: $P(|y| \mid x) \prod_{i=1}^{|y|} P(z_i \mid |x|)$
Modeling English conditioned on Swedish with bigrams

Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?

Step 1. Draw length of English, conditioned on Swedish.

Step 2. For each English position, draw a Swedish word uniformly at random. Let $|z| = |y|$ and let $z_i$ be position of aligned Swedish word for $y_i$.

Step 3. For each English word, draw its translation from a bigram translation probability.

Full model: $P(|y| \mid x) \prod_{i=1}^{\lfloor y \rfloor} P(z_i \mid |x|)P(y_i \mid x_{z_i})$
Så varför minskar inte vi våra utsläpp?

So why are we not reducing our emissions?

**Step 1.** Draw length of English, conditioned on Swedish.

**Step 2.** For each English position, draw a Swedish word uniformly at random. Let $|\mathbf{z}| = |\mathbf{y}|$ and let $z_i$ be position of aligned Swedish word for $y_i$.

**Step 3.** For each English word, draw its translation from a bigram translation probability.

**Full model:**

$$P(|\mathbf{y}| \mid \mathbf{x}) \prod_{i=1}^{\vert \mathbf{y} \vert} P(z_i \mid |\mathbf{x}|) P(y_i \mid x_{z_i})$$

Is this model familiar?
Modeling English conditioned on Swedish with bigrams

Input states: \{Så, varför, minskar, inte, vi, våra, utsläpp, ?\}

Input: So why are we not reducing our emissions?

Alternative view: each training example contains a set of states (Swedish words), and a sequence of English words that we tag with those states.
Modeling English conditioned on Swedish with bigrams

**Input states:** \{Så, varför, minskar, inte, vi, våra, utsläpp, ?\}

**Tags:** Så varför minskar vi inte minskar våra utsläpp

**Input:** So why are we not reducing our emissions

Alternative view: each training example contains a set of states (Swedish words), and a sequence of English words that we tag with those states.

This is just a (zero-order) hidden Markov model. You can also use higher order Markov models!

\[
P(\{y\} | x) = \prod_{i=1}^{|y|} P(z_i | x) \cdot P(y_i | x z_i)
\]

- Transition probability
- Emission probability
Counting expected alignments maximizes likelihood

Goal: estimate bigram translation probabilities, e.g. $P(\text{So} \mid \text{Så})$.

Problem: We can’t count, because the alignments are not in the data! In our model, $z$ is a latent variable (also called a hidden variable, unobserved variable, or nuisance variable). Let $\theta$ be the set of bigram parameters, and $P(y_i \mid x_j) = \theta_{x_j y_i}$. Maximum likelihood says:

$$\hat{\theta} = \arg \max_{\theta} P(D \mid \theta)$$

$$= \arg \max_{\theta} \sum_{x_j, y_i \in V^2} \theta_{x_j y_i} E_P(D \mid \theta) \left[ \text{Count}(x_j y_i) \right]$$

In words: use expected counts for unobserved events. Problem: to compute expected counts, we need to know $\theta$!
Counting expected alignments maximizes likelihood

Goal: estimate bigram translation probabilities, e.g. $P(\text{So} \mid \text{Så})$.

**Problem**: We can’t count, because the alignments are not in the data! In our model, $z$ is a *latent variable* (also called a hidden variable, unobserved variable, or nuisance variable).
Counting expected alignments maximizes likelihood

Goal: estimate bigram translation probabilities, e.g. \( P(\text{So} \mid \text{Så}) \).

**Problem:** We can’t count, because the alignments are not in the data! In our model, \( z \) is a *latent variable* (also called a hidden variable, unobserved variable, or nuisance variable).

Let \( \theta \) be the set of bigram parameters, and \( P(y_i \mid x_j) = \theta_{x_jy_i} \)

Maximum likelihood says:

\[
\hat{\theta} = \arg \max_{\theta} P(D \mid \theta) = \arg \max_{\theta} \prod_{x_j, y_i \in V^2} \theta_{x_jy_i} \mathbb{E}_{P(D \mid \theta)}[\text{Count}(x_jy_i)]
\]

In words: use *expected counts* for unobserved events.
Counting expected alignments maximizes likelihood

Goal: estimate bigram translation probabilities, e.g. $P(\text{So} \mid \text{Så})$.

**Problem:** We can’t count, because the alignments are not in the data! In our model, $z$ is a *latent variable* (also called a hidden variable, unobserved variable, or nuisance variable).

Let $\theta$ be the set of bigram parameters, and $P(y_i \mid x_j) = \theta_{x_jy_i}$

Maximum likelihood says:

$$\hat{\theta} = \arg \max_{\theta} P(\mathcal{D} \mid \theta)$$

$$= \arg \max_{\theta} \prod_{x_j, y_i \in V^2} \theta_{x_jy_i} \mathbb{E}_{P(\mathcal{D} \mid \theta)}[\text{Count}(x_jy_i)]$$

In words: use *expected counts* for unobserved events.

**Problem:** to compute expected counts, we need to know $\theta$!
Expectation maximization requires iteration

*Expectation maximization* iteratively improves an estimate of $\theta$:

1. Make an initial guess (random or uniform), called $\hat{\theta}_0$.
2. At iteration $i$, let $\hat{\theta}_i = \arg \max_{\theta} P(D \mid \theta_{i-1})$.

Likelihood is provably non-decreasing for each new estimate of $\theta$. 
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes' rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y) P(x \mid y)}{P(x)} \propto P(y)$$

The language model and translation model can be trained separately!

**Greedy search.** At time step $i$, predict $y_i$ that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.

**Beam search.** At time step $i$, keep the $k$ best $y_i$'s that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.

Greedy/beam search don't find optimal $y$ according to $P(y \mid x)$!
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y | x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y | x) = \frac{P(x, y)}{P(x)} = \frac{P(y) P(x | y)}{P(x)} \propto P(y | \{z\})$$

The language model and translation model can be trained separately!

**Greedy search.** At time step $i$, predict $y_i$ that maximizes $P(y_i | y_1, ..., y_{i-1}, x)$.

**Beam search.** At time step $i$, keep the $k$ best $y_i$’s that maximizes $P(y_i | y_1, ..., y_{i-1}, x)$.

Greedy/beam search don’t find optimal $y$ according to $P(y | x)$!
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} =$$
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y)P(x \mid y)}{P(x)}$$
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$
P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y)P(x \mid y)}{P(x)} \propto \underbrace{P(y)}_{\text{language model}} \underbrace{P(x \mid y)}_{\text{translation model}}
$$

The language model and translation model can be trained separately!
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y)P(x \mid y)}{P(x)} \propto \underbrace{P(y)}_{\text{language model}} \underbrace{P(x \mid y)}_{\text{translation model}}$$

The language model and translation model can be trained separately!

**Greedy search.** At time step $i$, predict $y_i$ that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y)P(x \mid y)}{P(x)} \propto P(y) P(x \mid y)$$

The language model and translation model can be trained separately!

**Greedy search.** At time step $i$, predict $y_i$ that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.

**Beam search.** At time step $i$, keep the $k$ best $y_i$’s that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$. 

Greedy/beam search don’t find optimal $y_i$ according to $P(y \mid x)$!
Decoding with (conditional) language models

**Question.** What is the most probable string, according to a language model $P(w)$, or a conditional language model $P(y \mid x)$?

**Note.** With conditional language models, we often use Bayes’ rule:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} = \frac{P(y)P(x \mid y)}{P(x)} \propto \frac{P(y)}{P(x \mid y)}$$

The language model and translation model can be trained separately!

**Greedy search.** At time step $i$, predict $y_i$ that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.

**Beam search.** At time step $i$, keep the $k$ best $y_i$’s that maximizes $P(y_i \mid y_1, \ldots, y_{i-1}, x)$.

Greedy/beam search don’t find optimal $y$ according to $P(y \mid x)$!
Why care about $n$-grams? Aren’t they obsolete?

1. Many of these ideas turn up again in neural models.
   - All machine learning maximizes some objective function.
   - Neural models still use beam search.
   - Latent variables are common in unsupervised learning.
   - Alignment directly inspired neural attention.
   - Neural models exploit same signals, though more powerful.

2. Older models are still often useful in low-data settings.

3. An extension of the model in this lecture translates $n$-grams to $n$-grams: phrase-based translation.
   - It is still used by Google for some languages, despite move to neural MT in 2017.

4. Understanding the tradeoffs of working with Markov assumptions will help you appreciate the fact that neural models usually make them go away!
n-gram models exemplify many key concepts in ML for NLP

Why care about n-grams? Aren’t they obsolete?

1. Many of these ideas turn up again in neural models.
   - All machine learning maximizes some objective function.
   - Neural models still use beam search.
   - Latent variables are common in unsupervised learning.
   - Alignment directly inspired neural attention.
   - Neural models exploit same signals, though more powerful.
Why care about $n$-grams? Aren’t they obsolete?

1. Many of these ideas turn up again in neural models.
   • All machine learning maximizes some **objective function**.
   • Neural models still use **beam search**.
   • Latent variables are common in **unsupervised learning**.
   • Alignment directly inspired neural **attention**.
   • Neural models exploit same signals, though more powerful.

2. Older models are still often useful in low-data settings.
Why care about $n$-grams? Aren’t they obsolete?

1. Many of these ideas turn up again in neural models.
   - All machine learning maximizes some objective function.
   - Neural models still use beam search.
   - Latent variables are common in unsupervised learning.
   - Alignment directly inspired neural attention.
   - Neural models exploit same signals, though more powerful.

2. Older models are still often useful in low-data settings.

3. An extension of the model in this lecture translates $n$-grams to $n$-grams: phrase-based translation. It is still used by Google for some languages, despite move to neural MT in 2017.
Why care about $n$-grams? Aren’t they obsolete?

1. Many of these ideas turn up again in neural models.
   • All machine learning maximizes some *objective function*.
   • Neural models still use *beam search*.
   • Latent variables are common in *unsupervised learning*.
   • Alignment directly inspired neural *attention*.
   • Neural models exploit same signals, though more powerful.

2. Older models are still often useful in low-data settings.

3. An extension of the model in this lecture translates $n$-grams to $n$-grams: *phrase-based translation*. It is still used by Google for some languages, despite move to neural MT in 2017.

4. Understanding the tradeoffs of working with *Markov assumptions* will help you appreciate the fact that neural models usually make them go away!
• Language models assign probabilities to discrete sequences.
• Useful for natural language generation in many applications.
• $n$-gram models use a Markov assumption to model an infinite sample space with a finite set of parameters.
• Machine translation is just conditional language modeling.
• To effectively model translation with $n$-grams, we need additional latent variables to model word alignment.
• One way to estimate the parameters of latent variable models is with a generalization of maximum likelihood estimation, called expectation maximization.
Next Week

• Feedforward NN
• Recurrent NN
• How to format the input and output data
• Assignment will be out next week.