Natural Language Understanding, Generation, and Machine Translation

Lecture 3: Conditional Language Modeling with *n*-grams

Adam Lopez

17 January 2019

School of Informatics
University of Edinburgh
alopez@inf.ed.ac.uk

Administrative update

- More guidance about prerequisites on piazza and in learn. **Take this seriously.**
- My office hours: Thursdays 1-2pm in Absorb Cafe (in Appleton Tower by the lecture theatres).
- Please sign up for piazza.

Revision

- Language models
- n-gram Language models

Conditional language models

Required, optional, and revision readings are listed on learn.

Revision

Predict the next word!

Predict the next word!

Summer is hot winter is _____

Predict the next word!

She is drinking a hot cup of _____

Predict the next word!

In the park I saw a _____

Predict the next word!

In the park I saw a _____

Image captioning

A language model is a probabilistic model of strings

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 strings *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

Applications of Language Modeling

Machine translation:

- word ordering: $P(\text{the cat is small}) > P(\text{small the is cat})$;
- word choice: $P(\text{walking home after school}) > P(\text{walking house after school})$.

Grammar checking:

- word substitutions: $P(\text{the principal resigned}) > P(\text{the principle resigned})$;
- agreement errors: $P(\text{the cats sleep in the basket}) > P(\text{the cats sleeps in the basket})$.

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 may not be identical.

Expect notation to be **internally consistent** in an individual lecture or paper.

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.

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 may not be identical.

Expect notation to be **internally consistent** in an individual lecture or paper.

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 version: notation is a kind of language.

Language modeling as probabilistic prediction

Given a finite vocabulary V, we want to define a probability distribution $P : V^* \to \mathbb{R}_+$.

Language modeling as probabilistic prediction

Given a finite vocabulary V, we want to define a probability distribution $P : V^* \to \mathbb{R}_+$.

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 $P : V^* \to \mathbb{R}_+$.

The *finite vocabulary* bit should worry you. We'll come back to this, but not today!

Revision questions:

- What is the sample space? strings that have any length consisting the symbols of vocabulary V
- What might be some useful random variables?
- What constraints must P satisfy? 0<P<1, the sum of output is 1

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 ith 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 ith 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 ith word. So, $w = w_1 \dots w_{|w|}$.

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

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

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 ith word. So, $w = w_1 \dots w_{|w|}$.

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

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

Use the chain rule:

$$
\begin{aligned}
P(w_1 \dots w_{|w|}) = & P(W_1 = w_1) \times \\
& P(W_2 = w_2 \mid W_1 = w_1) \times \\
& \dots \\
& P(W_{|w|} = w_{|w|} \mid W_1 = w_1, \dots, W_{k-1} = w_{|w|-1}) \\
& P(W_{|w|+1} = \langle \textsc{Stop} \rangle \mid W_1 = w_1, \dots, W_k = w_{|w|})
\end{aligned}
$$

Note: $\langle \textsc{Stop} \rangle$ is a symbol not in V.

Written more concisely

$$
\begin{aligned}
P(w_1 \ldots w_{|w|}) = & P(w_1)\times \\
& P(w_2 \mid w_1)\times \\
& \ldots \\
& P(w_{|w|} \mid w_1, \ldots, w_{|w|-1}) \\
& P(\langle \textsc{Stop} \rangle \mid w_1, \ldots, w_{|w|}) \\
= & \prod_{i=1}^{|w|+1} P(w_i | w_1, \ldots, w_{|w|})
\end{aligned}
$$

Defines a *joint distribution* over infinite sample space in terms of *conditional distributions*, each over finite sample spaces (but with potentially infinite history!)

Written more concisely

$$
\begin{aligned}
P(w_1 \dots w_{|w|}) = & P(w_1) \times \\
& P(w_2 \mid w_1) \times \\
& \dots \\
& P(w_{|w|} \mid w_1, \dots, w_{|w|-1}) \\
& P(\langle \textsc{Stop} \rangle \mid w_1, \dots, w_{|w|}) \\
= & \prod_{i=1}^{|w|+1} P(w_i | w_1, \dots, w_{|w|})
\end{aligned}
$$

Written more concisely

$$
\begin{aligned}
P(w_1 \dots w_{|w|}) = & P(w_1) \times \\
& P(w_2 \mid w_1) \times \\
& \dots \\
& P(w_{|w|} \mid w_1, \dots, w_{|w|-1}) \\
& P(\langle \textsc{Stop} \rangle \mid w_1, \dots, w_{|w|}) \\
= & \prod_{i=1}^{|w|+1} P(w_i | w_1, \dots, w_{|w|})
\end{aligned}
$$

Defines a *joint distribution* over infinite sample space in terms of *conditional distributions*, each over finite sample spaces (but with potentially infinite history!)

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

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

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

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

$$P(w_i \mid w_1, \ldots, w_{i-1}) \sim P(w_i \mid w_{\color{red}{i-n+1}}, \ldots, w_{i-1})$$

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 $\qquad P : V \to \mathcal{R}_+$

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 $\quad P : V \to \mathcal{R}_+$

... and all sum to one $\quad \sum_{w \in V} P(w \mid w_{i-n+1}, \ldots, w_{i-1}) = 1$

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 $\quad P : V \rightarrow \mathcal{R}_+$

... 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! Let's define one.

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	$P : V \rightarrow \mathcal{R}_+$
... 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! Let's define one.

Simplest 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 $w_{i-n+1}, \ldots, w_i$.

n-gram probabilities can be estimated by counting

We can get maximum likelihood estimates for the conditional probabilities from n-gram counts in a corpus:

$$P(w_2|w_1) = \frac{n_{(w_1,w_2)}}{n_{(w_1)}} \qquad P(w_3|w_1,w_2) = \frac{n_{(w_1,w_2,w_3)}}{n_{(w_1,w_2)}}$$

But building good n-gram language models can be difficult:

- the higher the n, the better the performance
- but most higher-order n-grams will never be observed—are these *sampling zeros* or *structural zeros*? most is sampling zero
- good models need to be trained on billions of words
- this entails large memory requirements
- smoothing and backoff techniques are required.

sampling zero: it appears but not sampled
structural zero: it shouldn't appear in the language

Using n-gram Language Models

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

$$\hat{w}_{k+1} = \underset{w_{k+1}}{\operatorname{argmax}} P(w_{k+1} | w_1 \dots w_k)$$

Being able to predict the next word is useful for applications that process input in real time (word-by-word).

Conditional language models

$p(yellow)$ $\quad$ $1 - p(yellow)$

$p(yellow)?$

$p(yellow)$?

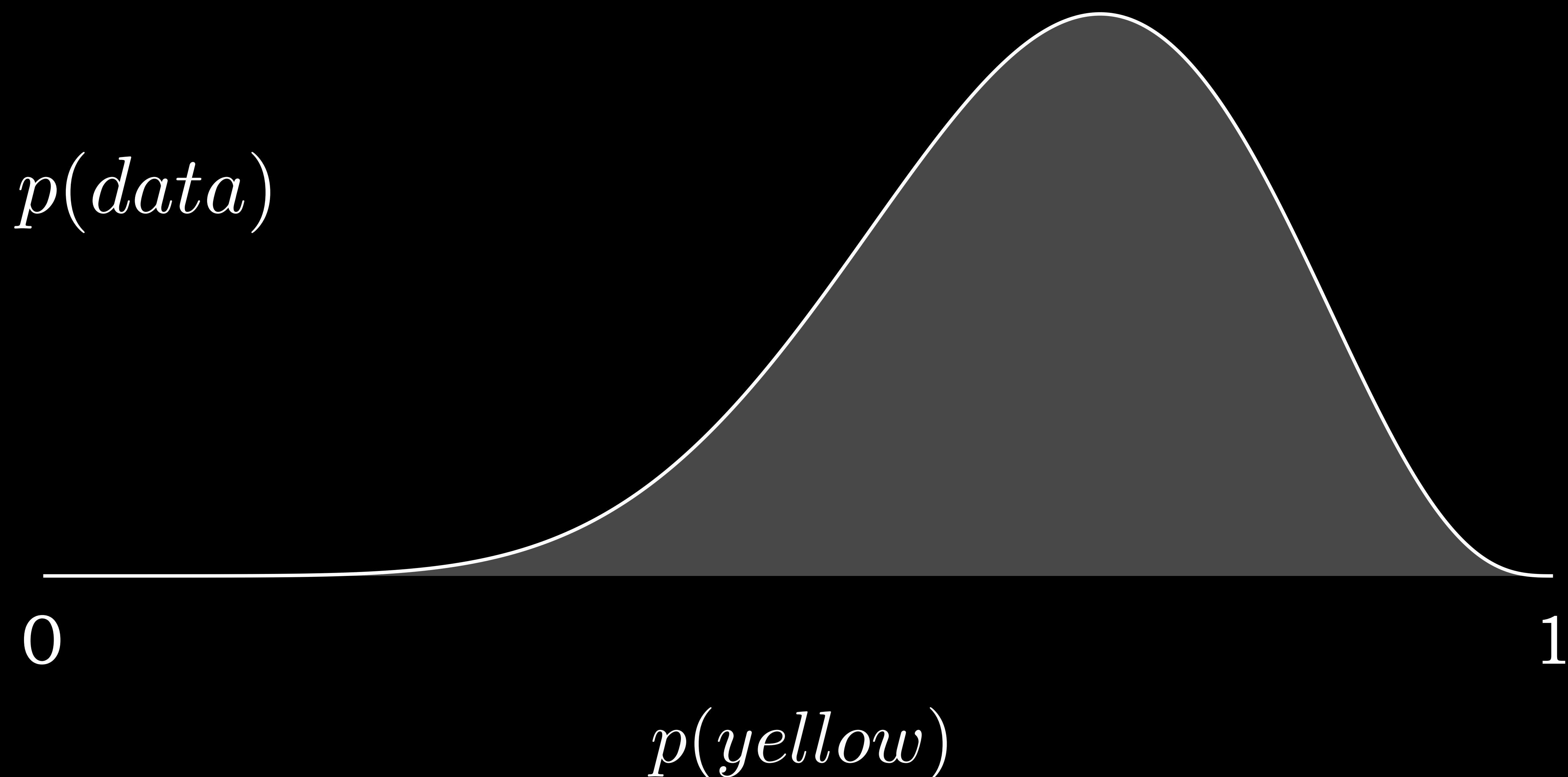
 $p(yellow)?$

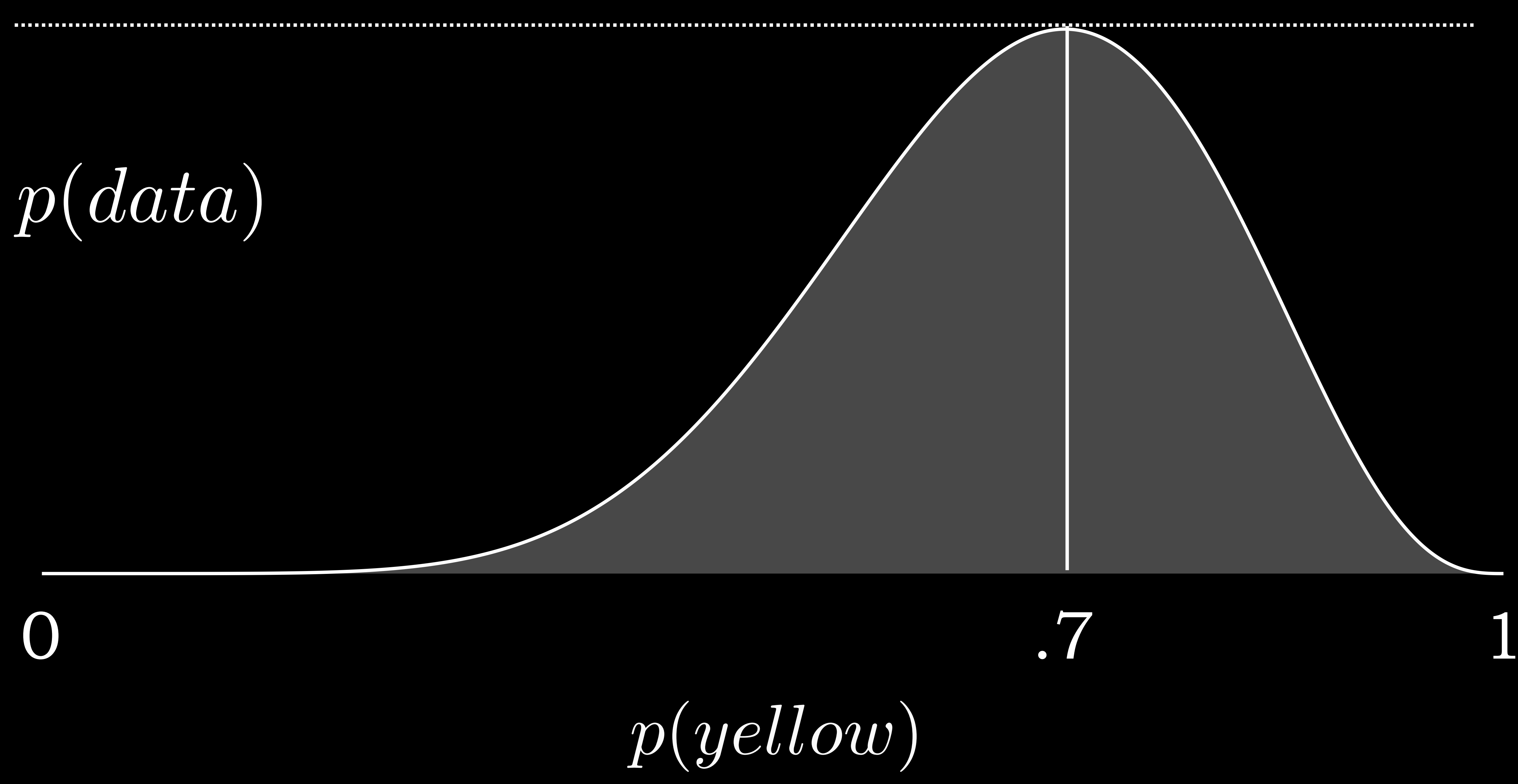
$$p(data) = p(yellow)^7 \times [1 - p(yellow)]^3$$

 $p(yellow)?$

$$p(data) = p(yellow)^7 \times [1 - p(yellow)]^3$$

Maximum likelihood chooses parameters to maximize this function (called the likelihood).


$p(data)$
0
1
$p(yellow)$


$p(data)$
0
.7
1
$p(yellow)$

Machine Translation

This is just a **conditional** language model.
It generates Chinese, conditioned on English.

Question: Could we use *n*-gram models here?

Conditional LMs

Given Chinese word sequence $f = f_1 \ldots f_{|f|}$
predict English word sequence $e = e_1 \ldots e_{|f|}$

Conditional LMs

Given Chinese word sequence $f = f_1 \dots f_{|f|}$
predict English word sequence $e = e_1 \dots e_{|f|}$

Will this work?

Let $w = f_1 \dots f_{|f|} e_1 \dots e_{|e|}$

Conditional LMs

Given Chinese word sequence $f = f_1 \ldots f_{|f|}$
predict English word sequence $e = e_1 \ldots e_{|f|}$

Will this work?

Let $w = f_1 \ldots f_{|f|} e_1 \ldots e_{|e|}$

Problem: model forgets Chinese sentence after generating first n-1 words of English

Conditional LMs

Given Chinese word sequence $f = f_1 \ldots f_{|f|}$
predict English word sequence $e = e_1 \ldots e_{|f|}$

What about this?

Let $w = f_1 e_1 \ldots f_{|f|} e_{|f|} \ldots e_{|e|}$

Conditional LMs

Given Chinese word sequence $f = f_1 \ldots f_{|f|}$
predict English word sequence $e = e_1 \ldots e_{|f|}$

What about this?

Let $w = f_1 e_1 \ldots f_{|f|} e_{|f|} \ldots e_{|e|}$

Problem: sentences might not be in the same length or have the same word order.

Conditional LMs

General problem: *n*-grams condition on finite history.

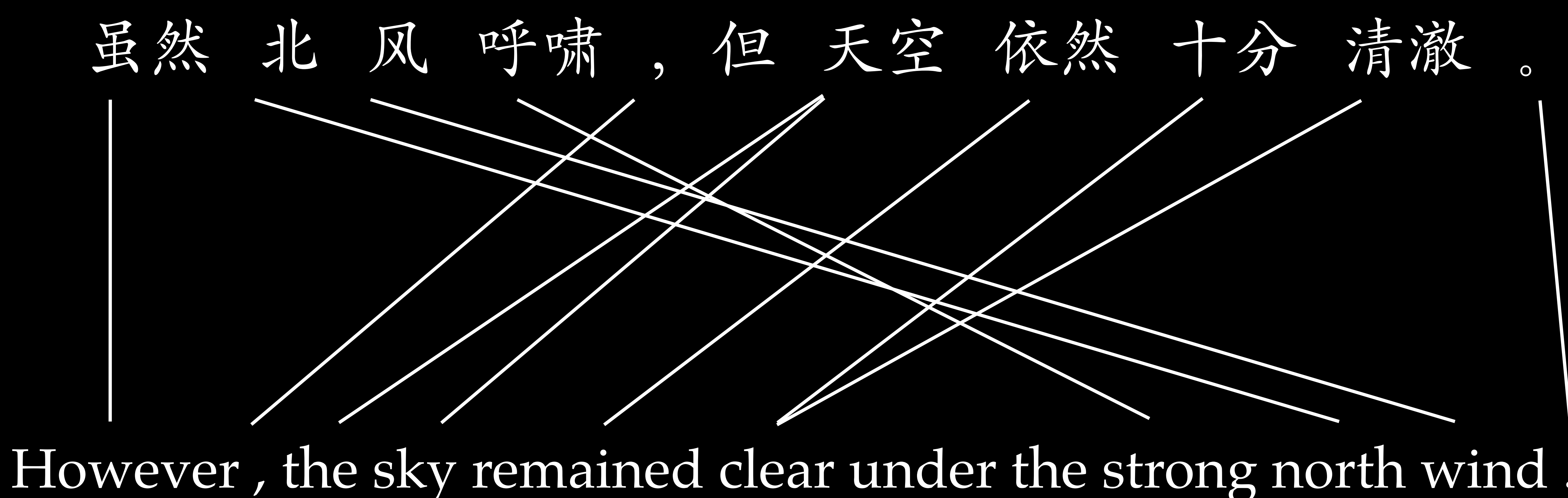
When we are generating an English word, how do we know which part of the history to condition on?

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。

However , the sky remained clear under the strong north wind .

Conditional LMs

Word alignments!



IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。

However , the sky remained clear under the strong north wind .

IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。

However , the sky remained clear under the strong north wind .

Let’s write a simple model in terms of word-to-word alignments:

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e})$$

IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

IBM Model 1

虽然 北 风 呼啸 ，但 天空 依然 十分 清澈 。ε

___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

predict English length given Chinese length

$$p(English\ length|Chinese\ length)$$

IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

1. which Chinese word is in this position
2. on the condition the Chinese word I chose , which English word to generate
3. Concern about the unknown words

___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

$$p(Chinese\ word\ position)$$

IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___

However

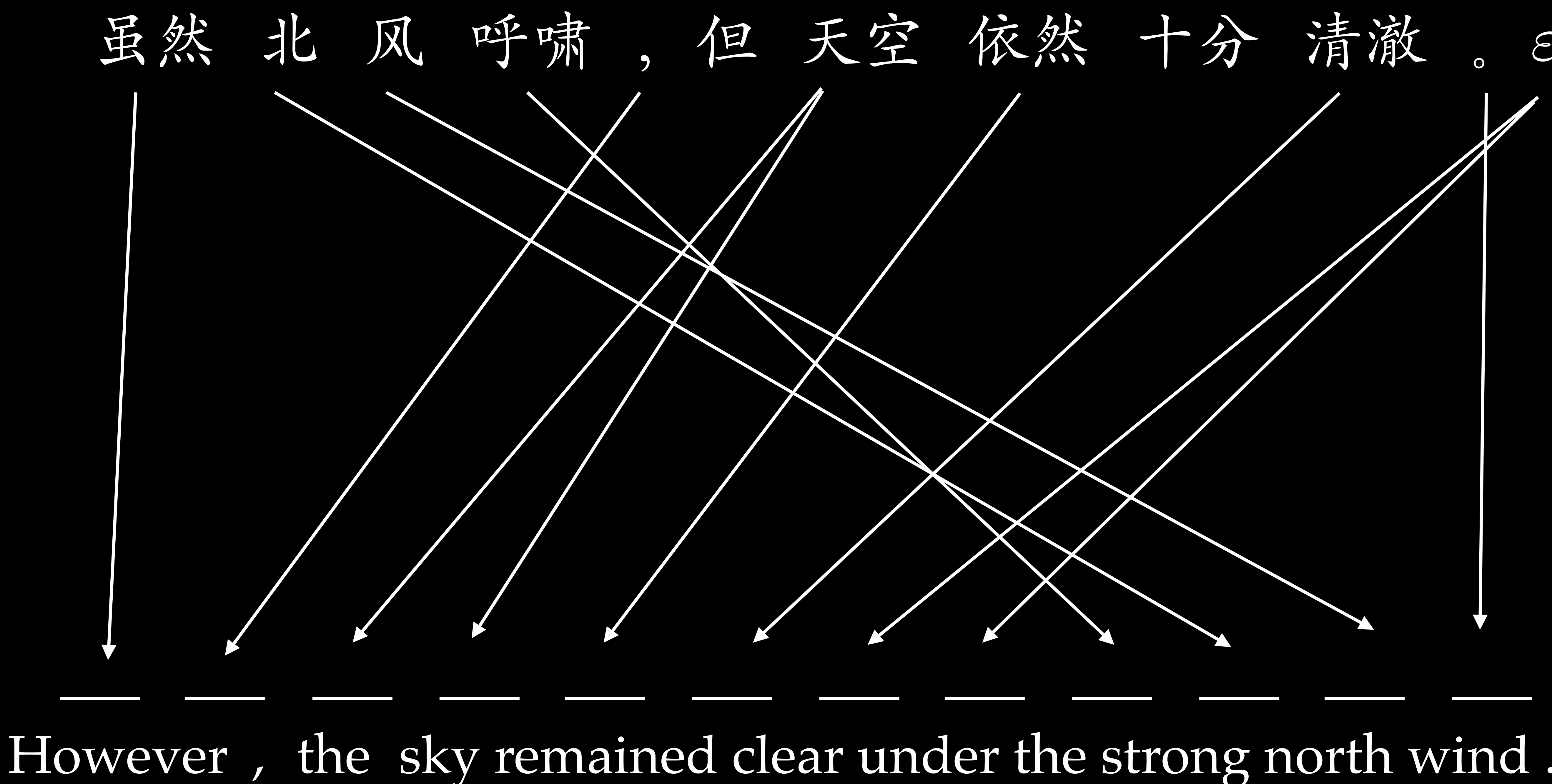
$$p(English\ word|Chinese\ word)$$

IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

However ,

IBM Model 1


虽然 北 风 呼啸 ，但 天空 依然 十分 清澈 。ε
However ， the sky remained clear under the strong north wind .

IBM Model 1

alignment of French
word at position i

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

French, English
sentence lengths

French, English
word pair

IBM Model 1

alignment of French
word at position i

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

French, English
sentence lengths

French, English
word pair

The alignment is a **latent variable** whose value is a sequence over Chinese word positions: $\{1, \ldots, |f|\}^{|e|}$

IBM Model 1

alignment of French
word at position i

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

French, English
sentence lengths

French, English
word pair

Does this equation look familiar?

IBM Model 1

transition to state at
position i

$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

emission at position i

Just a zero-order HMM!

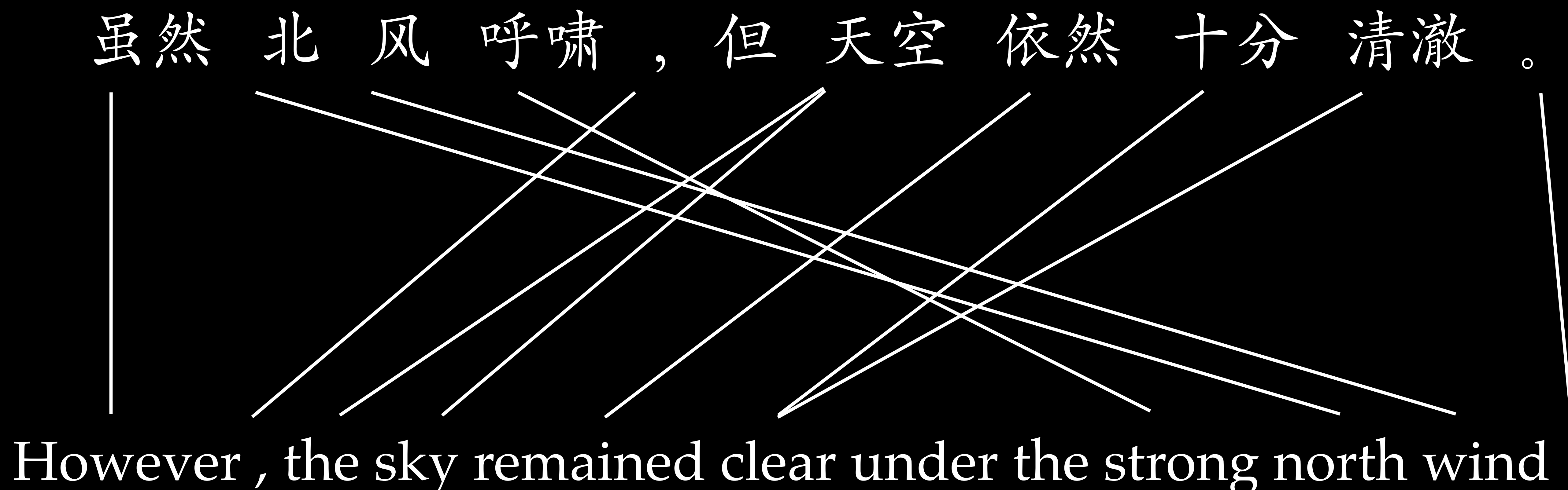
Only difference from standard HMM: set of tags (Chinese words) varies for each sentence.

IBM Model 1

$$\theta \begin{cases} p(despite|\text{虽然}) & ??? \\ p(however|\text{虽然}) & ??? \\ p(although|\text{虽然}) & ??? \\ \dots & \\ p(northern|\text{北}) & ??? \\ p(north|\text{北}) & ??? \end{cases}$$

Where do these parameters come from?

MLE for IBM Model 1



$$p(\text{north} \mid \text{北}) = \frac{\text{\# of times 北 aligns to “north”}}{\text{\# of times 北 aligns to any word}}$$

MLE for IBM Model 1

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。

However , the sky remained clear under the strong north wind .

$$p(\text{north} \mid \text{北}) = \frac{???}{???}$$

Problem: We do not get to observe the word alignments!

Expectation Maximization

虽然 北 风 呼啸 , 但 天空 依然 十分 清澈 。

However , the sky remained clear under the strong north wind .

$$p(\text{north} \mid \text{北}) = \frac{\textit{Expected}\ \#\ \text{of times 北 aligns to "north"}}{\textit{Expected}\ \#\ \text{of times 北 aligns to any word}}$$

The same maths as for MLE leads to this.

What are expected counts?

虽然 北 风 呼啸 ，但 天空 依然 十分 清澈 。ε

However ， the sky remained clear under the strong north wind .

What are expected counts?

虽然 北 风 呼啸 ，但 天空 依然 十分 清澈 。ε

if we had observed the alignment, this line would either be here (count 1) or it wouldn't (count 0).

However ， the sky remained clear under the strong north wind .

What are expected counts?

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

if we had observed the alignment, this line would either be here (count 1) or it wouldn’t (count 0).

since we didn’t observe the alignment, we calculate the probability that it’s there.

However ， the sky remained clear under the strong north wind .

What are expected counts?

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

if we had observed the alignment, this line would either be here (count 1) or it wouldn't (count 0).

since we didn't observe the alignment, we calculate the probability that it's there.

However ， the sky remained clear under the strong north wind .

But we need model parameters to compute this!

EM: the main idea

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

Parameters and alignments are both unknown.

However , the sky remained clear under the strong north wind .

$p(English\ word|Chinese\ word)$ unobserved!

EM: the main idea

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

Parameters and alignments are both unknown.

If we knew the alignments, we could calculate the values of the parameters.

However , the sky remained clear under the strong north wind .

$p(English\ word|Chinese\ word)$ unobserved!

EM: the main idea

虽然 北 风 呼啸 ，但 天空 依然 十分 清澈 。ε

Parameters and alignments are both unknown.

If we knew the alignments, we could calculate the values of the parameters.

If we knew the parameters, we could calculate the likelihood of the data.

However ， the sky remained clear under the strong north wind .

$p(English\ word|Chinese\ word)$ unobserved!

EM: the main idea

虽然 北 风 呼啸 ， 但 天空 依然 十分 清澈 。 ε

Parameters and alignments are both unknown.

If we knew the alignments, we could calculate the values of the parameters.

If we knew the parameters, we could calculate the likelihood of the data.

However ， the sky remained clear under the strong north wind .

$p(English\ word|Chinese\ word)$ unobserved!

The Plan: Bootstrapping

- Arbitrarily select a set of parameters (say, uniform).
- Calculate ***expected counts*** of the unseen events.
- Choose new parameters to maximize likelihood, using expected counts as proxy for observed counts.
- Iterate.

Computing expected counts

- Main computational bottleneck.
- For this model: dynamic programming, specifically the forward-backward algorithm
  - This is a special case of backpropagation!
- For most models: sample for while, then compute a Monte Carlo estimate of the expected counts.

Why EM works

Observation 1: We are still solving a maximum likelihood estimation problem.

Why EM works

Observation 1: We are still solving a maximum likelihood estimation problem.

$$p(Chinese|English) = \sum_{alignments} p(Chinese, alignment|English)$$

Why EM works

Observation 1: We are still solving a maximum likelihood estimation problem.

$$p(Chinese|English) = \sum_{alignments} p(Chinese, alignment|English)$$

MLE: choose parameters that maximize this expression.

Why EM works

Observation 1: We are still solving a maximum likelihood estimation problem.

$$p(Chinese|English) = \sum_{alignments} p(Chinese, alignment|English)$$

MLE: choose parameters that maximize this expression.

Minor problem: there is no analytic solution.

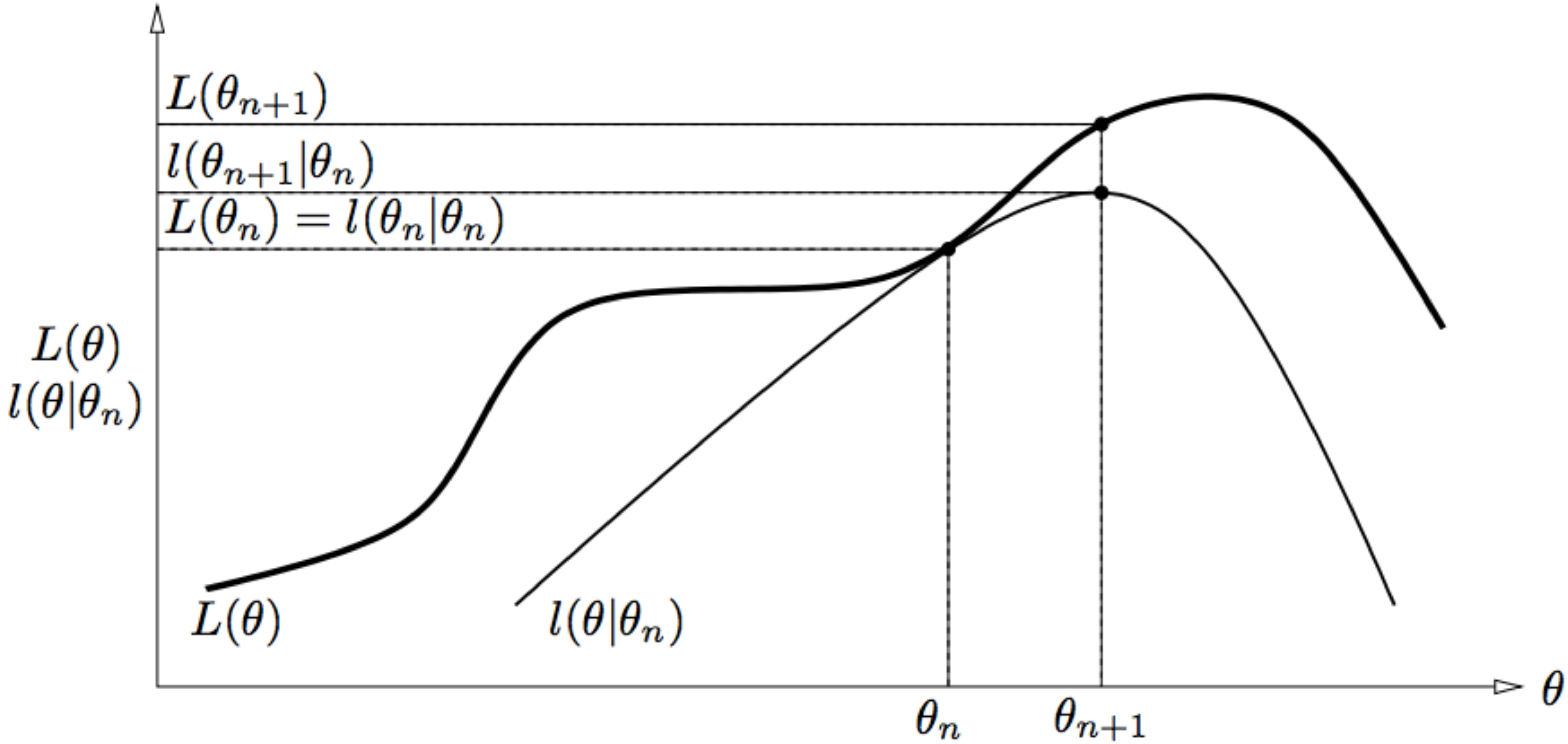


Figure 2: Graphical interpretation of a single iteration of the EM algorithm: The function $l(\theta|\theta_n)$ is bounded above by the likelihood function $L(\theta)$. The functions are equal at $\theta = \theta_n$. The EM algorithm chooses θ_{n+1} as the value of θ for which $l(\theta|\theta_n)$ is a maximum. Since $L(\theta) \geq l(\theta|\theta_n)$ increasing $l(\theta|\theta_n)$ ensures that the value of the likelihood function $L(\theta)$ is increased at each step.

(from Boorman '04)

Decoding

Once we have a model, we want to solve:

Given Chinese word sequence $f = f_1 \ldots f_{|f|}$
predict *most probable* English word sequence

- Doing this correctly involves Bayesian reasoning and NP-hard algorithms.
- Generally uses approximations (beam search).
- Will discuss similar approximations later in the course for neural MT.

Summary of key points (i.e. examinable content)

- Language models assign string probabilities
- Useful for word prediction in many NLP applications
- n-gram models use a Markov assumption to partition the infinite set of possible histories into a finite set of finite set of states, each with its own parameters.
- Machine translation is conditional language modeling.
- To 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*.