CMSC 473/673
Natural Language Processing

Instructor: Lara J. Martin (she/they)

TA: Duong Ta (he)

Slides modified from Dr. Frank Ferraro

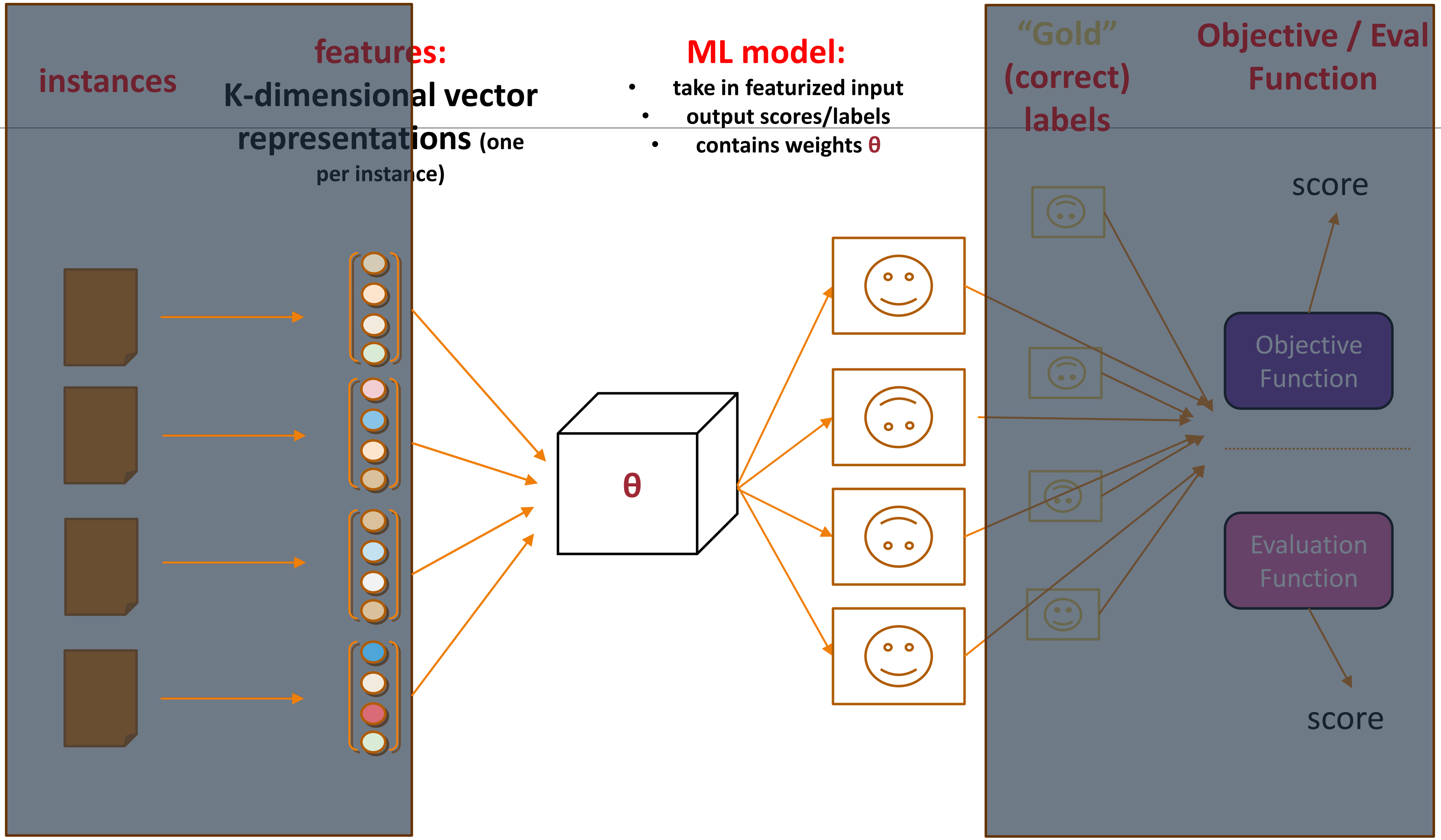
Learning Objectives

Code a LM using Maximum Likelihood Estimation (MLE)

Evaluate LMs with perplexity

Create a LM using smoothed counts

Defining the Model



Review: Goal of Language Modeling

$$p_\theta(\ \textit{[...text..]}\ )$$

Learn a probabilistic model of text

Accomplished through observing text and updating model parameters to make text more likely

Review: What Part of Language Do We Estimate?

$$p_\theta(\ \textit{[...text..]}\ )$$

Is *[...text..]* a

- Full document?
- Sequence of sentences?
- Sequence of words?
- Sequence of characters?

A: It's task-dependent!

Review: Probability Chain Rule

$$p(x_1, x_2, \dots, x_S) =$$
$$p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_S \mid x_1, \dots, x_{S-1}) =$$
$$\prod_i^S \underbrace{p(x_i \mid x_1, \dots, x_{i-1})}$$

Language modeling is about how to estimate each of these factors in {great, good, sufficient, …} ways

Language Models & Smoothing

Maximum likelihood (MLE): simple counting

Other count-based models
- Laplace smoothing, add- λ ← Easy to implement
- Interpolation models
- Discounted backoff
- Interpolated (modified) Kneser-Ney
- Good-Turing
- Witten-Bell

Advanced/ out of scope

Maxent n-gram models ← Featureful LMs

Neural n-gram models ← Feedforward LMs

Recurrent/autoregressive NNs ← Precursor to modern LMs

Review: Trigram Chaining

p(Colorless green ideas sleep furiously) =
p(Colorless | *<BOS> <BOS>*) *
p(green | *<BOS>* Colorless) *
p(ideas | Colorless green) *
p(sleep | green ideas) *
p(furiously | ideas sleep) *
p(*<EOS>* | sleep furiously)

Consistent notation: Pad the left with <BOS> (beginning of sentence) symbols
Fully proper distribution: Pad the right with a single <EOS> symbol

Review: N-Gram Probability

$$p(w_1, w_2, w_3, \cdots, w_S) =$$

$$\prod_{i=1}^{S} p(w_i | w_{i-N+1}, \cdots, w_{i-1})$$

Review: Count-Based N-Grams (Unigrams)

word type

word type

$$p(\mathrm{z}) \cong count(\mathrm{z})$$

$$= \frac{count(\mathrm{z})}{W}$$

number of tokens observed

Review: Count-Based N-Grams (Trigrams)

$$p(\mathrm{z}|\mathrm{x},\mathrm{y}) = \frac{count(\mathrm{x},\mathrm{y},\mathrm{z})}{\sum_{v} count(\mathrm{x},\mathrm{y},\mathrm{v})}$$

Knowledge Check: Make a Trigram LM

$$p(\mathrm{z}|\mathrm{x},\mathrm{y}) = \frac{count(\mathrm{x},\mathrm{y},\mathrm{z})}{\sum_{v} count(\mathrm{x},\mathrm{y},\mathrm{v})}$$

Review: Evaluating Language Models

What is “correct?”

What is working “well?”

Extrinsic: Evaluate LM in downstream task

Test an MT, ASR, etc. system and see which LM does better

Issue: Propagate & conflate errors

Intrinsic: Treat LM as its own downstream task

Use perplexity (from information theory)

Review: Perplexity

Lower is better : lower perplexity ➔ less surprised

$$\text{perplexity} = \exp(\text{avg crossentropy})$$

$$\text{perplexity} = \exp\left(\frac{-1}{M}\sum_{i=1}^{M} \log p(w_i \mid h_i)\right)$$

Example perplexity for trigram model

Trigrams	MLE p(trigram)
<BOS> <BOS> The	1
<BOS> The film	1
The film ,	0
film , a	0
, a hit	0
a hit !	0
hit ! <EOS>	0
Perplexity	???

"The film , a hit !"

$$\text{perplexity} = \exp\left(\frac{-1}{M}\sum_{i=1}^{M}\log p(w_i \mid h_i)\right)$$

Example perplexity for trigram model

Trigrams	MLE p(trigram)
<BOS> <BOS> The	1
<BOS> The film	1
The film ,	0
film , a	0
, a hit	0
a hit !	0
hit ! <EOS>	0
Perplexity	Infinity

"The film , a hit !"

$$\text{perplexity} = \exp\left(\frac{-1}{M}\sum_{i=1}^{M}\log p(w_i \mid h_i)\right)$$

Example perplexity for trigram model

Trigrams	MLE p(trigram)	Smoothed p(trigram)
<BOS> <BOS> The	1	2/17
<BOS> The film	1	2/17
The film ,	0	1/17
film , a	0	1/16
, a hit	0	1/16
a hit !	0	1/17
hit ! <EOS>	0	1/16
Perplexity	Infinity	???

"The film , a hit !"

$$\text{perplexity} = \exp\left(\frac{-1}{M}\sum_{i=1}^{M}\log p(w_i \mid h_i)\right)$$

Example perplexity for trigram model

Trigrams	MLE p(trigram)	Smoothed p(trigram)
<BOS> <BOS> The	1	2/17
<BOS> The film	1	2/17
The film ,	0	1/17
film , a	0	1/16
, a hit	0	1/16
a hit !	0	1/17
hit ! <EOS>	0	1/16
Perplexity	Infinity	13.59

"The film , a hit !"

$$\text{perplexity} = \exp\left(\frac{-1}{M}\sum_{i=1}^{M}\log p(w_i \mid h_i)\right)$$

0s Are Not Your (Language Model's) Friend

$$p(\text{item}) \cong count(\text{item}) = 0 \rightarrow p(\text{item}) = 0$$

0 probability → item is *impossible*

0s annihilate: x*y*z*0 = 0

Language is creative:

new words keep appearing

existing words could appear in known contexts

How much do you trust your data?

Language Models & Smoothing

~~Maximum likelihood (MLE): simple counting~~

Other count-based models
- **Laplace smoothing, add- λ** ← Easy to implement
- Interpolation models
- Discounted backoff
- Interpolated (modified) Kneser-Ney
- Good-Turing
- Witten-Bell

Advanced/ out of scope

Maxent n-gram models ← Featureful LMs

Neural n-gram models ← Feedforward LMs

Recurrent/autoregressive NNs ← Precursor to modern LMs

Add-λ estimation

Other names: Laplace smoothing, Lidstone smoothing

Pretend we saw each word λ more times than we did

$$p(\mathrm{z}) \cong count(\mathrm{z}) + \lambda$$

Add λ to all the counts

Add-λ estimation

Other names: Laplace smoothing, Lidstone smoothing

Pretend we saw each word λ more times than we did

Add λ to all the counts

$$p(\mathrm{z}) \cong count(\mathrm{z}) + \lambda$$

$$= \frac{count(\mathrm{z}) + \lambda}{\sum_{v}(count(\mathrm{v}) + \lambda)}$$

Add-λ estimation

Other names: Laplace smoothing, Lidstone smoothing

Pretend we saw each word λ more times than we did

Add λ to all the counts

$$p(\mathrm{z}) \cong count(\mathrm{z}) + \lambda$$

$$= \frac{count(\mathrm{z}) + \lambda}{\underbrace{W}_{\#\ \text{tokens}} + \underbrace{V}_{\#\ \text{types}}\lambda}$$

Add-λ N-Grams (Unigrams)

The film got a great opening and the film went on to become a hit .

Word (Type)	Raw Count	Norm	Prob.	Add-λ Count	Add-λ Norm.	Add-λ Prob.
The	1	16	1/16			
film	2		1/8			
got	1		1/16			
a	2		1/8			
great	1		1/16			
opening	1		1/16			
and	1		1/16			
the	1		1/16			
went	1		1/16			
on	1		1/16			
to	1		1/16			
become	1		1/16			
hit	1		1/16			
.	1		1/16			

Add-1 N-Grams (Unigrams)

The film got a great opening and the film went on to become a hit .

Word (Type)	Raw Count	Norm	Prob.	Add-1 Count	Add-1 Norm.	Add-1 Prob.
The	1	16	1/16	2		
film	2		1/8	3		
got	1		1/16	2		
a	2		1/8	3		
great	1		1/16	2		
opening	1		1/16	2		
and	1		1/16	2		
the	1		1/16	2		
went	1		1/16	2		
on	1		1/16	2		
to	1		1/16	2		
become	1		1/16	2		
hit	1		1/16	2		
.	1		1/16	2		

Add-1 N-Grams (Unigrams)

The film got a great opening and the film went on to become a hit .

Word (Type)	Raw Count	Norm	Prob.	Add-1 Count	Add-1 Norm.	Add-1 Prob.
The	1	16	1/16	2	16 + 14*1 = 30	
film	2		1/8	3		
got	1		1/16	2		
a	2		1/8	3		
great	1		1/16	2		
opening	1		1/16	2		
and	1		1/16	2		
the	1		1/16	2		
went	1		1/16	2		
on	1		1/16	2		
to	1		1/16	2		
become	1		1/16	2		
hit	1		1/16	2		
.	1		1/16	2		

Add-1 N-Grams (Unigrams)

The film got a great opening and the film went on to become a hit .

Word (Type)	Raw Count	Norm	Prob.	Add-1 Count	Add-1 Norm.	Add-1 Prob.
The	1	16	1/16	2	16 + 14*1 = 30	=1/15
film	2		1/8	3		=1/10
got	1		1/16	2		=1/15
a	2		1/8	3		=1/10
great	1		1/16	2		=1/15
opening	1		1/16	2		=1/15
and	1		1/16	2		=1/15
the	1		1/16	2		=1/15
went	1		1/16	2		=1/15
on	1		1/16	2		=1/15
to	1		1/16	2		=1/15
become	1		1/16	2		=1/15
hit	1		1/16	2		=1/15
.	1		1/16	2		=1/15

An Extended Trigram Example

The film got a great opening and the film went on to become a hit .

Q: With OOV, EOS, and BOS, how many types (for normalization)?

Context: x y	Word (Type): z	Raw Count	Add-1 count	Norm.	Probability p(z | x y)
The film	The	0			
The film	film	0			
The film	got	1			
The film	went	0			
The film	OOV	0			
The film	EOS	0			
…					
a great	great	0			
a great	opening	1			
a great	and	0			
a great	the	0			
…					

An Extended Trigram Example

The film got a great opening and the film went on to become a hit .

Q: With OOV, EOS, and BOS, how many types (for normalization)?

A: 16
(why don't we count BOS?)

Context: x y	Word (Type): z	Raw Count	Add-1 count	Norm.	Probability p(z | x y)
The film	The	0			
The film	film	0			
The film	got	1			
The film	went	0			
The film	OOV	0			
The film	EOS	0			
		…			
a great	great	0			
a great	opening	1			
a great	and	0			
a great	the	0			
		…			

An Extended Trigram Example

The film got a great opening and the film went on to become a hit .

Q: With OOV, EOS, and BOS, how many types (for normalization)?

A: 16
(why don't we count BOS?)

Context: x y	Word (Type): z	Raw Count	Add-1 count	Norm.	Probability p(z | x y)
The film	The	0	1	17 (=1+16*1)	1/17
The film	film	0	1		1/17
The film	got	1	2		2/17
The film	went	0	1		1/17
…					…
The film	OOV	0	1		1/17
The film	EOS	0	1		1/17
…					
a great	great	0	1	17	1/17
a great	opening	1	2		2/17
a great	and	0	1		1/17
a great	the	0	1		1/17
…					

An Extended Trigram Example

The film got a great opening and the film went on to become a hit .

Context: x y	Word (Type): z	Raw Count	Add-1 count	Norm.	Probability p(z | x y)
The film	The	0	1	17 (=1+16*1)	1/17
The film	film	0	1		1/17
The film	got	1	2		2/17
The film	went	0	1		1/17
…					…
The film	OOV	0	1		1/17
The film	EOS	0	1		1/17
…					
a great	great	0	1	17	1/17
a great	opening	1	2		2/17
a great	and	0	1		1/17
a great	the	0	1		1/17
…					

Q: What is the perplexity for the sentence "The film , a hit !"

What are the tri-grams for “The film , a hit !”

Trigrams	MLE p(trigram)
<BOS> <BOS> The	1
<BOS> The film	1
The film ,	0
film , a	0
, a hit	0
a hit !	0
hit ! <EOS>	0

What are the tri-grams for "The film , a hit !"

Trigrams	MLE p(trigram)	UNK-ed trigrams
<BOS> <BOS> The	1	<BOS> <BOS> The
<BOS> The film	1	<BOS> The film
The film ,	0	The film <UNK>
film , a	0	film <UNK> a
, a hit	0	<UNK> a hit
a hit !	0	a hit <UNK>
hit ! <EOS>	0	hit <UNK> <EOS>

What are the tri-grams for “The film , a hit !”

Trigrams	MLE p(trigram)	UNK-ed trigrams	Smoothed p(trigram)
<BOS> <BOS> The	1	<BOS> <BOS> The	2/17
<BOS> The film	1	<BOS> The film	2/17
The film ,	0	The film <UNK>	1/17
film , a	0	film <UNK> a	1/16
, a hit	0	<UNK> a hit	1/16
a hit !	0	a hit <UNK>	1/17
hit ! <EOS>	0	hit <UNK> <EOS>	1/16

Setting Hyperparameters

Use a **development** corpus

Training Data | Dev Data | Test Data

Choose λs to maximize the probability of dev data:

- Fix the N-gram probabilities (on the training data)
- Then search for λs that give largest probability to held-out set:

Language Models & Smoothing

~~Maximum likelihood (MLE): simple counting~~

~~Other count-based models~~

- ~~Laplace smoothing, add-λ~~ ← Easy to implement
- ~~Interpolation models~~
- ~~Discounted backoff~~
- ~~Interpolated (modified) Kneser-Ney~~
- ~~Good-Turing~~
- ~~Witten-Bell~~

Advanced/ out of scope

Maxent n-gram models ← Featureful LMs

Neural n-gram models ← Feedforward LMs

Recurrent/autoregressive NNs ← Precursor to modern LMs

Maxent Models as Featureful n-gram Language Models

p(Colorless green ideas sleep furiously | Label) =
p(Colorless | Label, <BOS>) * … * p(<EOS> | Label , furiously)

Model each n-gram term with a maxent model

$$p(x_i \mid y, x_{i-N+1:i-1}) = \text{maxent}(y, x_{i-N+1:i-1}, x_i)$$

generatively trained:
learn to model *(class-specific) language*

Language Model with Maxent n-grams

$$p_n(🗎|y) = \prod_{i=1}^{M} \text{maxent}(\overset{\text{label}}{y}, \underbrace{x_{i-n+1:i-1}, x_i}_{\text{n-gram}})$$

$$= \prod_{i=1}^{M} \frac{\exp(\theta_{x_i}^T f(y, x_{i-n+1:i-1}))}{\sum_{x'} \exp(\theta_{x'}^T f(y, x_{i-n+1:i-1}))}$$

Iterate through all possible output vocab types x'---just like in count-based LMs

What Should These Features Do?

$p(x_i \mid y, x_{i-N+1:i-1}) = \text{maxent}(y, x_{i-N+1:i-1}, x_i)$, e.g.,

$$p(\text{sleep} \mid y, \text{green}, \text{ideas}) = \text{maxent}\big(y, x_{i-2,i-1} = (\text{green}, \text{ideas}), x_i = \text{sleep}\big)$$

$$\propto \exp({\theta_{x_i=\text{sleep}}}^T f(y, x_{i-2,i-1} = (\text{green}, \text{ideas})))$$

(in-class discussion)

N-gram Language Models

given some context…

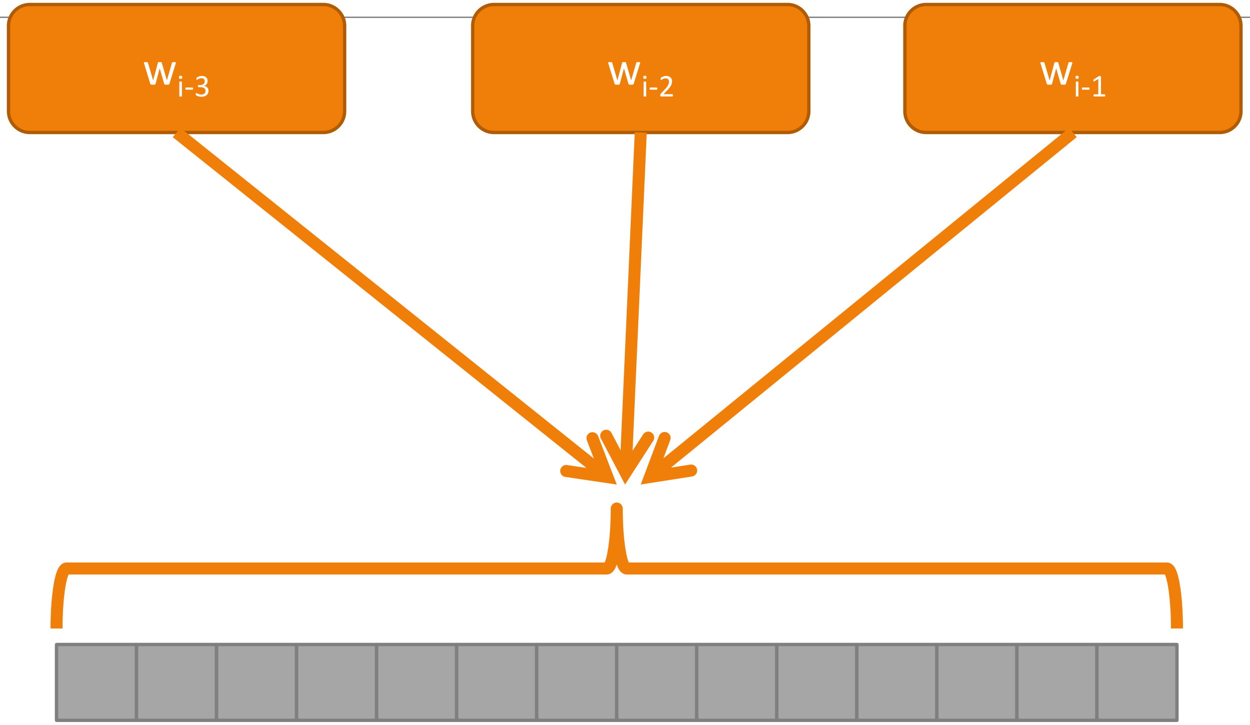
w_{i-3} w_{i-2} w_{i-1}

predict the next word

w_i

N-gram Language Models

given some context…



compute beliefs about what is likely…

$$p(w_i \mid w_{i-3}, w_{i-2}, w_{i-1}) \cong count(w_{i-3}, w_{i-2}, w_{i-1}, w_i)$$

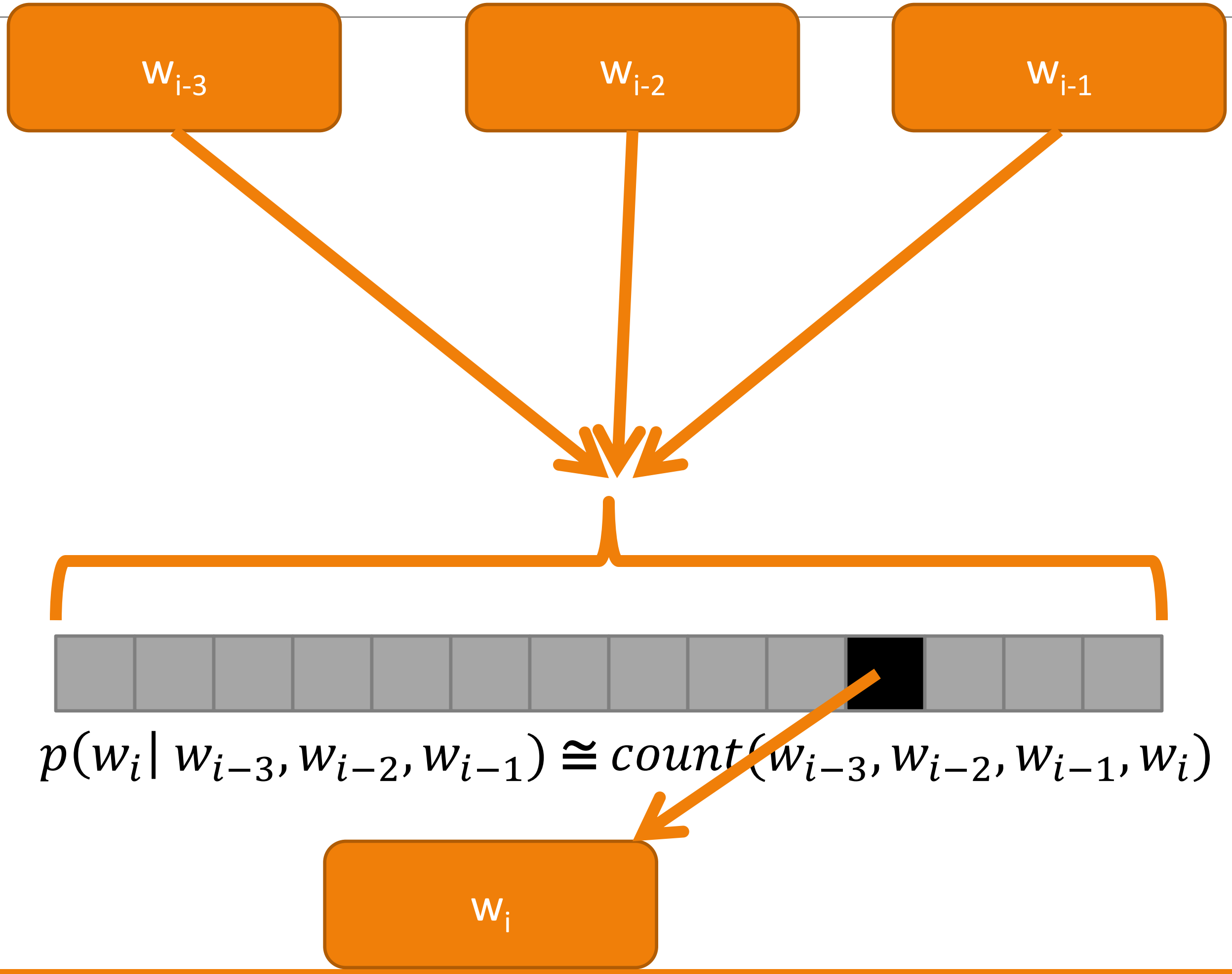
predict the next word

N-gram Language Models

given some context…

compute beliefs about what is likely…

predict the next word



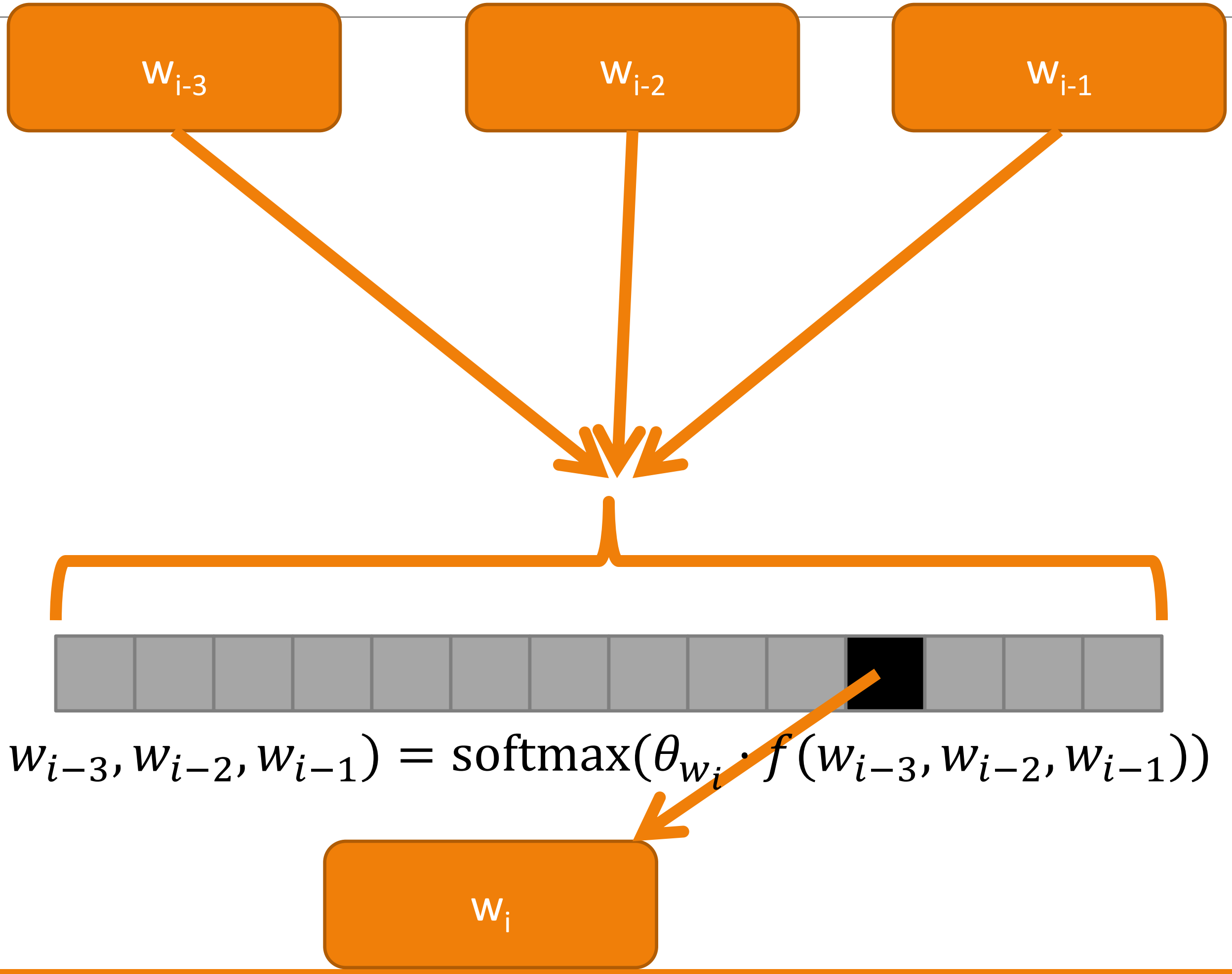
$$p(w_i \mid w_{i-3}, w_{i-2}, w_{i-1}) \cong count(w_{i-3}, w_{i-2}, w_{i-1}, w_i)$$

Maxent Language Models

given some context…

compute beliefs about what is likely…

predict the next word



$$p(w_i|\, w_{i-3}, w_{i-2}, w_{i-1}) = \text{softmax}(\theta_{w_i} \cdot f(w_{i-3}, w_{i-2}, w_{i-1}))$$

A Closer Look at Maxent $p($ Won’t you please donate? | Primary $)$

This is a *class-based* language model, but incorporate the label into the features

To learn $p($ Won’t you please donate? | Class $)$:

Define features f that make use of the specific label Class

Unlike count-based models, you don’t *need* “separate” models here