Introduction to Latent Sequences & Expectation Maximization

CMSC 473/673

UMBC

October 2nd, 2017

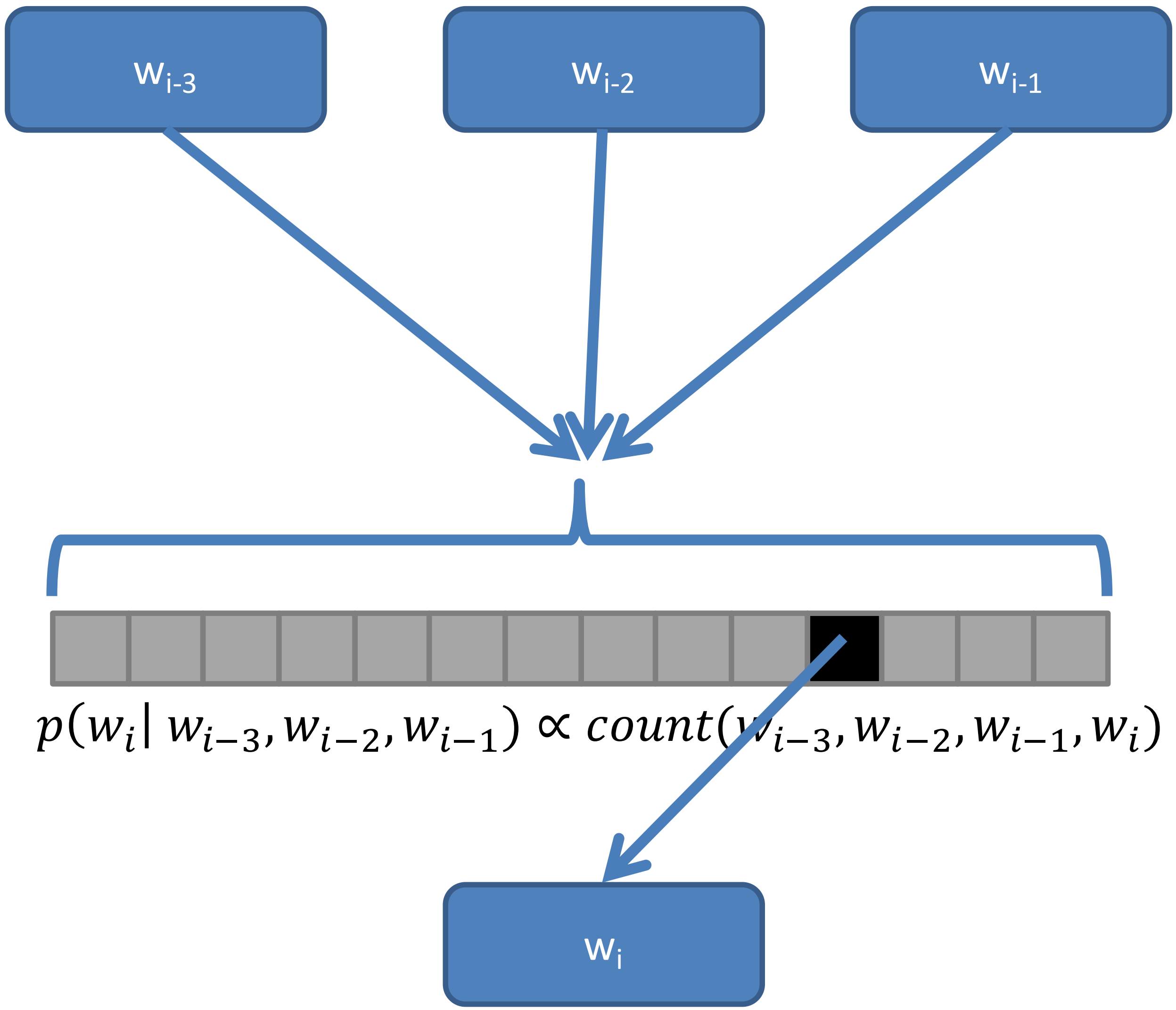
Recap from last time (and the first unit)…

N-gram Language Models

given some context…

compute beliefs about what is likely…

predict the next word

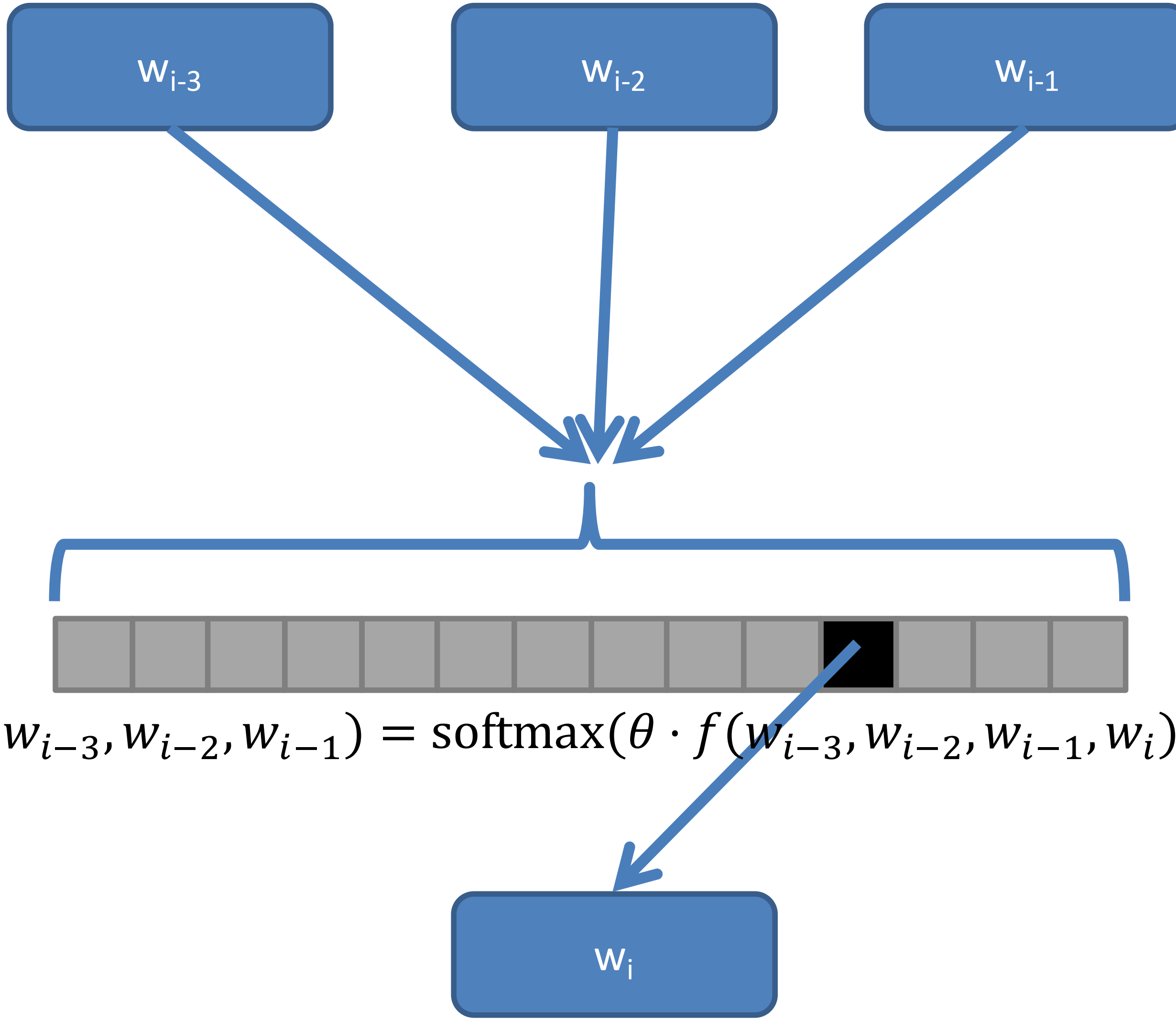


Maxent Language Models

given some context…

compute beliefs about what is likely…

predict the next word



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

Neural Language Models

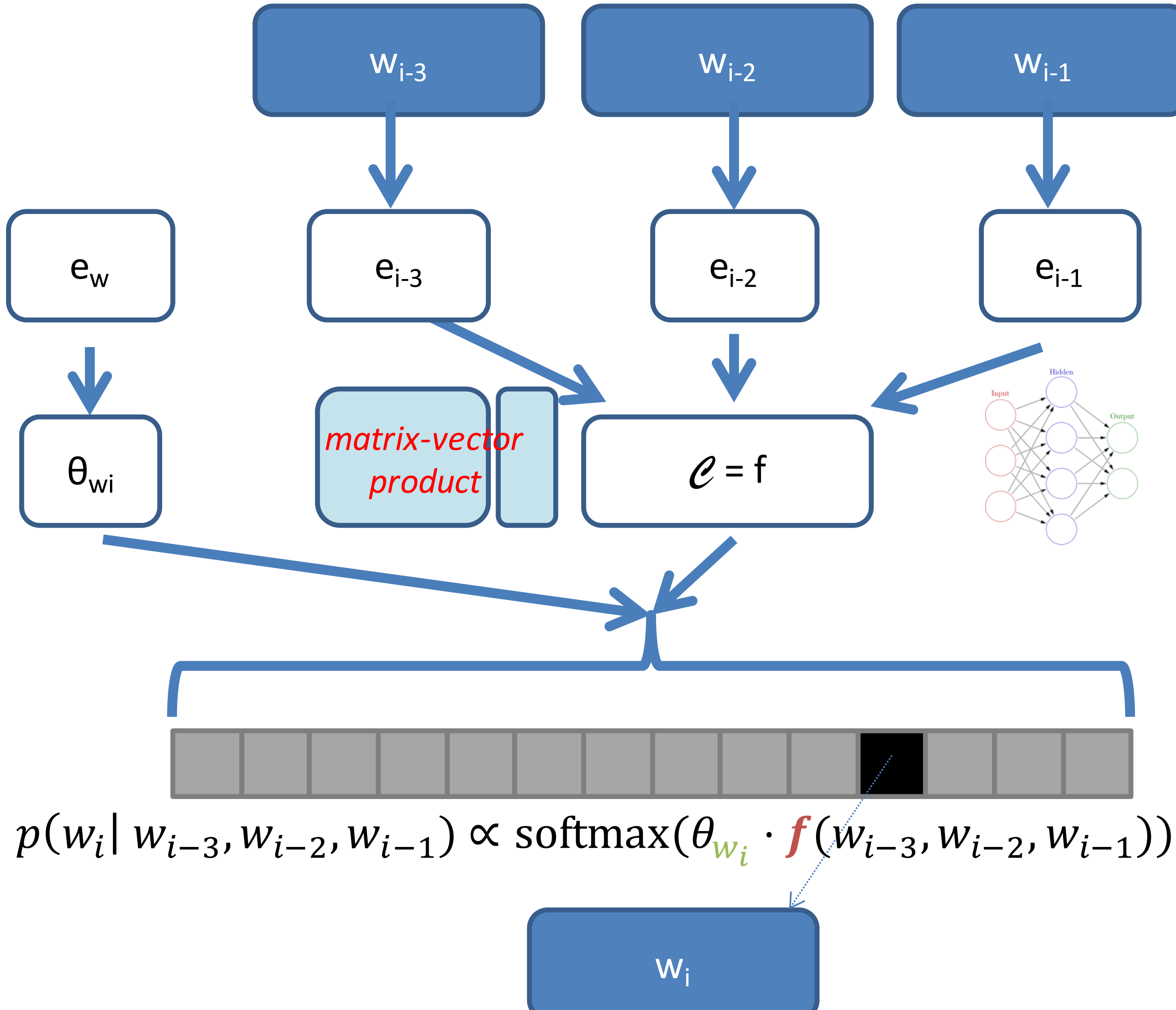
given some context…

create/use "distributed representations"…

combine these representations…

compute beliefs about what is likely…

predict the next word



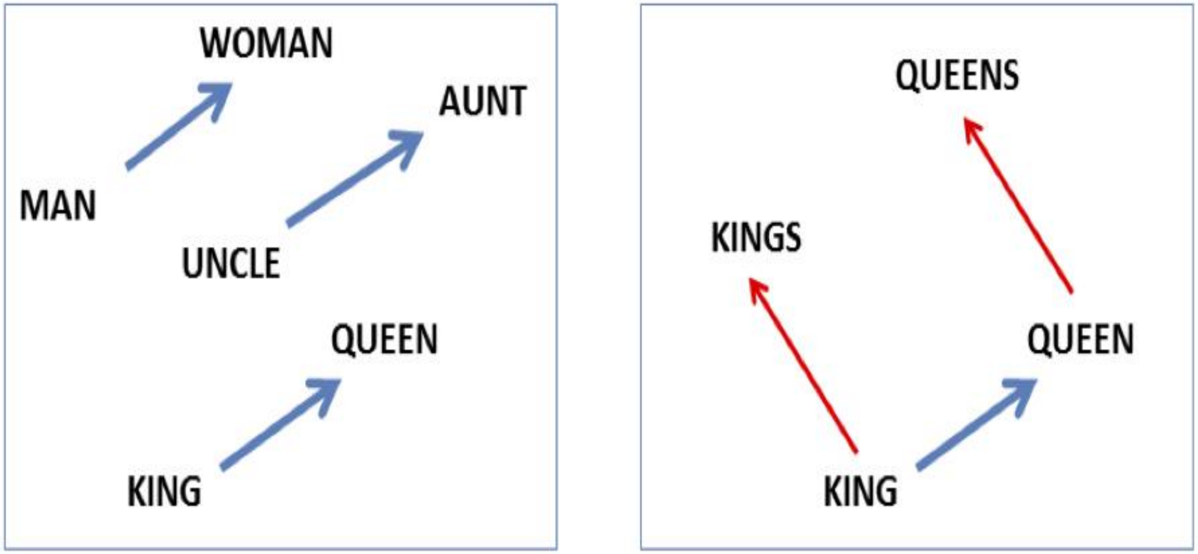
$$p(w_i | w_{i-3}, w_{i-2}, w_{i-1}) \propto \text{softmax}(\theta_{w_i} \cdot f(w_{i-3}, w_{i-2}, w_{i-1}))$$

(Some) Properties of Embeddings

Capture "like" (similar) words

target:	Redmond	Havel	ninjutsu	graffiti	capitulate
	Redmond Wash.	Vaclav Havel	ninja	spray paint	capitulation
	Redmond Washington	president Vaclav Havel	martial arts	grafitti	capitulated
	Microsoft	Velvet Revolution	swordsmanship	taggers	capitulating

Capture relationships



Mikolov et al. (2013)

vector(*'king'*) –
vector(*'man'*) +
vector(*'woman'*) ≈
vector('queen')

vector(*'Paris'*) -
vector(*'France'*) +
vector(*'Italy'*) ≈
vector('Rome')

Four kinds of vector models

Sparse vector representations

1. Mutual-information weighted word co-occurrence matrices

Dense vector representations:

2. Singular value decomposition/Latent Semantic Analysis
3. Neural-network-inspired models (skip-grams, CBOW)
4. Brown clusters

Learn more in:

- Your project
- Paper (673)
- Other classes (478/678)

Shared Intuition

Model the meaning of a word by “embedding” in a vector space

The meaning of a word is a vector of numbers

Contrast: word meaning is represented in many computational linguistic applications by a vocabulary index (“word number 545”) or the string itself

Intrinsic Evaluation: Cosine Similarity

Divide the dot product by the length of the two vectors

$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|}$$

This is the cosine of the angle between them

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos\theta$$

$$\frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \cos\theta$$

Are the vectors parallel?

-1: vectors point in opposite directions

+1: vectors point in same directions

0: vectors are orthogonal

Course Recap So Far

Basics of Probability

Requirements to be a distribution ("proportional to", $\propto$)

Definitions of conditional probability, joint probability, and independence

Bayes rule, (probability) chain rule

Course Recap So Far

Basics of Probability

Requirements to be a distribution (“proportional to”, $\propto$)

Definitions of conditional probability, joint probability, and independence

Bayes rule, (probability) chain rule

Basics of language modeling

Goal: model (be able to predict) and give a score to *language* (whole sequences of characters or words)

Simple count-based model

Smoothing (and why we need it): Laplace (add-λ), interpolation, backoff

Evaluation: perplexity

Course Recap So Far

Basics of Probability

- Requirements to be a distribution ("proportional to", ∝)
- Definitions of conditional probability, joint probability, and independence
- Bayes rule, (probability) chain rule

Basics of language modeling

- Goal: model (be able to predict) and give a score to *language* (whole sequences of characters or words)
- Simple count-based model
- Smoothing (and why we need it): Laplace (add-λ), interpolation, backoff
- Evaluation: perplexity

Tasks and Classification (use Bayes rule!)

- Posterior decoding vs. noisy channel model
- Evaluations: accuracy, precision, recall, and F_β (F_1) scores
- Naïve Bayes (given the label, generate/explain each feature independently) and connection to language modeling

Course Recap So Far

Basics of Probability

- Requirements to be a distribution (“proportional to”, $\propto$)
- Definitions of conditional probability, joint probability, and independence
- Bayes rule, (probability) chain rule

Basics of language modeling

- Goal: model (be able to predict) and give a score to *language* (whole sequences of characters or words)
- Simple count-based model
- Smoothing (and why we need it): Laplace (add-λ), interpolation, backoff
- Evaluation: perplexity

Tasks and Classification (use Bayes rule!)

- Posterior decoding vs. noisy channel model
- Evaluations: accuracy, precision, recall, and F_β (F_1) scores
- Naïve Bayes (given the label, generate/explain each feature independently) and connection to language modeling

Maximum Entropy Models

- Meanings of feature functions and weights
- Use for language modeling or conditional classification (“posterior in one go”)
- How to learn the weights: gradient descent

Course Recap So Far

Basics of Probability

- Requirements to be a distribution ("proportional to", $\propto$)
- Definitions of conditional probability, joint probability, and independence
- Bayes rule, (probability) chain rule

Basics of language modeling

- Goal: model (be able to predict) and give a score to *language* (whole sequences of characters or words)
- Simple count-based model
- Smoothing (and why we need it): Laplace (add-λ), interpolation, backoff
- Evaluation: perplexity

Tasks and Classification (use Bayes rule!)

- Posterior decoding vs. noisy channel model
- Evaluations: accuracy, precision, recall, and F_β (F_1) scores
- Naïve Bayes (given the label, generate/explain each feature independently) and connection to language modeling

Maximum Entropy Models

- Meanings of feature functions and weights
- Use for language modeling or conditional classification ("posterior in one go")
- How to learn the weights: gradient descent

Distributed Representations & Neural Language Models

- What embeddings are and what their motivation is
- A common way to evaluate: cosine similarity

Course Recap So Far

Basics of Probability

- Requirements to be a distribution ("proportional to", $\propto$)
- Definitions of conditional probability, joint probability, and independence
- Bayes rule, (probability) chain rule

Basics of language modeling

- Goal: model (be able to predict) and give a score to *language* (whole sequences of characters or words)
- Simple count-based model
- Smoothing (and why we need it): Laplace (add-λ), interpolation, backoff
- Evaluation: perplexity

Tasks and Classification (use Bayes rule!)

- Posterior decoding vs. noisy channel model
- Evaluations: accuracy, precision, recall, and F_β (F_1) scores
- Naïve Bayes (given the label, generate/explain each feature independently) and connection to language modeling

Maximum Entropy Models

- Meanings of feature functions and weights
- Use for language modeling or conditional classification ("posterior in one go")
- How to learn the weights: gradient descent

Distributed Representations & Neural Language Models

- What embeddings are and what their motivation is
- A common way to evaluate: cosine similarity

LATENT SEQUENCES AND LATENT VARIABLE MODELS

Is Language Modeling "Latent?"

p(Colorless green ideas sleep furiously) =
p(Colorless) *
p(green | Colorless) *
p(ideas | Colorless green) *
p(sleep | green ideas) *
p(furiously | ideas sleep)

Is Language Modeling "Latent?"
Most* of What We've Discussed: Not Really

p(Colorless green ideas sleep furiously) =

p(Colorless) *

p(green | Colorless) *

p(ideas | Colorless green) *

p(sleep | green ideas) *

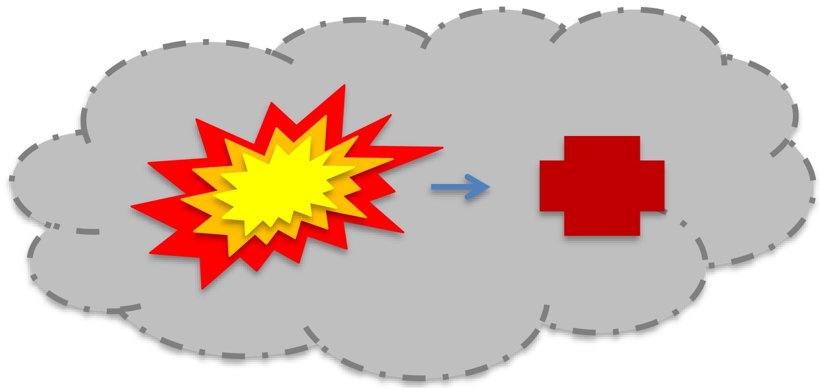
p(furiously | ideas sleep)

these *values* are unknown

but the generation process (explanation) is transparent

*Neural language modeling as an exception

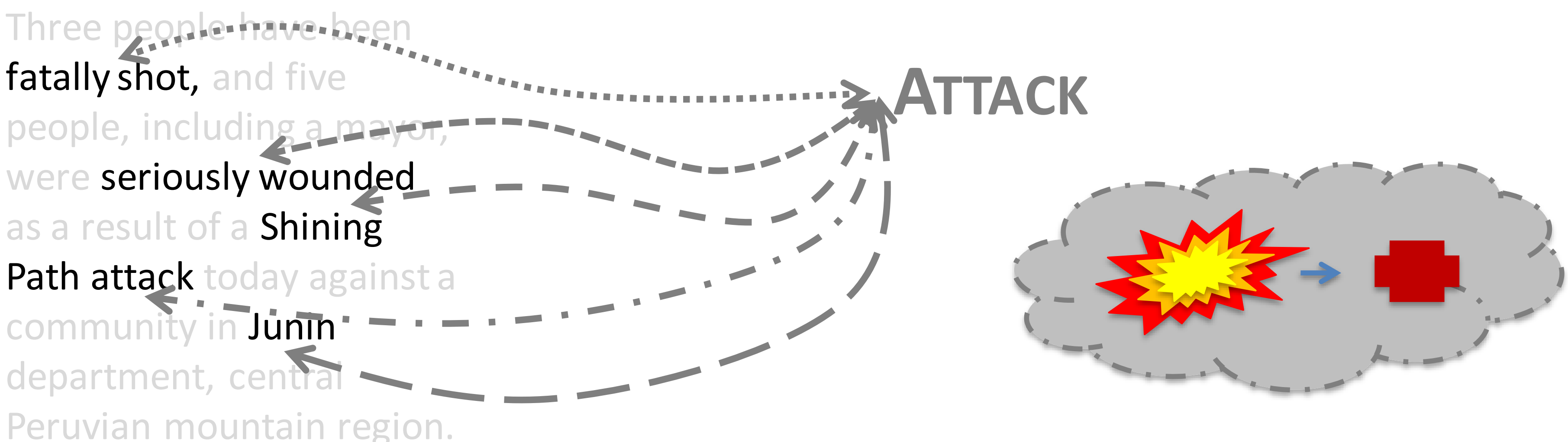
Is Document Classification “Latent?”



Three people have been fatally shot, and five people, including a mayor, were seriously wounded as a result of a Shining Path attack today against a community in Junin department, central Peruvian mountain region.

ATTACK

Is Document Classification "Latent?" As We've Discussed



$$\operatorname{argmax}_X \prod_i p(Y_i|X) * p(X)$$

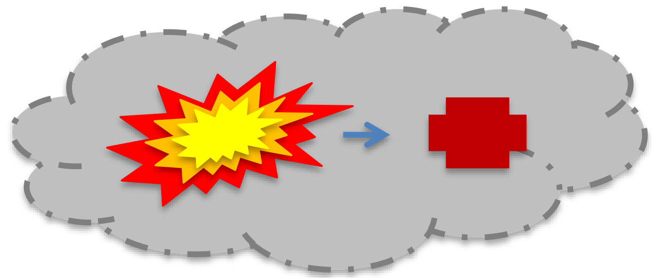
$$\operatorname{argmax}_X \frac{\exp\left(\theta \cdot f(x,y)\right)}{Z(x)} * p(x)$$

$$\operatorname{argmax}_X \exp(\theta \cdot f(x,y))$$

Is Document Classification "Latent?" As We've Discussed: Not Really

Three people have been **fatally shot,** and five people, including a mayor, were **seriously wounded** as a result of a **Shining Path attack** today against a community in **Junin** department, central Peruvian mountain region.

ATTACK



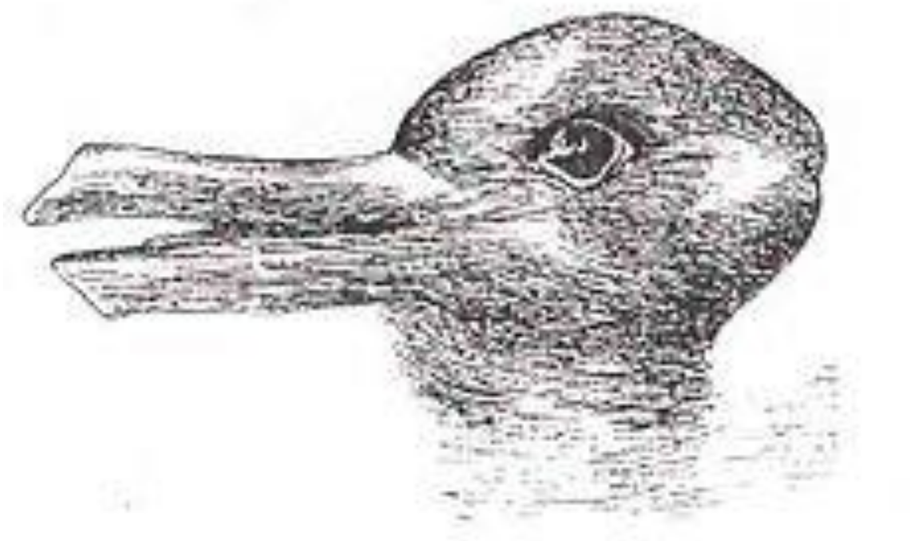
these *values* are unknown

$$\text{argmax}_X \prod_i p(Y_i|X) * p(X)$$

but the generation process (explanation) is transparent

$$\text{argmax}_X \frac{\exp(\theta \cdot f(x,y))}{Z(x)} * p(x)$$

$$\text{argmax}_X \exp(\theta \cdot f(x,y))$$



Ambiguity → Part of Speech Tagging

British Left Waffles on Falkland Islands

British Left Waffles on Falkland Islands

Adjective *Noun* *Verb*

British Left Waffles on Falkland Islands

Noun *Verb* *Noun*


Adapted from Jason Eisner, Noah Smith

observed text

orthography

morphology

lexemes

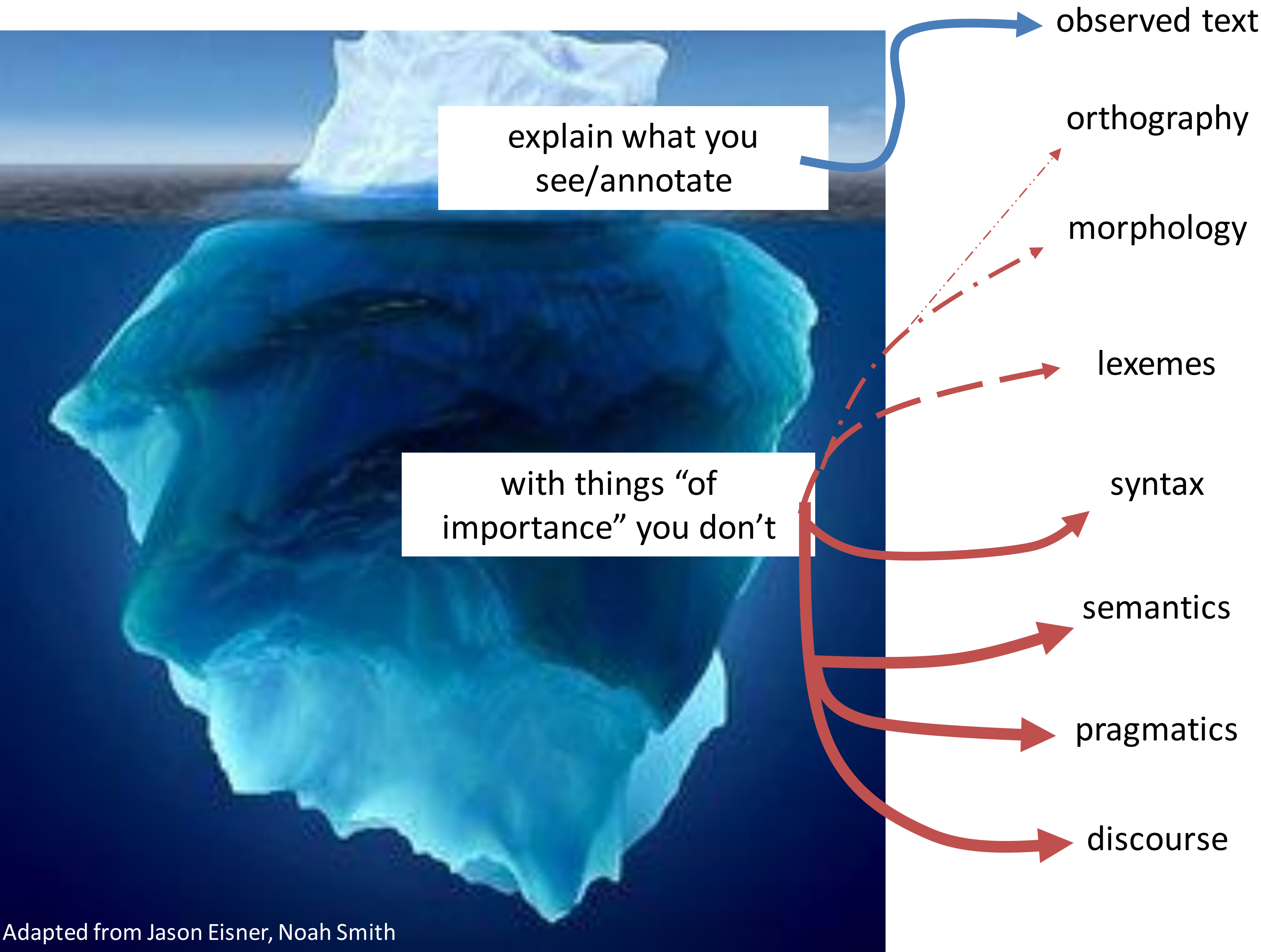
syntax

semantics

pragmatics

discourse

Latent Modeling



Latent Sequence Models: Part of Speech

p(British Left Waffles on Falkland Islands)

Latent Sequence Models: Part of Speech

(i):	*Adjective*	*Noun*	*Verb*	*Prep*	*Noun*	*Noun*
(ii):	*Noun*	*Verb*	*Noun*	*Prep*	*Noun*	*Noun*

p(British Left Waffles on Falkland Islands)

Latent Sequence Models: Part of Speech

(i):	*Adjective*	*Noun*	*Verb*	*Prep*	*Noun*	*Noun*
(ii):	*Noun*	*Verb*	*Noun*	*Prep*	*Noun*	*Noun*

p(British Left Waffles on Falkland Islands)

1. Explain this sentence as a sequence of (likely?) latent (unseen) tags (labels)

Latent Sequence Models: Part of Speech

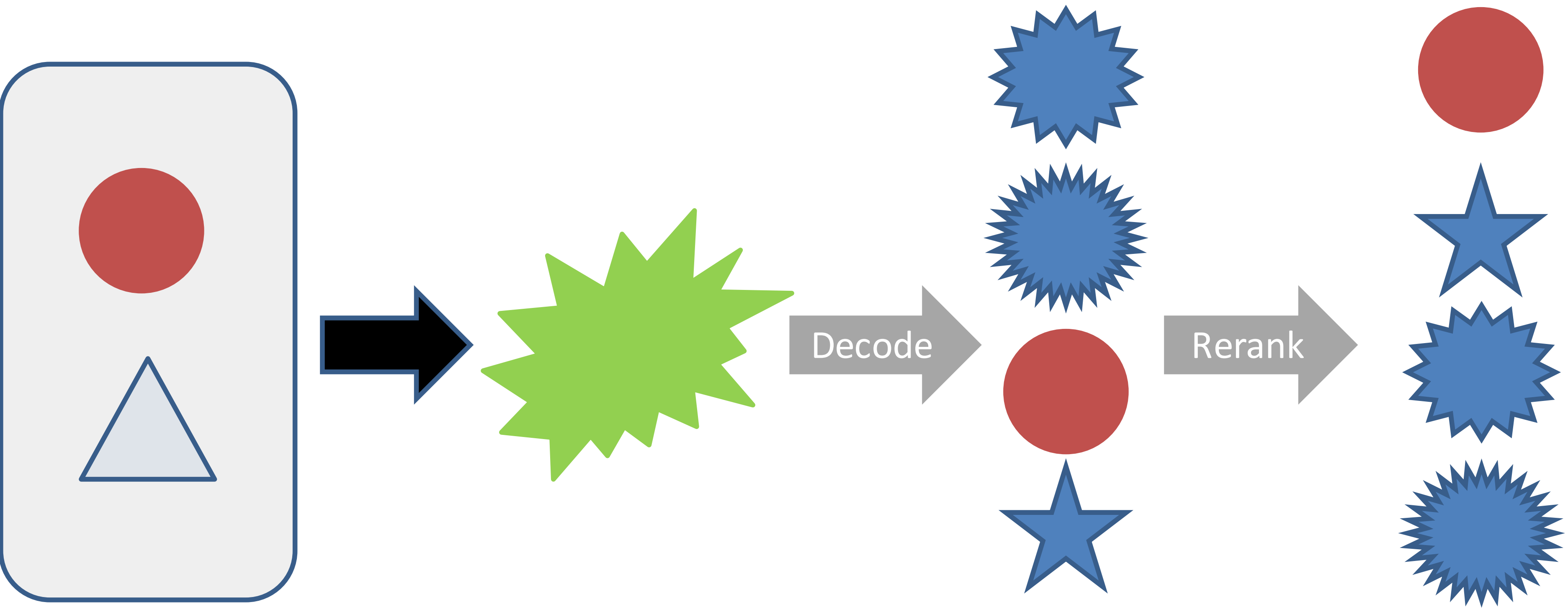
(i):	*Adjective*	*Noun*	*Verb*	*Prep*	*Noun*	*Noun*
(ii):	*Noun*	*Verb*	*Noun*	*Prep*	*Noun*	*Noun*

p(British Left Waffles on Falkland Islands)

1. Explain this sentence as a sequence of (likely?) latent (unseen) tags (labels)

2. Produce a tag sequence for this sentence

Noisy Channel Model



possible (clean) output

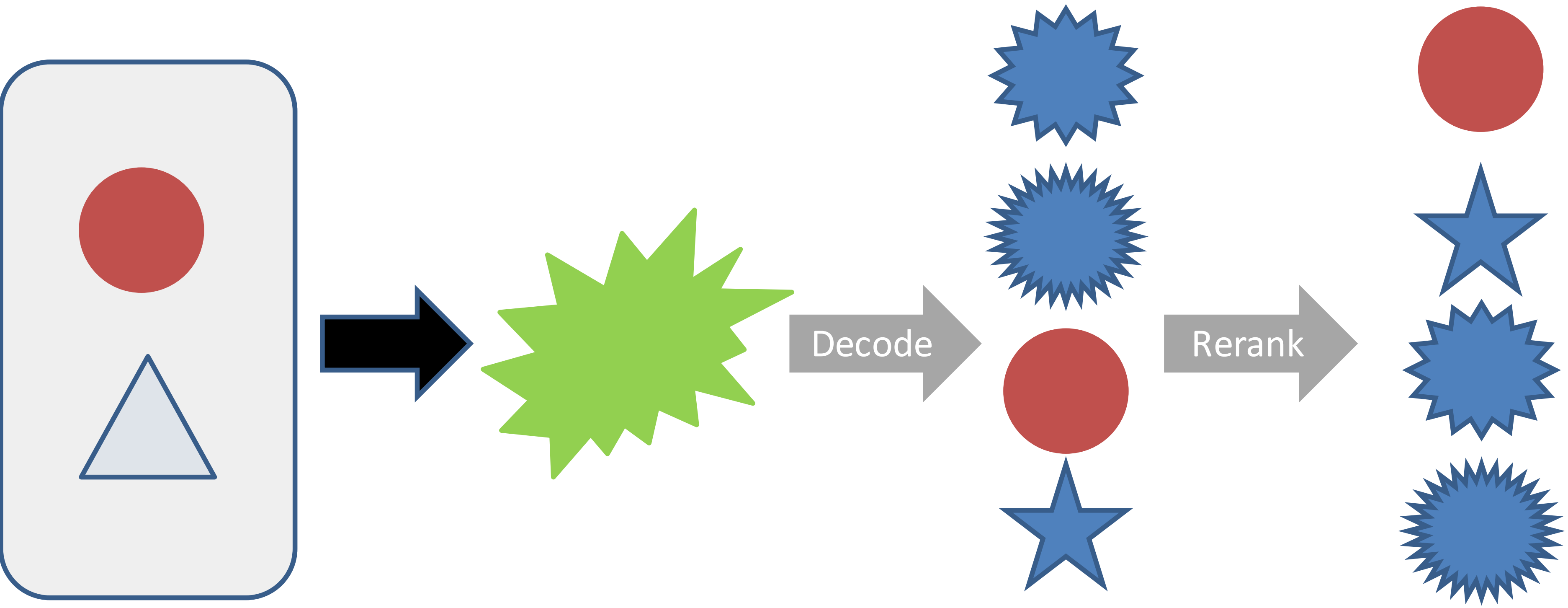
translation/ decode model

(clean) language model

$$p(X \mid Y) \propto p(Y \mid X) * p(X)$$

observed (noisy) text

Latent Sequence Model: Machine Translation



possible (clean) output

translation/ decode model

(clean) language model

$$p(X \mid Y) \propto p(Y \mid X) * p(X)$$

observed (noisy) text

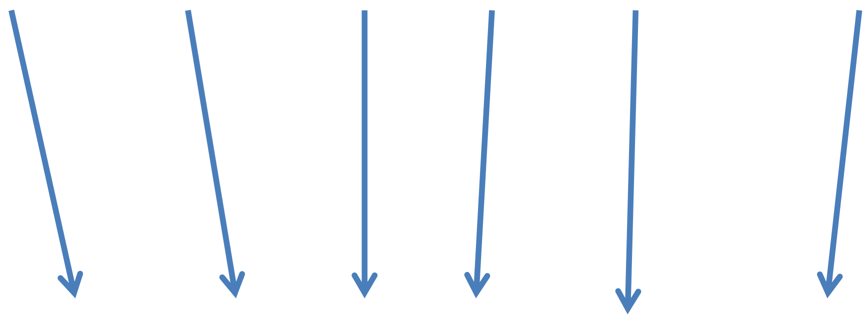
Latent Sequence Model: Machine Translation

Le chat est sur la chaise.

Eddie Izzard, “Dress to Kill” (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

Latent Sequence Model: Machine Translation

Le chat est sur la chaise.



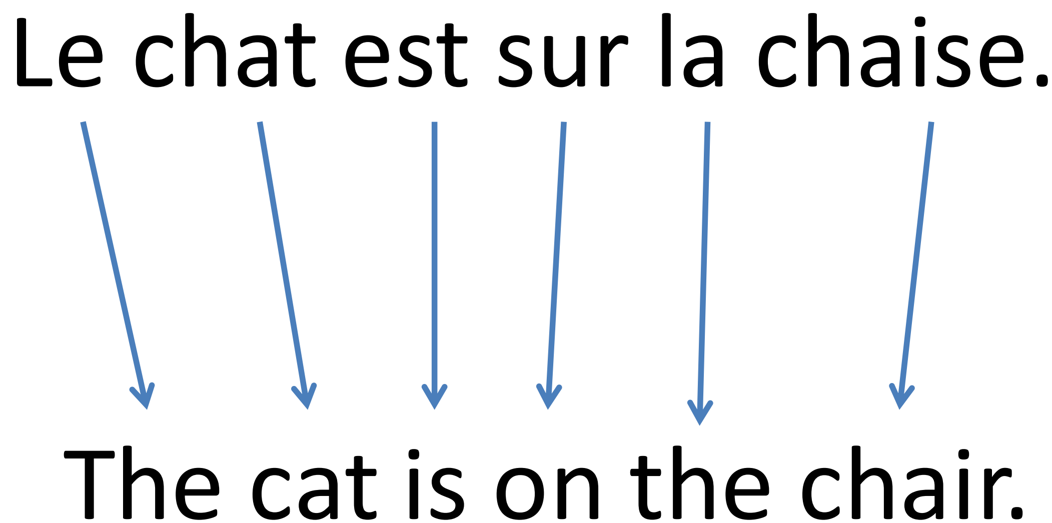
The cat is on the chair.

Eddie Izzard, “Dress to Kill” (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

Latent Sequence Model: Machine Translation

How do you know what words translate as?

Learn the translations!



Eddie Izzard, “Dress to Kill” (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

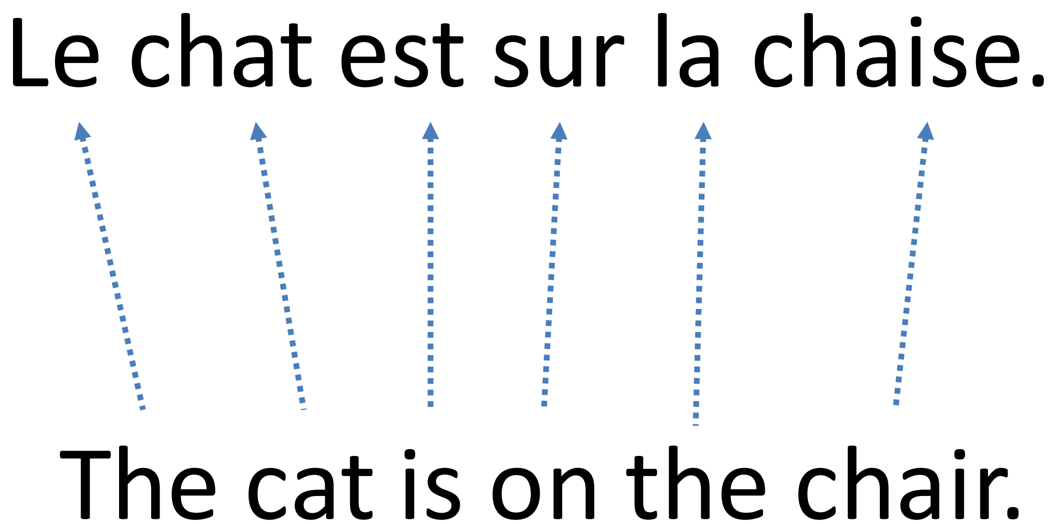
Latent Sequence Model: Machine Translation

How do you know what words translate as?

Learn the translations!

How?

Learn a "reverse" latent alignment model p(French words, alignments | English words)



Eddie Izzard, "Dress to Kill" (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

Latent Sequence Model: Machine Translation

How do you know what words translate as?

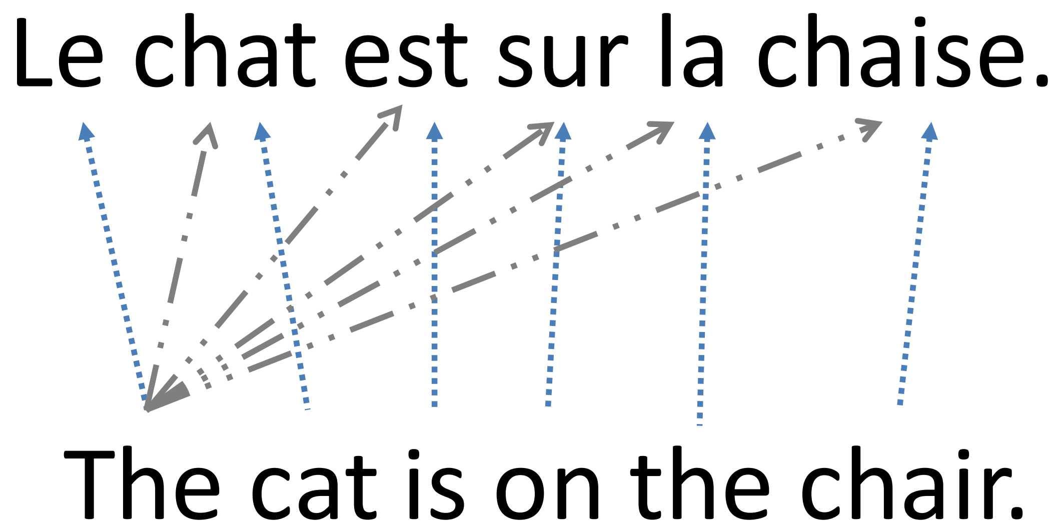
Learn the translations!

How?

Learn a "reverse" latent alignment model p(French words, alignments | English words)

Alignment?

Words can have different meaning/senses



Eddie Izzard, "Dress to Kill" (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

Latent Sequence Model: Machine Translation

How do you know what words translate as?

Learn the translations!

How?

Learn a “reverse” latent alignment model

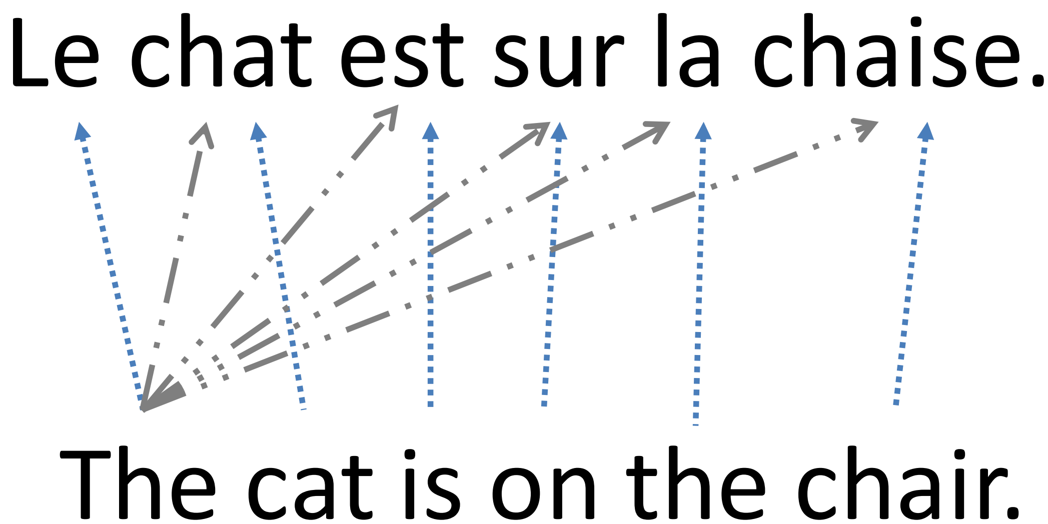
p(French words, alignments | English words)

Alignment?

Words can have different meaning/senses

Why Reverse?

$$p(\text{English} \mid \text{French}) \propto p(\text{French} \mid \text{English}) * p(\text{English})$$



Eddie Izzard, “Dress to Kill” (MPAA: R)
https://www.youtube.com/watch?v=x1sQkEfAdfY

How to Learn With Latent Variables (Sequences)

Expectation Maximization

Example: Unigram Language Modeling

$$p(w_1, w_2, \dots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

Example: Unigram Language Modeling

$$p(w_1, w_2, \dots, w_N) = p(w_1)p(w_2) \cdots p(w_N) = \prod_i p(w_i)$$

maximize (log-)likelihood to learn the probability parameters

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \ldots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

add complexity to better explain what we see

$$p(z_1, w_1, z_2, w_2, \ldots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$

$$= \prod_i p(w_i|z_i)\, p(z_i)$$

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \ldots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

add complexity to better explain what we see

$$p(z_1, w_1, z_2, w_2, \ldots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$

$$= \prod_i p(w_i|z_i)\, p(z_i)$$

examples of latent classes z:

- part of speech tag
- topic (“sports” vs. “politics”)

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \dots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

↓ *add complexity to better explain what we see*

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$
$$= \prod_i p(w_i|z_i)\, p(z_i)$$

goal: maximize (log-)likelihood

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \ldots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

add complexity to better explain what we see

$$p(z_1, w_1, z_2, w_2, \ldots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$

$$= \prod_i p(w_i|z_i)\,p(z_i)$$

goal: maximize (log-)likelihood

we don't actually observe these z values

we just see the words w

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \dots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

⬇ *add complexity to better explain what we see*

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$

$$= \prod_i p(w_i|z_i)\, p(z_i)$$

goal: maximize (log-)likelihood

we don't actually observe these z values
we just see the words w

if we *did* observe *z*, estimating the probability parameters would be easy…
but we don't! :(

Example: Unigram Language Modeling with Hidden Class

$$p(w_1, w_2, \dots, w_N) = p(w_1)p(w_2)\cdots p(w_N) = \prod_i p(w_i)$$

add complexity to better explain what we see

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$
$$= \prod_i p(w_i|z_i)\, p(z_i)$$

goal: maximize (log-)likelihood

we don't actually observe these z values
we just see the words w

if we *did* observe *z*, estimating the probability parameters would be easy… but we don't! :(

if we *knew* the probability parameters then we could estimate *z* and evaluate likelihood… but we don't! :(

Example: Unigram Language Modeling with Hidden Class

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1) p(w_1 | z_1) \cdots p(z_N) p(w_N | z_N)$$
$$= \prod_i p(w_i | z_i)\, p(z_i)$$

we don't actually observe these z values

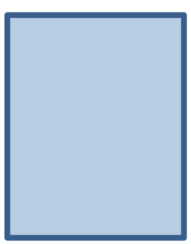
goal: maximize ***marginalized*** (log-)likelihood

Example: Unigram Language Modeling with Hidden Class

$$p(z_1, w_1, z_2, w_2, \ldots, z_N, w_N) = p(z_1)p(w_1|z_1) \cdots p(z_N)p(w_N|z_N)$$
$$= \prod_i p(w_i|z_i)\, p(z_i)$$

we don't actually observe these z values

goal: maximize ***marginalized*** (log-)likelihood



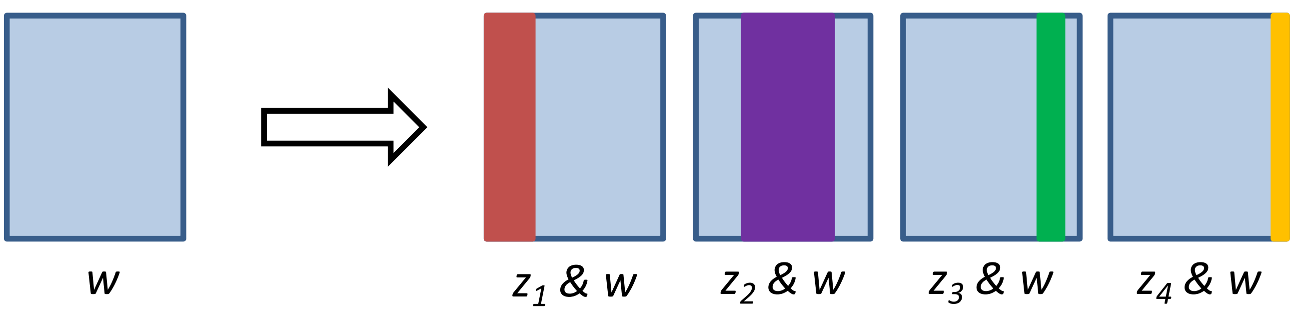
w

Example: Unigram Language Modeling with Hidden Class

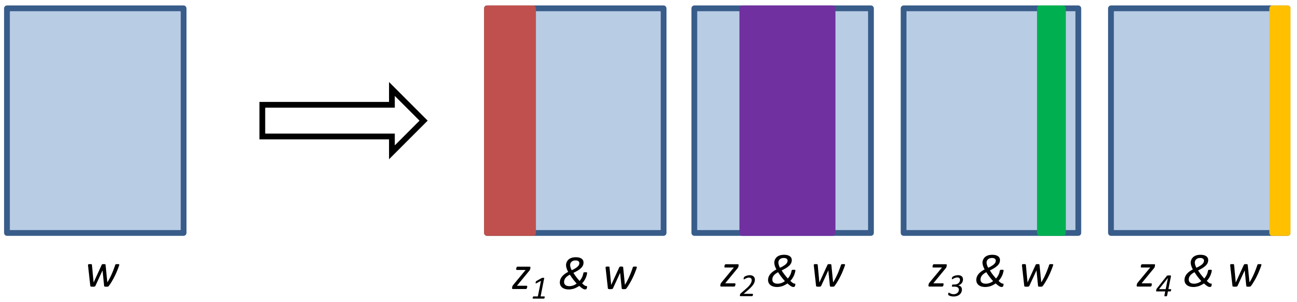
$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1)\cdots p(z_N)p(w_N|z_N)$$
$$= \prod_i p(w_i|z_i)\, p(z_i)$$

we don't actually observe these z values

goal: maximize ***marginalized*** (log-)likelihood

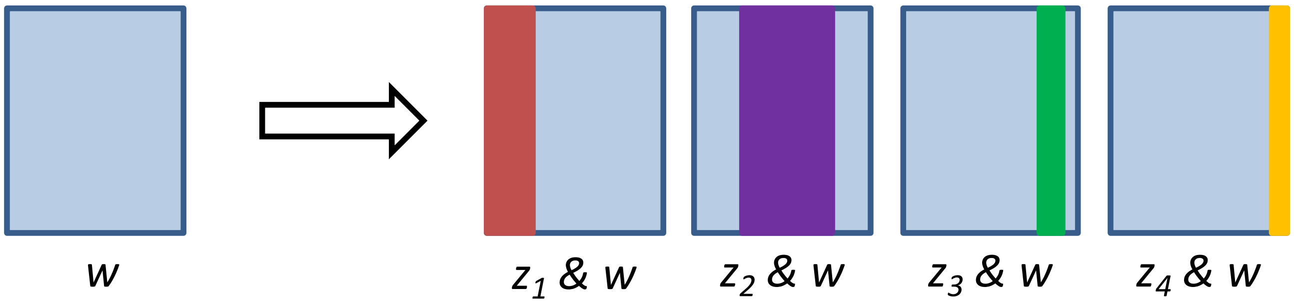


Marginal(ized) Probability



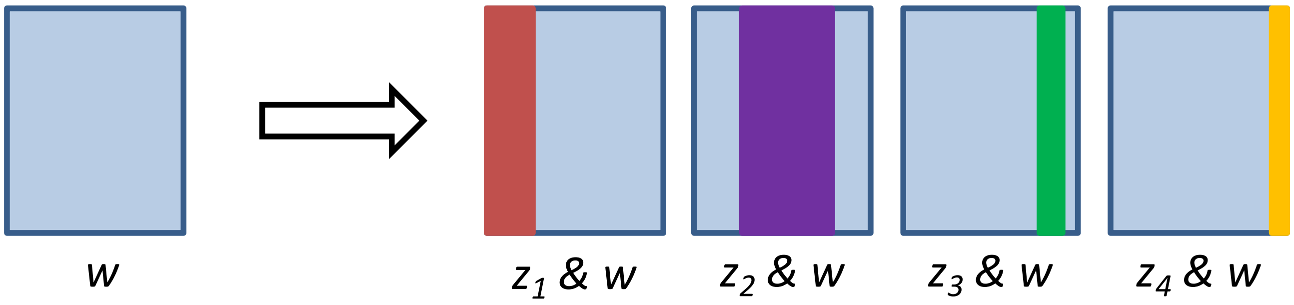
$$p(w) = p(z_1, w) + p(z_2, w) + p(z_3, w) + p(z_4, w)$$

Marginal(ized) Probability



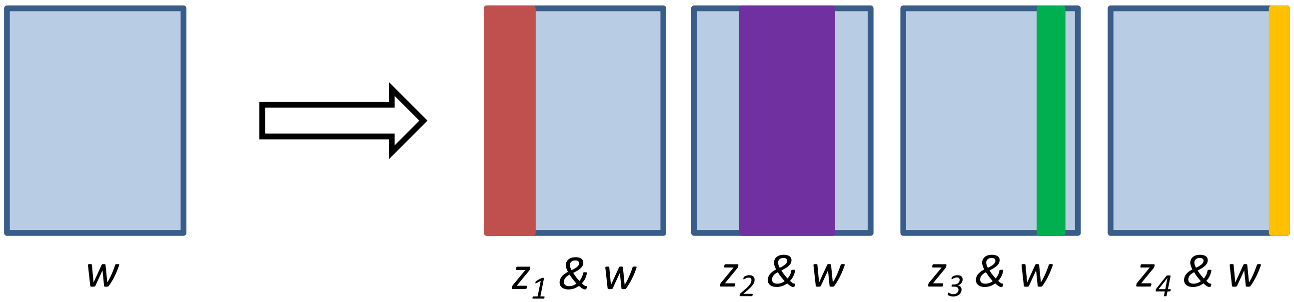
$$p(w) = p(z_1, w) + p(z_2, w) + p(z_3, w) + p(z_4, w) = \sum_{z=1}^{4} p(z_i, w)$$

Marginal(ized) Probability



$$p(w) = \sum_{z} p(z, w)$$

Marginal(ized) Probability



$$p(w) = \sum_z p(z, w)$$

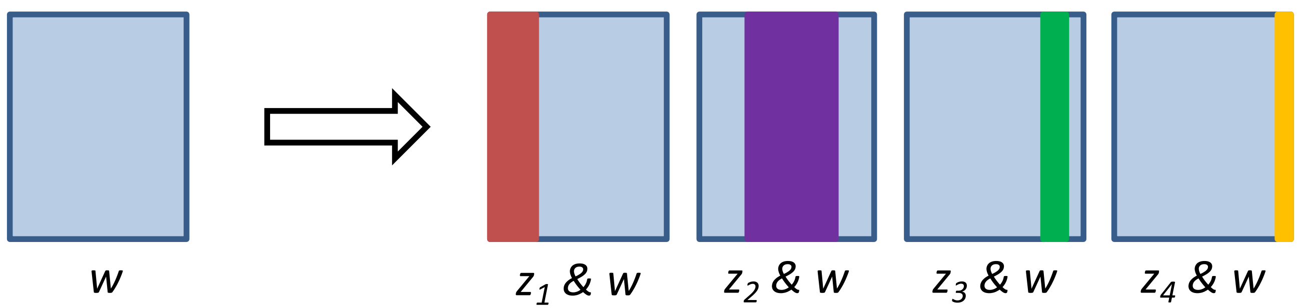
$$= \sum_z p(z) p(w \mid z)$$

Example: Unigram Language Modeling with Hidden Class

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1) \cdots p(z_N)p(w_N|z_N)$$
$$= \prod_i p(w_i|z_i)\, p(z_i)$$

we don't actually observe these z values

goal: maximize ***marginalized*** (log-)likelihood

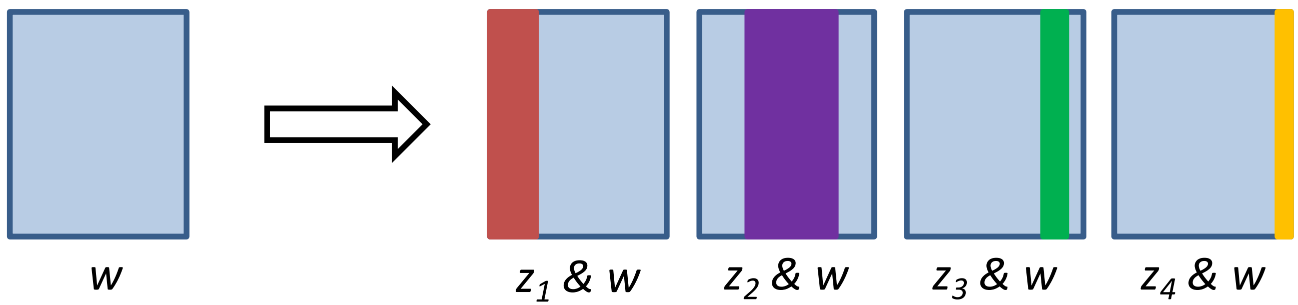


$$p(w_1, w_2, \dots, w_N) = \left(\sum_{z_1} p(z_1, w)\right)\left(\sum_{z_2} p(z_2, w)\right) \cdots \left(\sum_{z_N} p(z_N, w)\right)$$

Example: Unigram Language Modeling with Hidden Class

$$p(z_1, w_1, z_2, w_2, \dots, z_N, w_N) = p(z_1)p(w_1|z_1) \cdots p(z_N)p(w_N|z_N)$$

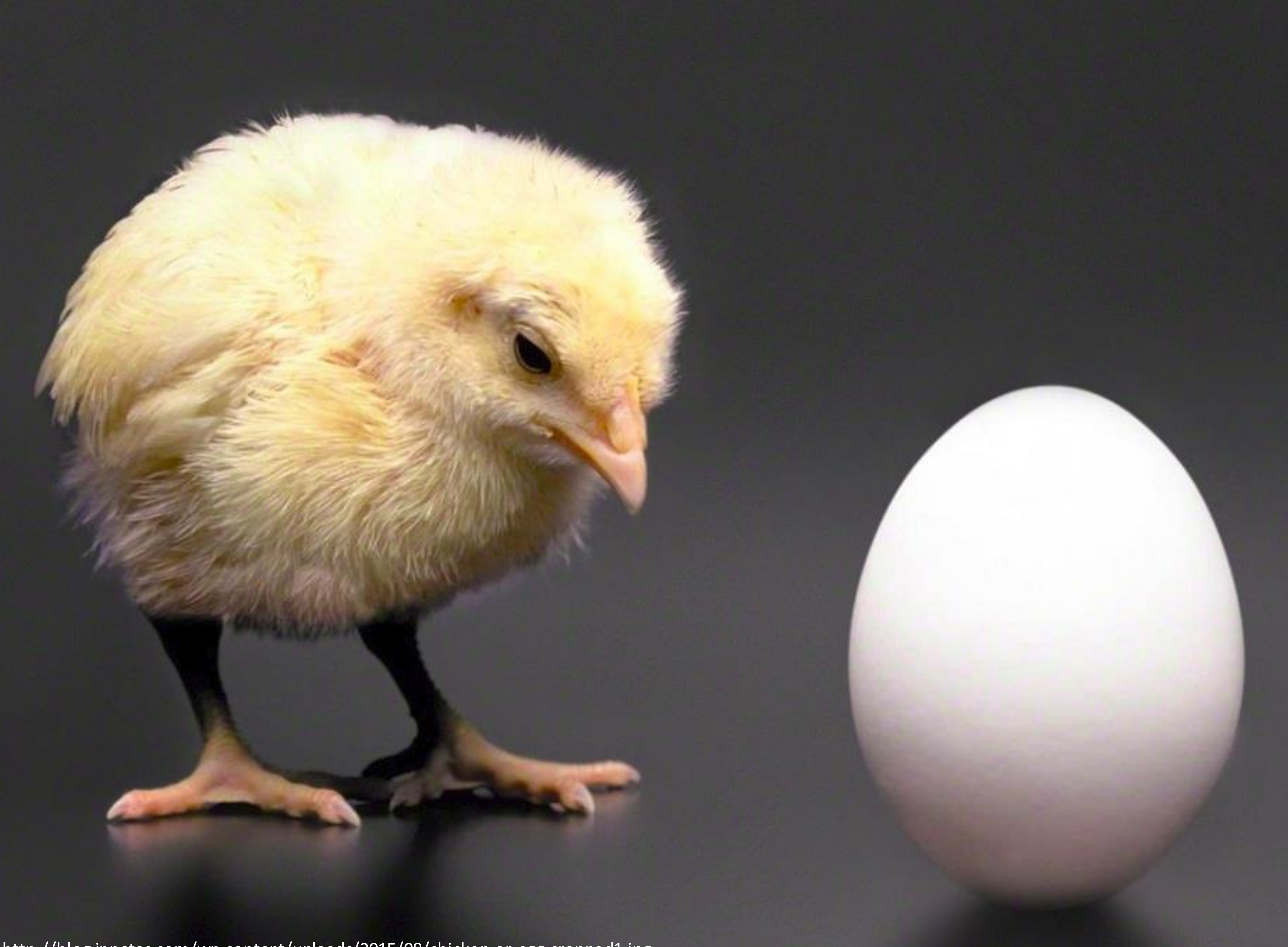
goal: maximize ***marginalized*** (log-)likelihood



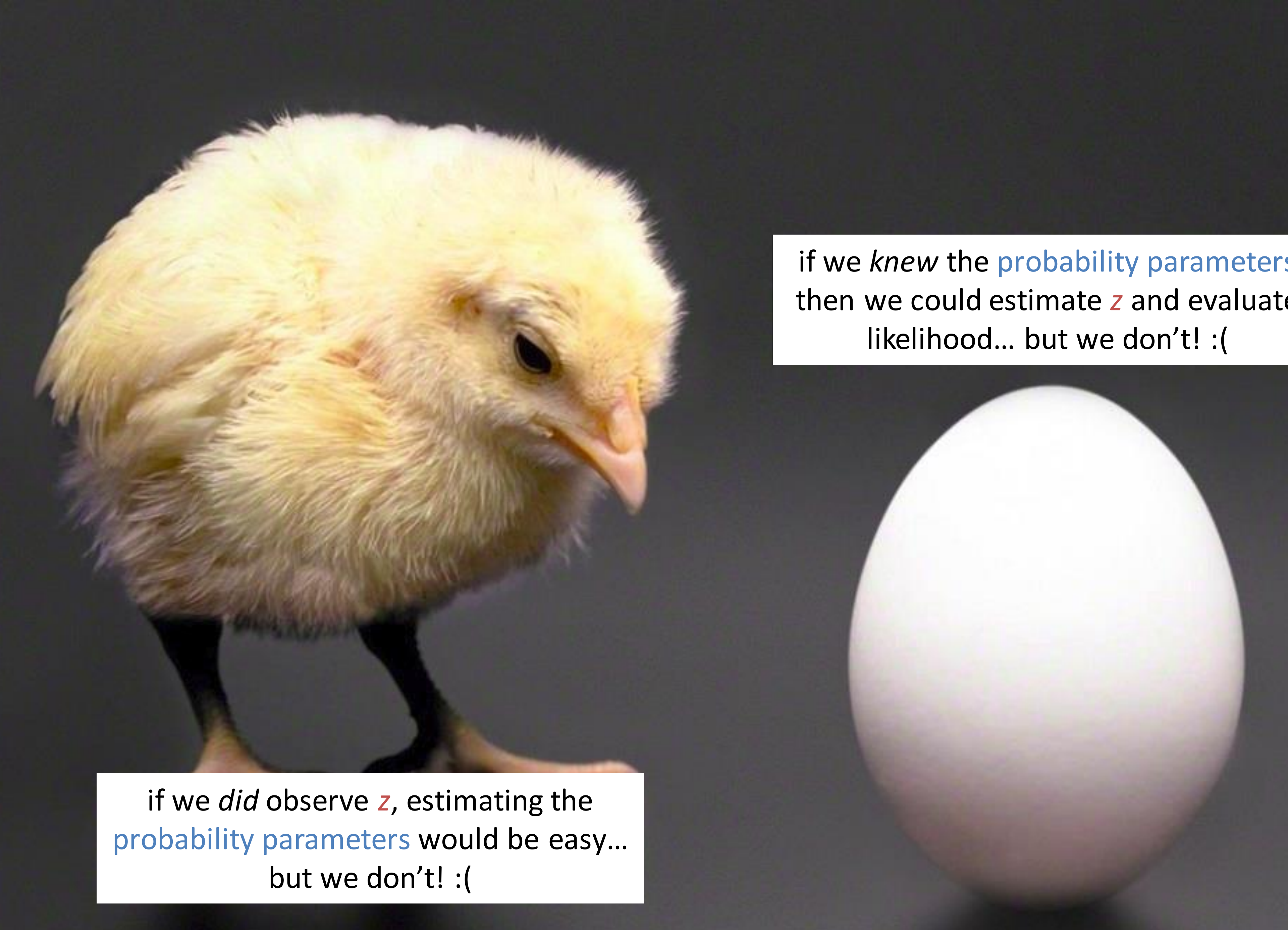
$$p(w_1, w_2, \dots, w_N) = \left(\sum_{z_1} p(z_1, w)\right)\left(\sum_{z_2} p(z_2, w)\right) \cdots \left(\sum_{z_N} p(z_N, w)\right)$$

if we *did* observe z, estimating the probability parameters would be easy… but we don't! :(

if we *knew* the probability parameters then we could estimate z and evaluate likelihood… but we don't! :(



http://blog.innotas.com/wp-content/uploads/2015/08/chicken-or-egg-cropped1.jpg



if we *knew* the probability parameter
then we could estimate z and evaluat
likelihood… but we don't! :(

if we *did* observe z, estimating the probability parameters would be easy… but we don't! :(

http://blog.innotas.com/wp-content/uploads/2015/08/chicken-or-egg-cropped1.jpg


Expectation Maximization
if we kn
ility parameters
then we c
z and evaluate
don't! :(
if we did
ating the
probability pa
easy…
but
http://blog.innotas.com/wp-content/uploads/2015/

Expectation Maximization (EM)

0. Assume *some* value for your parameters

Two step, iterative algorithm

1. E-step: count under uncertainty (compute expectations)

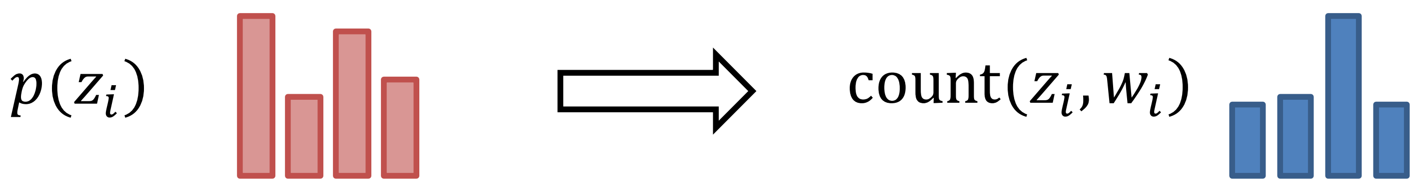
2. M-step: maximize log-likelihood, assuming these uncertain counts

Expectation Maximization (EM): E-step

0. Assume *some* value for your parameters

Two step, iterative algorithm

1. E-step: count under uncertainty, assuming these parameters



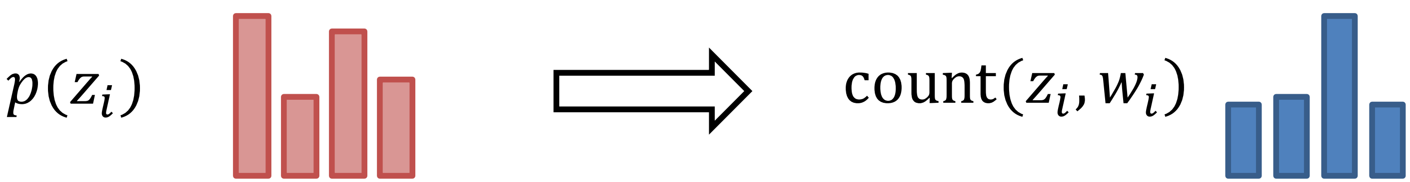
2. M-step: maximize log-likelihood, assuming these uncertain counts

Expectation Maximization (EM): E-step

0. Assume *some* value for your parameters

Two step, iterative algorithm

1. E-step: count under uncertainty, assuming these parameters



2. M- … se

uncer…

We’ve already seen this type of counting, when computing the gradient in maxent models.

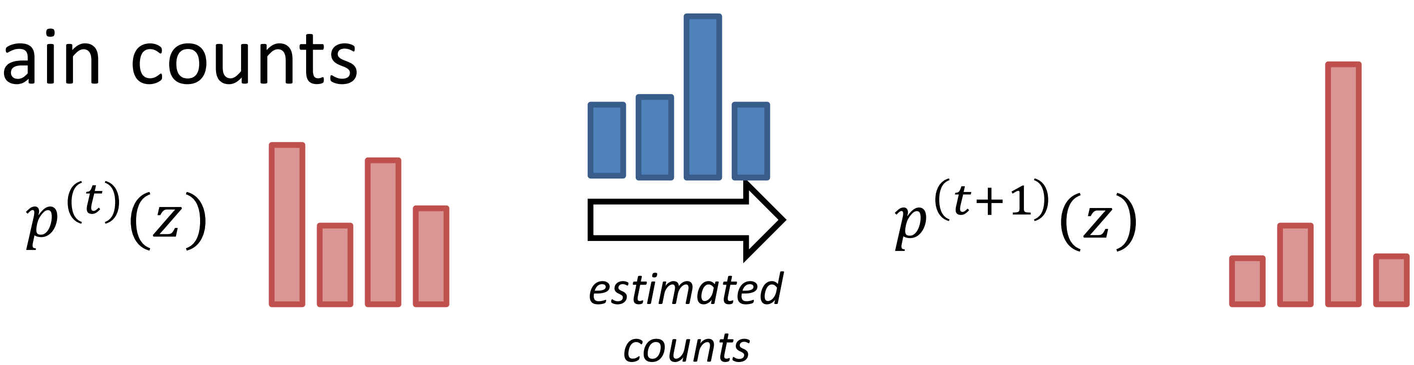
Expectation Maximization (EM): M-step

0. Assume *some* value for your parameters

Two step, iterative algorithm

1. E-step: count under uncertainty, assuming these parameters

2. M-step: maximize log-likelihood, assuming these uncertain counts



EM Math

$$\max_{\theta} \mathbb{E}_{z \sim p_{\theta^{(t)}}(\cdot|w)} [\log p_{\theta}(z, w)]$$

EM Math

E-step: count under uncertainty

$$\max_{\theta} \mathbb{E}_{z \sim p_{\theta^{(t)}}(\cdot|w)} \left[\log p_{\theta}(z, w)\right]$$

M-step: maximize log-likelihood

EM Math

E-step: count under uncertainty

old parameters

$$\max_{\theta} \mathbb{E}_{z \sim p_{\theta^{(t)}}(\cdot|w)} [\log p_{\theta}(z, w)]$$

posterior distribution

M-step: maximize log-likelihood

EM Math

E-step: count under uncertainty

old parameters

$$\max_{\theta} \mathbb{E}_{z \sim p_{\theta^{(t)}}(\cdot|w)} [\log p_{\theta}(z, w)]$$

new parameters

posterior distribution

new parameters

M-step: maximize log-likelihood

Three Coins/Unigram With Class Example

Imagine three coins

Flip 1st coin (penny)

If heads: flip 2nd coin (dollar coin)

If tails: flip 3rd coin (dime)

Three Coins/Unigram With Class Example

Imagine three coins

Flip 1st coin (penny) ← don't observe this

If heads: flip 2nd coin (dollar coin)

If tails: flip 3rd coin (dime)

only observe these (record heads vs. tails outcome)

Three Coins/Unigram With Class Example

Imagine three coins

Flip 1st coin (penny)

unobserved:
vowel or constonant?
part of speech?

If heads: flip 2nd coin (dollar coin)

If tails: flip 3rd coin (dime)

observed:
a, *b*, *e*, etc.
We run the code, vs.
The *run* failed

Three Coins/Unigram With Class Example

Imagine three coins

Flip 1st coin (penny)

$p(\text{heads}) = \lambda$ $\quad$ $p(\text{tails}) = 1 - \lambda$

If heads: flip 2nd coin (dollar coin)

$p(\text{heads}) = \gamma$ $\quad$ $p(\text{tails}) = 1 - \gamma$

If tails: flip 3rd coin (dime)

$p(\text{heads}) = \psi$ $\quad$ $p(\text{tails}) = 1 - \psi$

Three Coins/Unigram With Class Example

Imagine three coins

$p(\text{heads}) = \lambda$ $\quad$ $p(\text{heads}) = \gamma$ $\quad$ $p(\text{heads}) = \psi$

$p(\text{tails}) = 1 - \lambda$ $\quad$ $p(\text{tails}) = 1 - \gamma$ $\quad$ $p(\text{tails}) = 1 - \psi$

Three parameters to estimate: λ, γ, and ψ

Three Coins/Unigram With Class Example

H H T H T H

H T H T T T

If *all* flips were observed

$p(\text{heads}) = \lambda$ $\quad p(\text{heads}) = \gamma$ $\quad p(\text{heads}) = \psi$

$p(\text{tails}) = 1 - \lambda$ $\quad p(\text{tails}) = 1 - \gamma$ $\quad p(\text{tails}) = 1 - \psi$

Three Coins/Unigram With Class Example

H H T H T H
H T H T T T

If *all* flips were observed

$p(\text{heads}) = \lambda$ $\quad$ $p(\text{heads}) = \gamma$ $\quad$ $p(\text{heads}) = \psi$

$p(\text{tails}) = 1 - \lambda$ $\quad$ $p(\text{tails}) = 1 - \gamma$ $\quad$ $p(\text{tails}) = 1 - \psi$

$$p(\text{heads}) = \frac{4}{6} \qquad p(\text{heads}) = \frac{1}{4} \qquad p(\text{heads}) = \frac{1}{2}$$

$$p(\text{tails}) = \frac{2}{6} \qquad p(\text{tails}) = \frac{3}{4} \qquad p(\text{tails}) = \frac{1}{2}$$

Three Coins/Unigram With Class Example

~~*H H T H T H*~~

H T H T T T

But not all flips are observed → *set* parameter values

$p(\text{heads}) = \lambda = .6$

$p(\text{tails}) = .4$

$p(\text{heads}) = .8$

$p(\text{tails}) = .2$

$p(\text{heads}) = .6$

$p(\text{tails}) = .4$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

But not all flips are observed → *set* parameter values

$p(\text{heads}) = \lambda = .6$ $p(\text{heads}) = .8$ $p(\text{heads}) = .6$

$p(\text{tails}) = .4$ $p(\text{tails}) = .2$ $p(\text{tails}) = .4$

Use these values to compute posteriors

$$p(\text{heads} \mid \text{observed item H}) = \frac{p(\text{heads} \ \& \ \text{H})}{p(\text{H})}$$

$$p(\text{heads} \mid \text{observed item T}) = \frac{p(\text{heads} \ \& \ \text{T})}{p(\text{T})}$$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

But not all flips are observed → *set* parameter values

$p(\text{heads}) = \lambda = .6$ $p(\text{heads}) = .8$ $p(\text{heads}) = .6$

$p(\text{tails}) = .4$ $p(\text{tails}) = .2$ $p(\text{tails}) = .4$

Use these values to compute posteriors

rewrite joint using Bayes rule

$$p(\text{heads} \mid \text{observed item H}) = \frac{p(\text{H} \mid \text{heads})\,p(\text{heads})}{p(\text{H})}$$

marginal likelihood

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

But not all flips are observed → *set* parameter values

$p(\text{heads}) = \lambda = .6$ $p(\text{heads}) = .8$ $p(\text{heads}) = .6$

$p(\text{tails}) = .4$ $p(\text{tails}) = .2$ $p(\text{tails}) = .4$

Use these values to compute posteriors

$$p(\text{heads} \mid \text{observed item H}) = \frac{p(\text{H} \mid \text{heads})p(\text{heads})}{p(\text{H})}$$

$p(\text{H} \mid \text{heads}) = .8$ $p(\text{T} \mid \text{heads}) = .2$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

But not all flips are observed → *set* parameter values

$p(\text{heads}) = \lambda = .6$ $\quad$ $p(\text{heads}) = .8$ $\quad$ $p(\text{heads}) = .6$

$p(\text{tails}) = .4$ $\quad$ $p(\text{tails}) = .2$ $\quad$ $p(\text{tails}) = .4$

Use these values to compute posteriors

$$p(\text{heads} \mid \text{observed item H}) = \frac{p(\text{H} \mid \text{heads})p(\text{heads})}{p(\text{H})}$$

$p(\text{H} \mid \text{heads}) = .8$ $\quad$ $p(\text{T} \mid \text{heads}) = .2$

$$p(\text{H}) = p(\text{H} \mid \text{heads}) * p(\text{heads}) + p(\text{H} \mid \text{tails}) * p(\text{tails})$$
$$= .8 * .6 + .6 * .4$$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

Use posteriors to update parameters

$$p(\text{heads} \mid \text{obs. H}) = \frac{p(\text{H} \mid \text{heads})\,p(\text{heads})}{p(\text{H})} = \frac{.8 * .6}{.8 * .6 + .6 * .4} \approx 0.667$$

$$p(\text{heads} \mid \text{obs. T}) = \frac{p(\text{T} \mid \text{heads})\,p(\text{heads})}{p(\text{T})} = \frac{.2 * .6}{.2 * .6 + .6 * .4} \approx 0.334$$

(in general, p(heads | obs. H) and p(heads | obs. T) do NOT sum to 1)

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

Use posteriors to update parameters

$$p(\text{heads} \mid \text{obs. H}) = \frac{p(\text{H} \mid \text{heads})p(\text{heads})}{p(\text{H})}$$

$$= \frac{.8 * .6}{.8 * .6 + .6 * .4} \approx 0.667$$

$$p(\text{heads} \mid \text{obs. T}) = \frac{p(\text{T} \mid \text{heads})p(\text{heads})}{p(\text{T})}$$

$$= \frac{.2 * .6}{.2 * .6 + .6 * .4} \approx 0.334$$

(in general, p(heads | obs. H) and p(heads | obs. T) do NOT sum to 1)

fully observed setting

$$p(\text{heads}) = \frac{\#\ \text{heads from penny}}{\#\ \text{total flips of penny}}$$

our setting: partially-observed

$$p(\text{heads}) = \frac{\#\ \textit{expected}\ \text{heads from penny}}{\#\ \text{total flips of penny}}$$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

Use posteriors to update parameters

$$p(\text{heads} \mid \text{obs. H}) = \frac{p(\text{H} \mid \text{heads})p(\text{heads})}{p(\text{H})} = \frac{.8 * .6}{.8 * .6 + .6 * .4} \approx 0.667$$

$$p(\text{heads} \mid \text{obs. T}) = \frac{p(\text{T} \mid \text{heads})p(\text{heads})}{p(\text{T})} = \frac{.2 * .6}{.2 * .6 + .6 * .4} \approx 0.334$$

our setting: partially-observed

$$p^{(t+1)}(\text{heads}) = \frac{\#\ \textit{expected}\ \text{heads from penny}}{\#\ \text{total flips of penny}} = \frac{\mathbb{E}_{p^{(t)}}[\#\ \textit{expected}\ \text{heads from penny}]}{\#\ \text{total flips of penny}}$$

Three Coins/Unigram With Class Example

~~H H T H T H~~

H T H T T T

Use posteriors to update parameters

$$p(\text{heads} \mid \text{obs. H}) = \frac{p(\text{H} \mid \text{heads})p(\text{heads})}{p(\text{H})} = \frac{.8 * .6}{.8 * .6 + .6 * .4} \approx 0.667$$

$$p(\text{heads} \mid \text{obs. T}) = \frac{p(\text{T} \mid \text{heads})p(\text{heads})}{p(\text{T})} = \frac{.2 * .6}{.2 * .6 + .6 * .4} \approx 0.334$$

our setting: partially-observed

$$p^{(t+1)}(\text{heads}) = \frac{\#\ \textit{expected}\ \text{heads from penny}}{\#\ \text{total flips of penny}} = \frac{\mathbb{E}_{p^{(t)}}[\#\ \textit{expected}\ \text{heads from penny}]}{\#\ \text{total flips of penny}} = \frac{2 * p(\text{heads} \mid \text{obs. H}) + 4 * p(\text{heads} \mid \text{obs.}\ T)}{6} \approx 0.444$$

Expectation Maximization (EM)

0. Assume *some* value for your parameters

Two step, iterative algorithm:

1. E-step: count under uncertainty (compute expectations)

2. M-step: maximize log-likelihood, assuming these uncertain counts

Related to EM

Latent clustering

K-means:
https://www.csee.umbc.edu/courses/undergraduate/473/f17/kmeans/

Gaussian mixture modeling