89688: Statistical Machine Translation

Lecture 3: Statistical Methods, IBM models and the EM algorithm

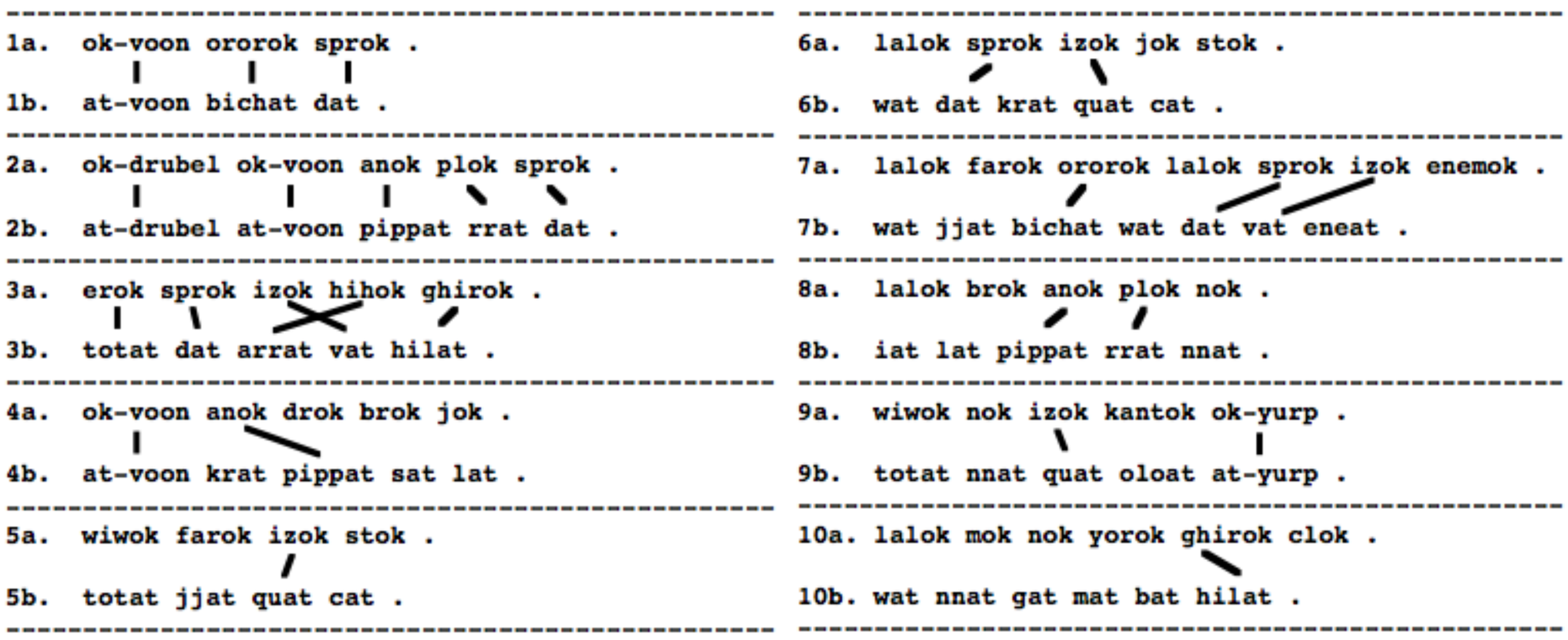
April 2020

Roee Aharoni
Computer Science Department
Bar Ilan University

Based in part on slides from Edinburgh University's MT class and by Kevin Knight

How can we learn to translate?

👽 vs. 👾



How can we learn to translate?

How can we learn to translate?

- What can we learn from?

How can we learn to translate?

- What can we learn from?
  - Parallel Corpora (human translations)

How can we learn to translate?

- What can we learn from?
  - Parallel Corpora (human translations)
  - Monolingual corpora (books, wikipedia, the web…)

How can we learn to translate?

- What can we learn from?
  - Parallel Corpora (human translations)
  - Monolingual corpora (books, wikipedia, the web…)
- What do we learn?

How can we learn to translate?

- What can we learn from?
  - Parallel Corpora (human translations)
  - Monolingual corpora (books, wikipedia, the web…)
- What do we learn?

Translation dictionary:

anok - pippat	ok-yurp - at-yurp
erok - total	ok-voon - at-voon
ghirok - hilat	ororok - bichat
hihok - arrat	plok - rrat
izok - vat	sprok - dat
ok-drubel - at-drubel	zanzanok - zanzanat

Translation Model

How can we learn to translate?

- What can we learn from?
  - Parallel Corpora (human translations)
  - Monolingual corpora (books, wikipedia, the web…)
- What do we learn?

Translation dictionary:

anok - pippat	ok-yurp - at-yurp
erok - total	ok-voon - at-voon
ghirok - hilat	ororok - bichat
hihok - arrat	plok - rrat
izok - vat	sprok - dat
ok-drubel - at-drubel	zanzanok - zanzanat

Translation Model

Word pair counts:

1 . erok	1 hihok yorok
7 . lalok	1 izok enemok
2 . ok-drubel	2 izok hihok
2 . ok-voon	1 izok jok
3 . wiwok	1 izok kantok
1 anok drok	1 izok stok
1 anok ghirok	1 izok vok

Language Model

Let’s try to formalize this…

Let’s try to formalize this…

- **Learning/Training** Phase

```
def learn(parallel_data):
  # do something
  return parameters
```

$$L : (\Sigma_f^* \times \Sigma_e^*)^* \rightarrow \Theta$$

Let’s try to formalize this…

- **Learning/Training** Phase

```
def learn(parallel_data):
  # do something
  return parameters
```

$$L : (\Sigma_f^* \times \Sigma_e^*)^* \rightarrow \Theta$$

- **Inference** Phase

```
def translate(French, parameters):
  # do something
  return English
```

$$T : \Sigma_f^* \times \Theta \rightarrow \Sigma_e^*$$

Let's try to formalize this...

- **Learning/Training** Phase

```
def learn(parallel_data):
  # do something
  return parameters
```

$$L : (\Sigma_f^* \times \Sigma_e^*)^* \rightarrow \Theta$$

- **Inference** Phase

```
def translate(French, parameters):
  # do something
  return English
```

$$T : \Sigma_f^* \times \Theta \rightarrow \Sigma_e^*$$

- **How?**

Using probability: $T(f, \theta) = \arg\max_{e \in \Sigma_e^*} p_\theta(e|f)$

Why probability?

Why probability?

Al-Khalil ibn Ahmad al-Farahidi, 718-786

Image: Wikipedia

Why probability?

- Formalizes…

Al-Khalil ibn Ahmad al-Farahidi, 718-786

Image: Wikipedia

Why probability?

- Formalizes…
  - The concept of **models**

Al-Khalil ibn Ahmad al-Farahidi, 718-786
Image: Wikipedia

Why probability?

- Formalizes…
  - The concept of **models**
  - The concept of **data**

Al-Khalil ibn Ahmad al-Farahidi, 718-786
Image: Wikipedia

Why probability?

- Formalizes…
  - The concept of **models**
  - The concept of **data**
  - The concept of **learning**

Al-Khalil ibn Ahmad al-Farahidi, 718-786
Image: Wikipedia

Why probability?

- Formalizes…
  - The concept of **models**
  - The concept of **data**
  - The concept of **learning**
  - The concept of **inference** (prediction)

Al-Khalil ibn Ahmad al-Farahidi, 718-786

Image: Wikipedia

Why probability?

- Formalizes…
  - The concept of **models**
  - The concept of **data**
  - The concept of **learning**
  - The concept of **inference** (prediction)
- Enables to model **ambiguity**

Al-Khalil ibn Ahmad al-Farahidi, 718-786
Image: Wikipedia

Translation as a probabilistic problem

Translation as a probabilistic problem

- We would like to **model** the **probability** of a **translation** given a **source sentence**

Translation as a probabilistic problem

- We would like to **model** the **probability** of a **translation** given a **source sentence**

$$p(English|Chinese) =$$

Translation as a probabilistic problem

$$p(English|Chinese) =$$

- We would like to **model** the **probability** of a **translation** given a **source sentence**
- We can use **Bayes Rule**

Translation as a probabilistic problem

- We would like to **model** the **probability** of a **translation** given a **source sentence**

$$p(English|Chinese) = \frac{p(English) \times p(Chinese|English)}{p(Chinese)}$$

- We can use **Bayes Rule**

Translation as a probabilistic problem

- We would like to **model** the **probability** of a **translation** given a **source sentence**
- We can use **Bayes Rule**
  - Why would we want that?

$$p(English|Chinese) = \frac{p(English) \times p(Chinese|English)}{p(Chinese)}$$

Translation as a probabilistic problem

- We would like to **model** the **probability** of a **translation** given a **source sentence**
- We can use **Bayes Rule**
  - Why would we want that?

$$\frac{p(English) \times p(Chinese|English)}{p(Chinese)}$$

language model

translation model

normalization (ensures we're working with valid probabilities).

How do we define *p(Chinese|English)*?

How do we define *p(Chinese|English)*?

- IBM Models (Brown, Dellapietra, Dellapietra, and Mercer, 93’)

How do we define *p(Chinese|English)*?

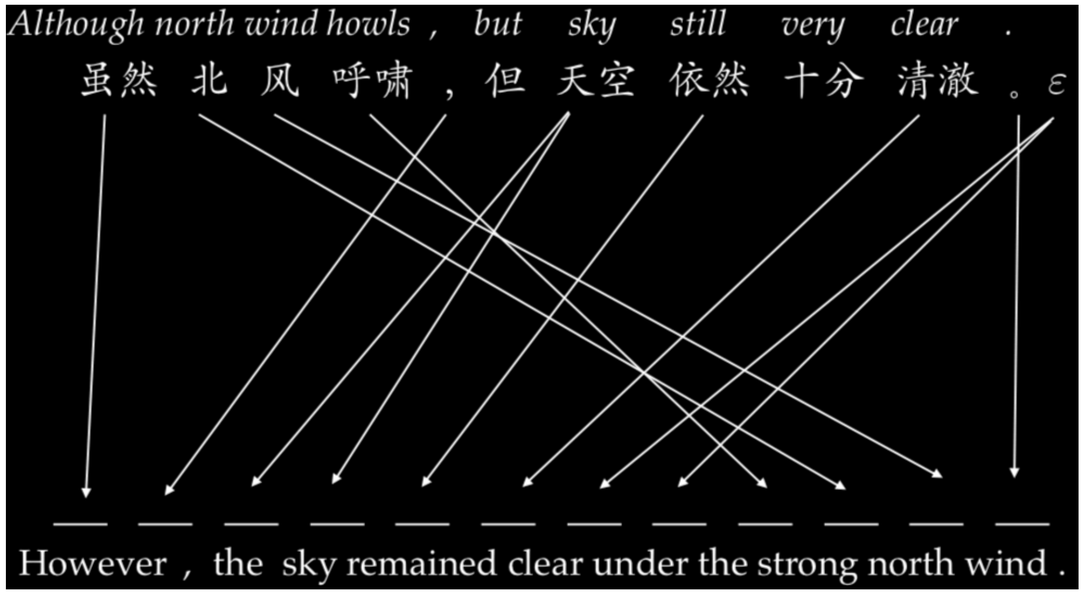
- IBM Models (Brown, Dellapietra, Dellapietra, and Mercer, 93’)
- “We define a concept of word-by-word **alignment**”

How do we define *p(Chinese|English)*?

- IBM Models (Brown, Dellapietra, Dellapietra, and Mercer, 93')
- "We define a concept of word-by-word **alignment**"

How do we define *p(Chinese|English)*?

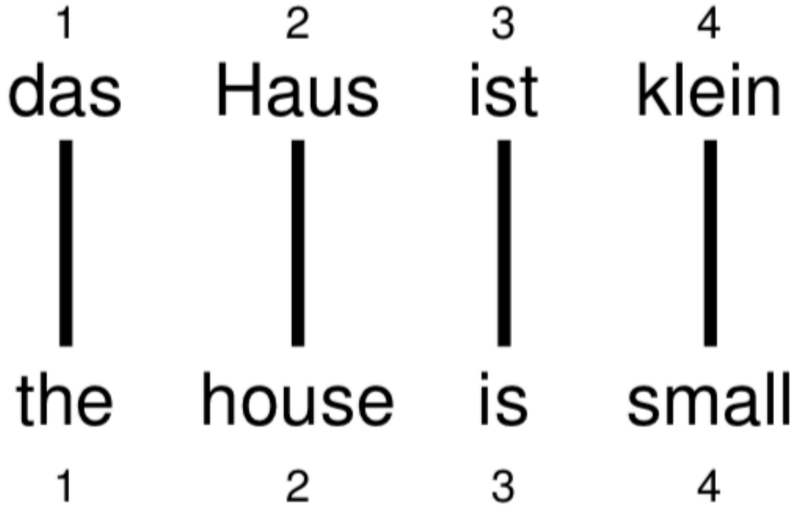
- IBM Models (Brown, Dellapietra, Dellapietra, and Mercer, 93')
- "We define a concept of word-by-word **alignment**"



Understanding Alignments

Understanding Alignments

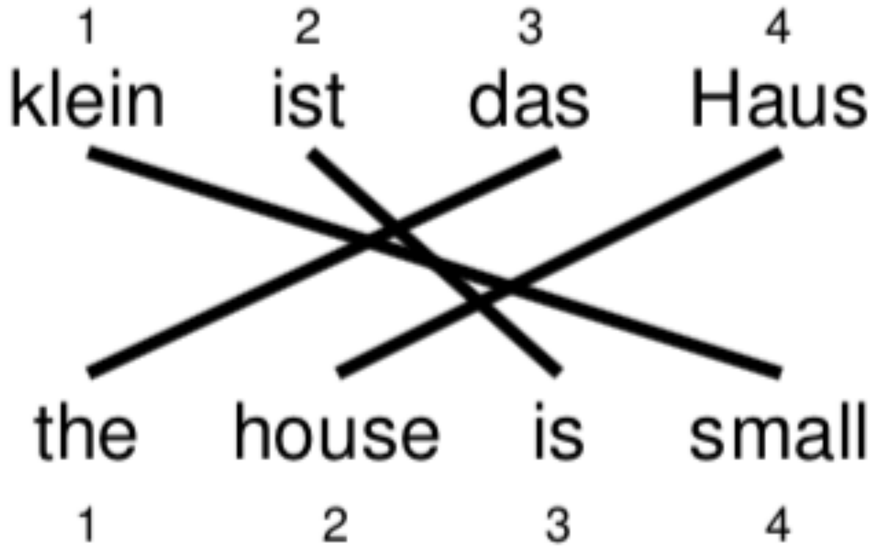
- Alignment function



$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4\}$$

Understanding Alignments

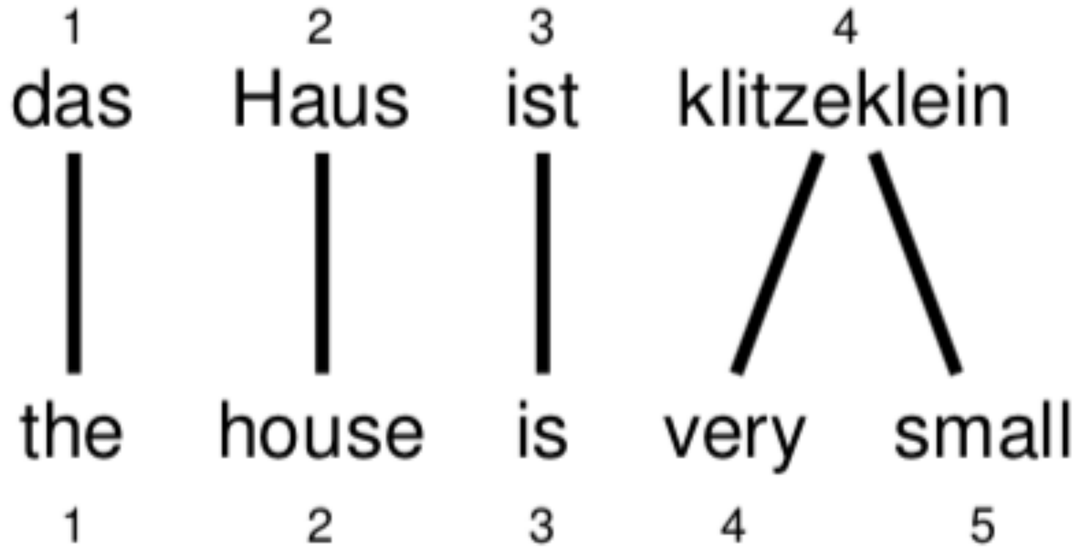
- Alignment function
- Reordering



$$a : \{1 \rightarrow 3, 2 \rightarrow 4, 3 \rightarrow 2, 4 \rightarrow 1\}$$

Understanding Alignments

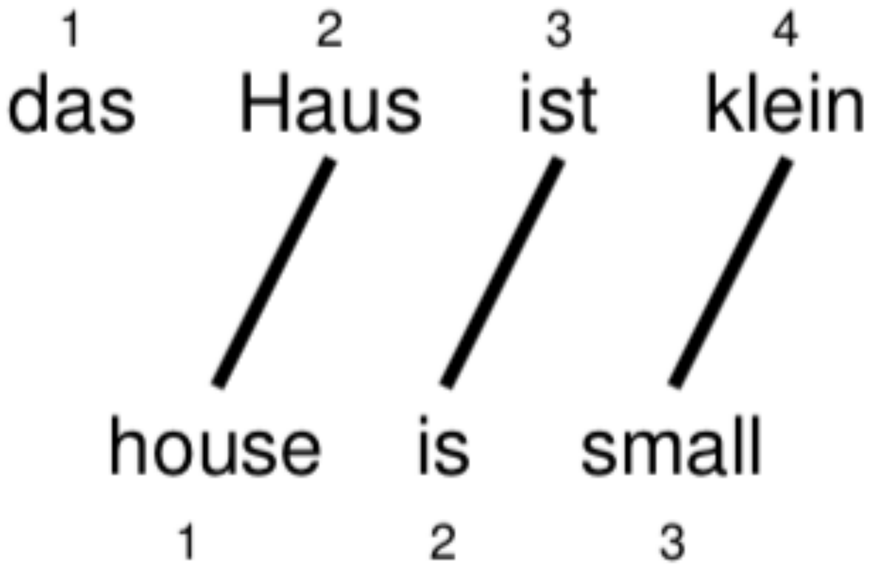
- Alignment function
- Reordering
- One-to-Many



$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4, 5 \rightarrow 4\}$$

Understanding Alignments

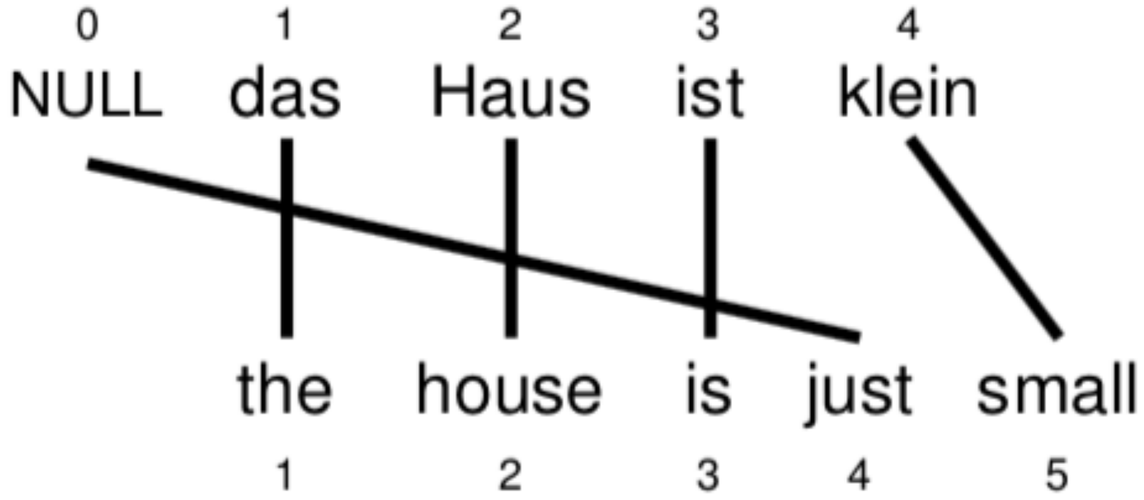
- Alignment function
- Reordering
- One-to-Many
- Dropping words



$$a : \{1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4\}$$

Understanding Alignments

- Alignment function
- Reordering
- One-to-Many
- Dropping words
- Inserting words



$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 0, 5 \rightarrow 4\}$$

IBM Model1's Generative Story

IBM Model1's Generative Story

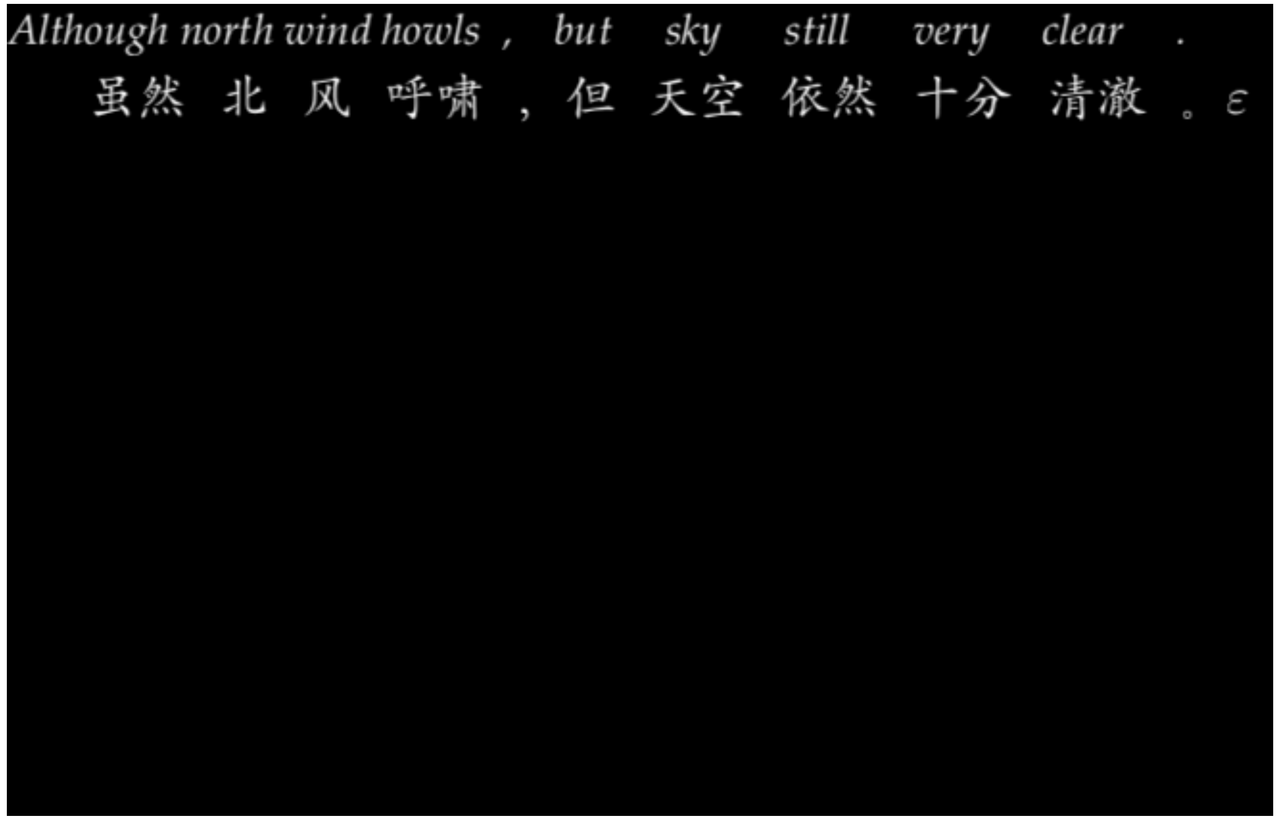
- Given a source sentence, how was the target sentence generated?

IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?

IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?



IBM Model1's Generative Story

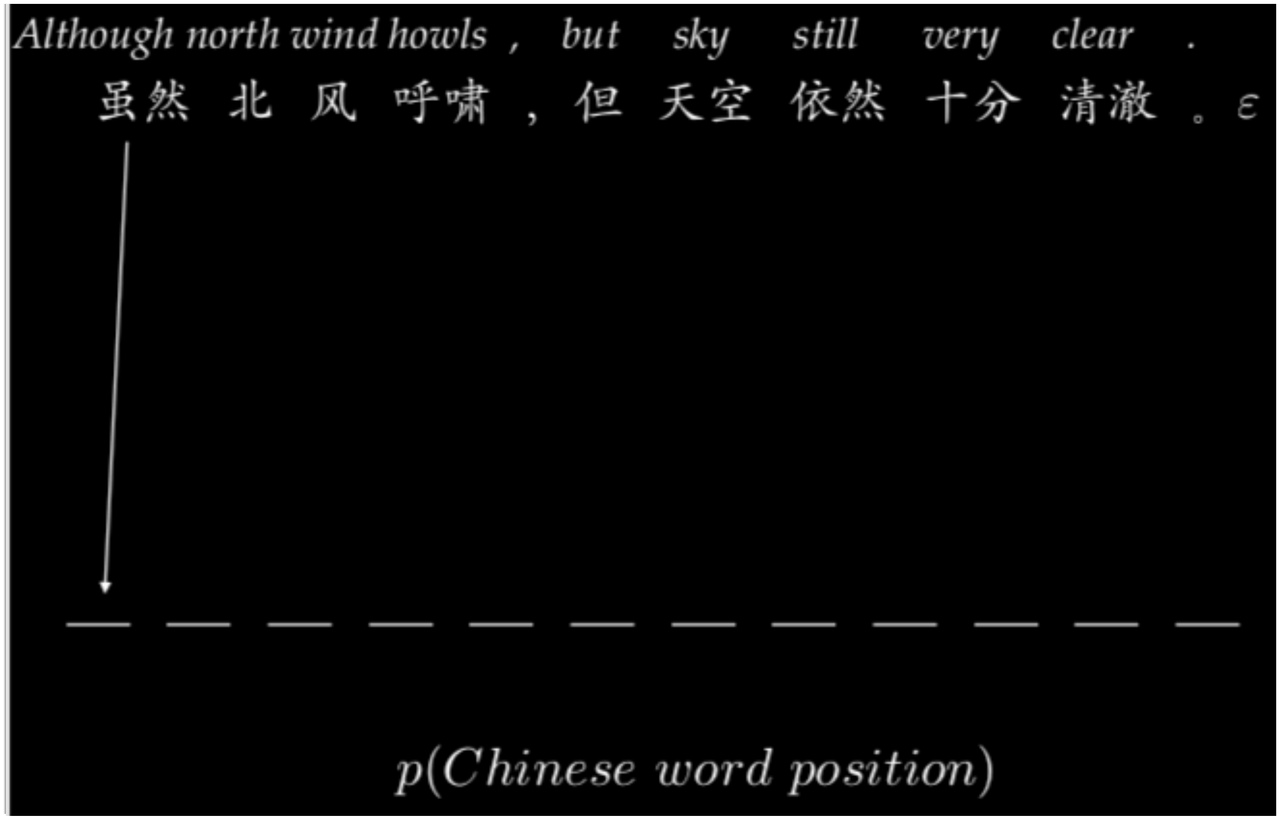
- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence

IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence
- For each target position:

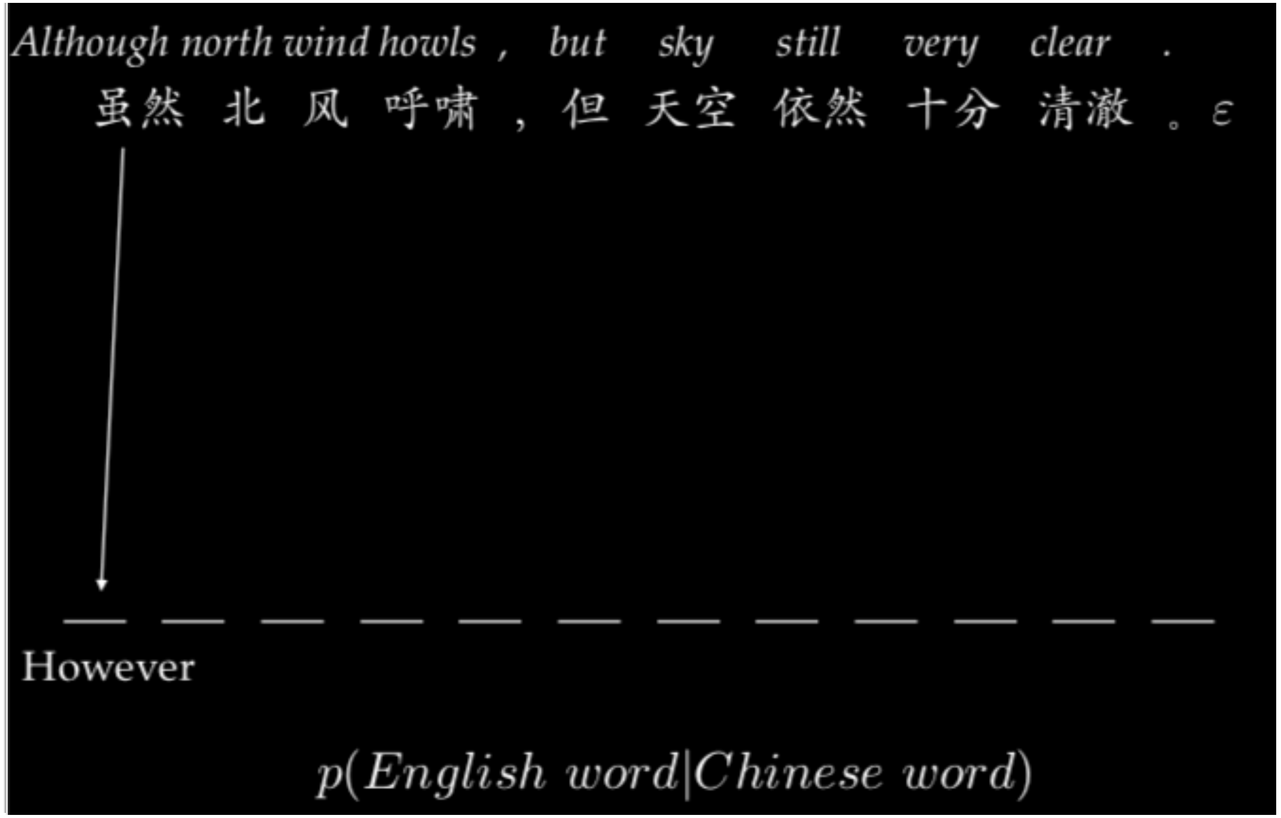
IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence
- For each target position:
  - Sample an alignment (to a position in the source)



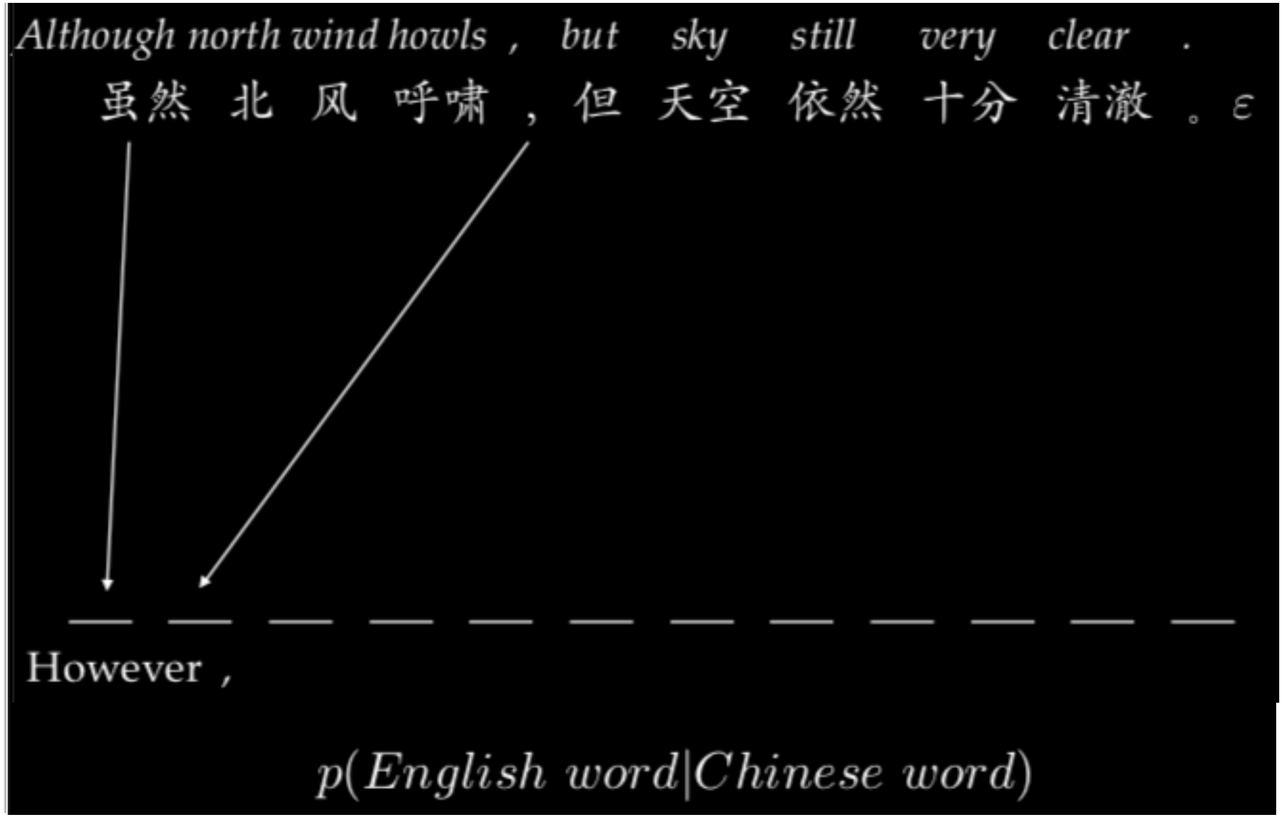
IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence
- For each target position:
  - Sample an alignment (to a position in the source)
  - Sample a word translation given this alignment



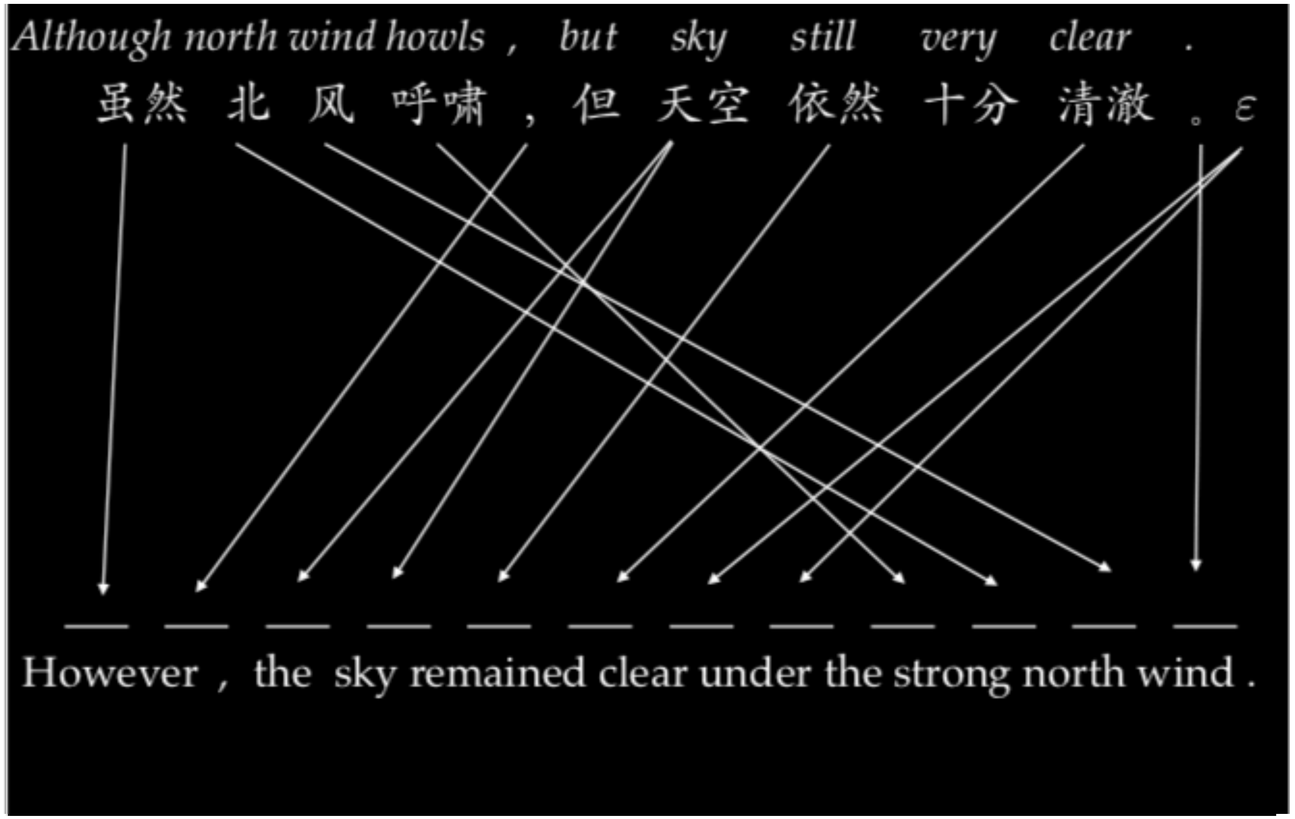
IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence
- For each target position:
  - Sample an alignment (to a position in the source)
  - Sample a word translation given this alignment
  - Repeat until done



IBM Model1's Generative Story

- Given a source sentence, how was the target sentence generated?
- Sample a length for the target sentence
- For each target position:
  - Sample an alignment (to a position in the source)
  - Sample a word translation given this alignment
  - Repeat until done



IBM Model1

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

IBM Model1

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

sample sentence length

IBM Model1

sample alignment

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

sample sentence length

IBM Model1

sample alignment

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

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)

sample alignment

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

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment

sample alignment

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

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence

sample alignment

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

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence
- I - foreign sentence length

sample alignment

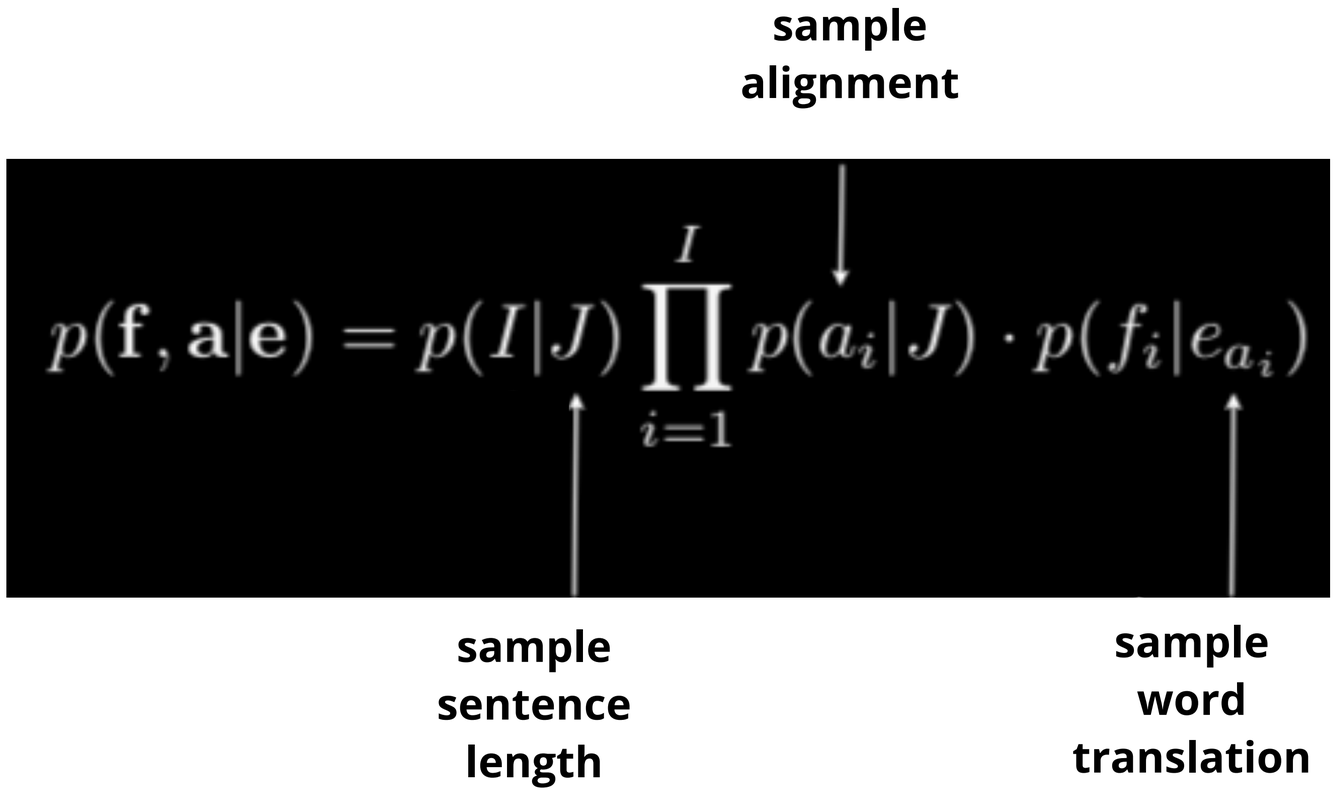
$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence
- I - foreign sentence length
- J - English sentence length



IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence
- I - foreign sentence length
- J - English sentence length
- a_i - alignment of i-th foreign word

sample alignment

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

sample sentence length

sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence
- I - foreign sentence length
- J - English sentence length
- a_i - alignment of i-th foreign word
- f_i - foreign word in position i

sample alignment

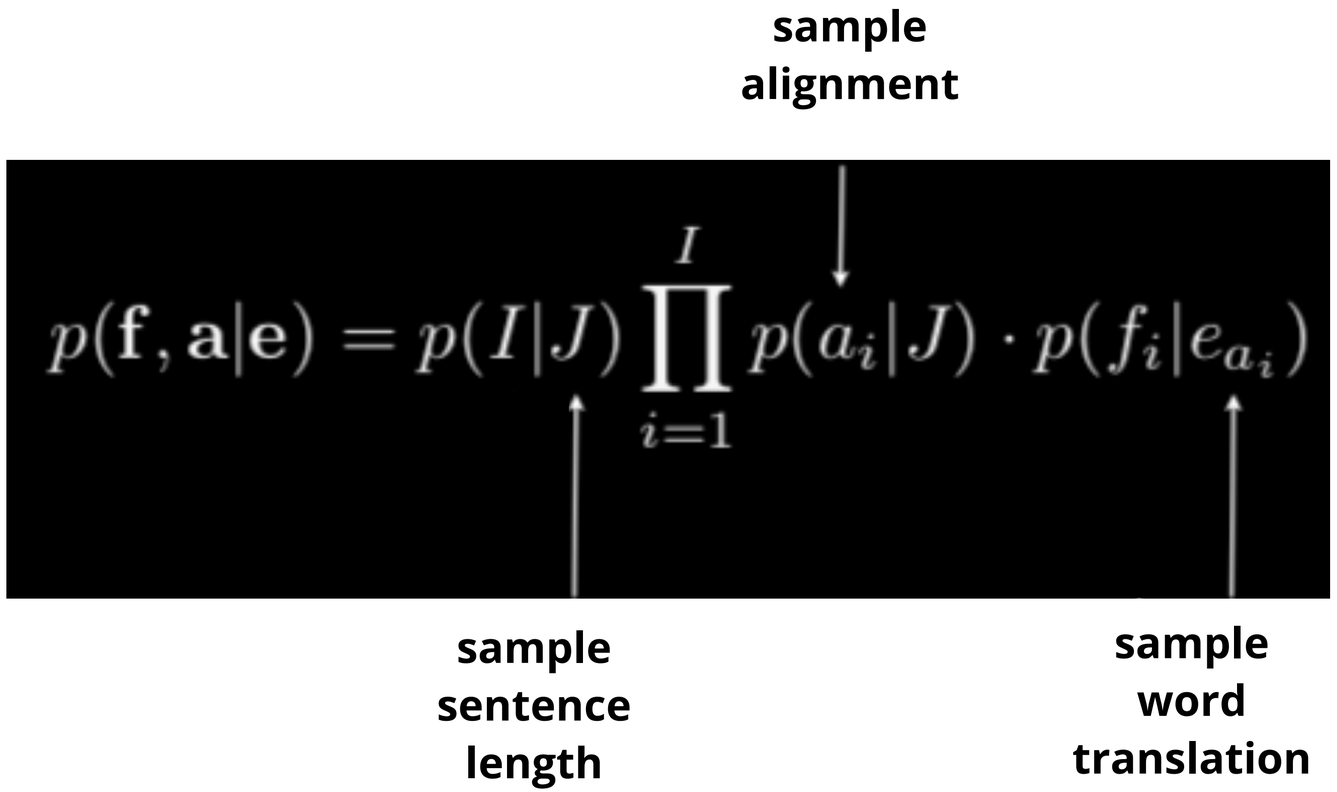
$$p(\mathbf{f}, \mathbf{a}|\mathbf{e}) = p(I|J) \prod_{i=1}^{I} p(a_i|J) \cdot p(f_i|e_{a_i})$$

sample sentence length

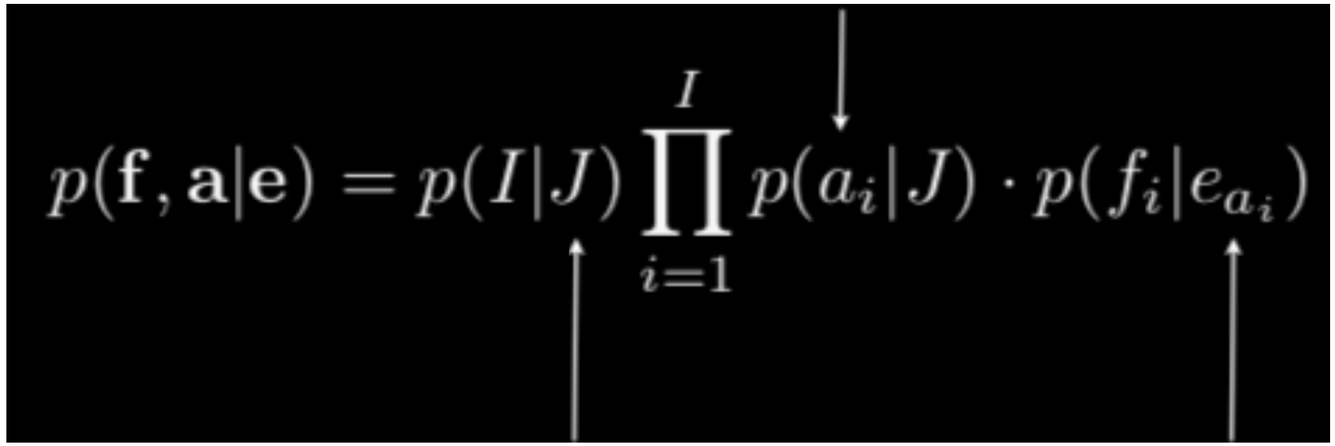
sample word translation

IBM Model1

- f - foreign sentence (Chinese)
- a - alignment
- e - English sentence
- I - foreign sentence length
- J - English sentence length
- a_i - alignment of i-th foreign word
- f_i - foreign word in position i
- e_{a_i} - English word in position a_i

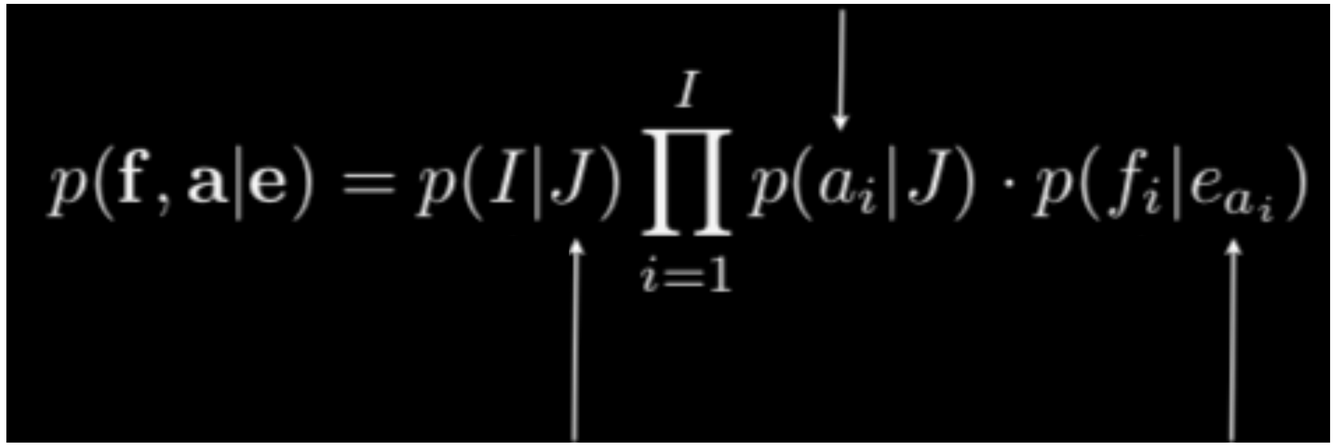


How can we learn this from data?



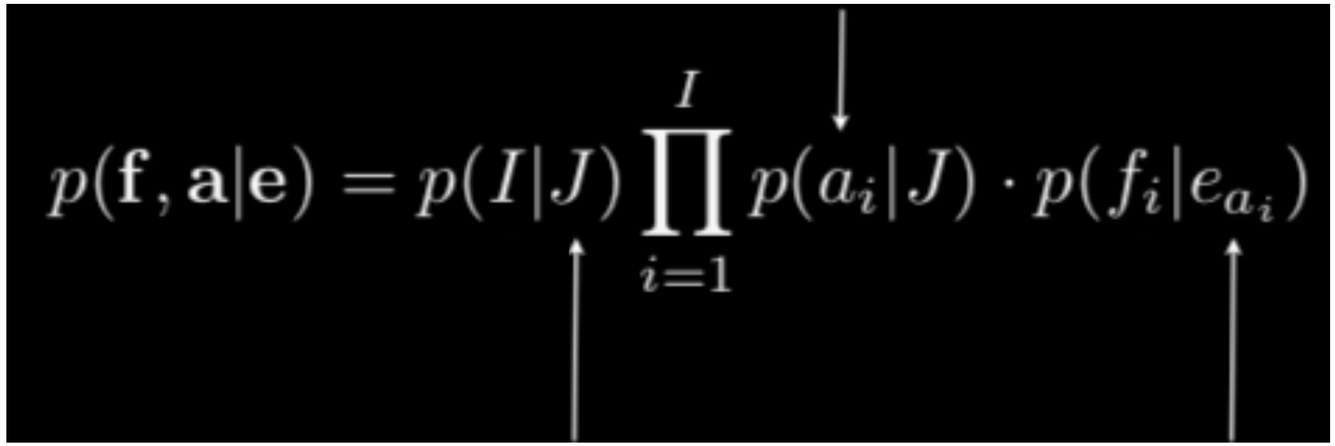
How can we learn this from data?

- What are the parameters we should learn?



How can we learn this from data?

- What are the parameters we should learn?
  - Sentence length distributions



How can we learn this from data?

- What are the parameters we should learn?
  - Sentence length distributions
  - Alignment distributions

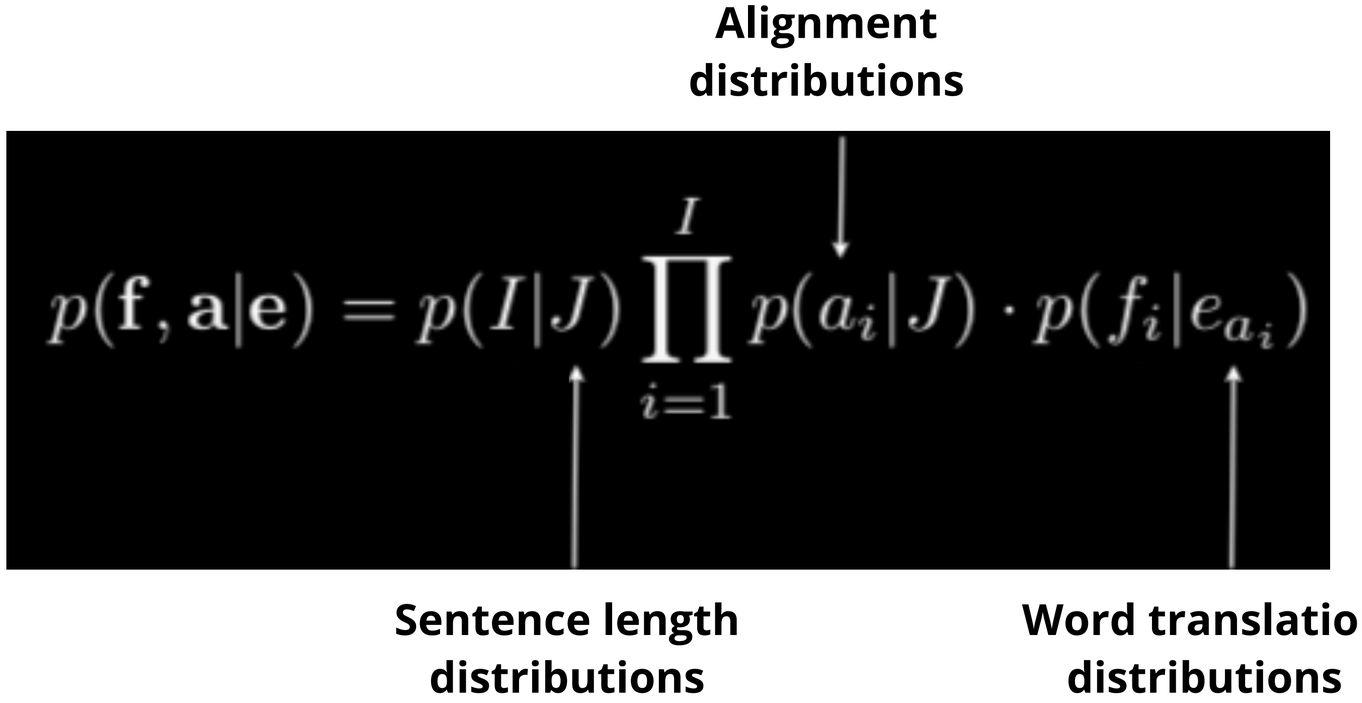
Alignment distributions

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

Sentence length distributions

How can we learn this from data?

- What are the parameters we should learn?
  - Sentence length distributions
  - Alignment distributions
  - Word translation distributions



How can we learn this from data?

How can we learn this from data?

- If we **know** the **alignments**, easy:

How can we learn this from data?

- If we **know** the **alignments**, easy:
  - $p(I|J)$ - learn by counting

$$p(I|J) = \frac{\textbf{\# Aligned Chinese sentences of length I}}{\textbf{\# English sentences of length J}}$$

How can we learn this from data?

- If we **know** the **alignments**, easy:
  - $p(I|J)$ - learn by counting
  - Alignment distributions - use uniform distribution

$$p(I|J) = \frac{\text{\# Aligned Chinese sentences of length I}}{\text{\# English sentences of length J}}$$

$$p(a\_i|J) = \frac{1}{J+1}$$

How can we learn this from data?

- If we **know** the **alignments**, easy:
  - $p(I|J)$ - learn by counting
  - Alignment distributions - use uniform distribution
  - Word translation distributions - again by counting

$$p(I|J) = \frac{\text{\# Aligned Chinese sentences of length I}}{\text{\# English sentences of length J}}$$

$$p(a\_i|J) = \frac{1}{J+1}$$

$$p(\text{however}|\text{然而}) = \frac{\text{\# times “然而” aligned to “however”}}{\text{\# times “然而” aligned to any word}}$$

How can we learn this from data?

- If we **know** the **alignments**, easy:
  - $p(I|J)$ - learn by counting
  - Alignment distributions - use uniform distribution
  - Word translation distributions - again by counting
- But do we know the alignments?

$$p(I|J) = \frac{\text{\# Aligned Chinese sentences of length I}}{\text{\# English sentences of length J}}$$

$$p(a\_i|J) = \frac{1}{J+1}$$

$$p(\text{however}|\text{然而}) = \frac{\text{\# times “然而” aligned to “however”}}{\text{\# times “然而” aligned to any word}}$$

Alignments as Latent Variables

Alignments as Latent Variables

- Parameters (translation probabilities) and alignments are both unknown!

Alignments as Latent Variables

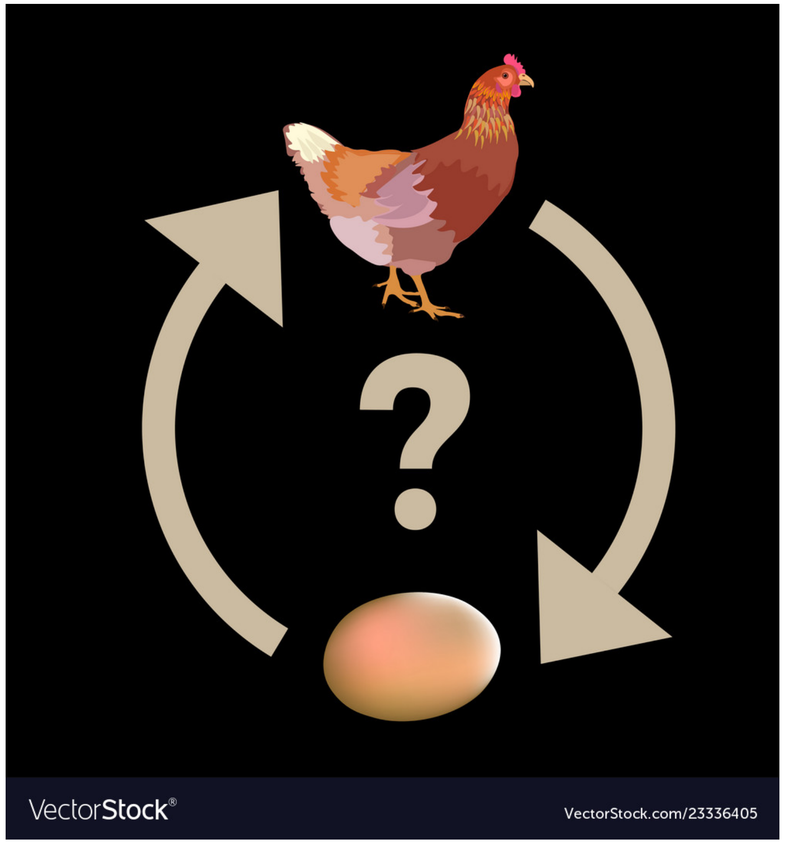
- Parameters (translation probabilities) and alignments are both unknown!
- If we knew the alignments in the training data, we could calculate the parameters by counting (prev. slide)

Alignments as Latent Variables

- Parameters (translation probabilities) and alignments are both unknown!
- If we knew the alignments in the training data, we could calculate the parameters by counting (prev. slide)
- If we knew the parameters, we could calculate the *expected* alignment counts

Alignments as Latent Variables

- Parameters (translation probabilities) and alignments are both unknown!
- If we knew the alignments in the training data, we could calculate the parameters by counting (prev. slide)
- If we knew the parameters, we could calculate the *expected* alignment counts



The Expectation-Maximization (EM) Algorithm

The Expectation-Maximization (EM) Algorithm

- Dempster, Laird and Rubin (1977)

The Expectation-Maximization (EM) Algorithm

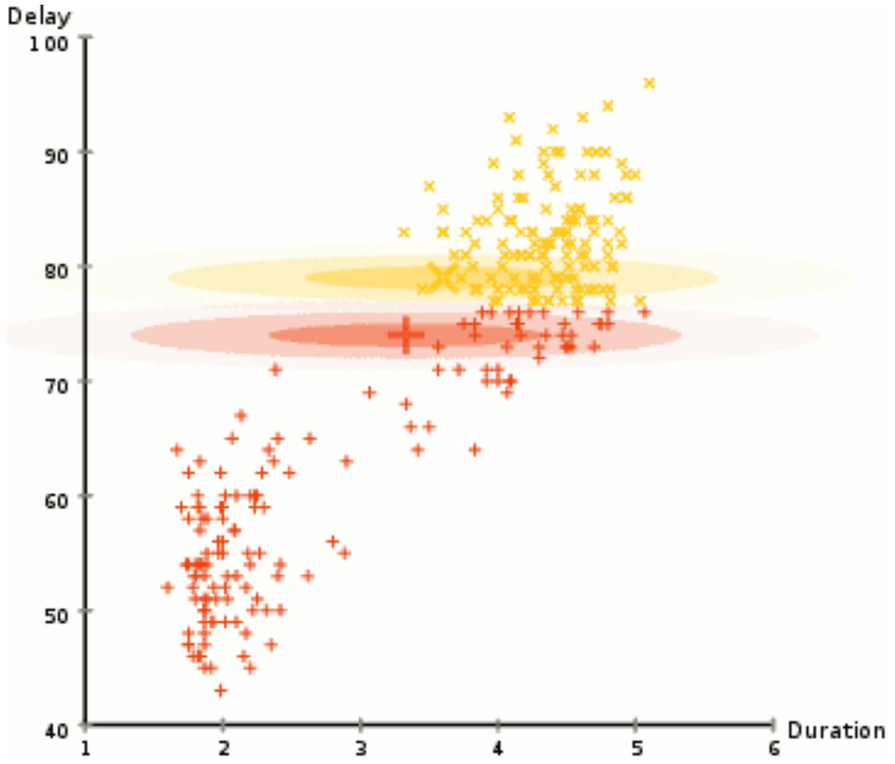
- Dempster, Laird and Rubin (1977)
- One of the most widely used algorithms in machine learning

The Expectation-Maximization (EM) Algorithm

- Dempster, Laird and Rubin (1977)
- One of the most widely used algorithms in machine learning
- Many applications, e.g. clustering

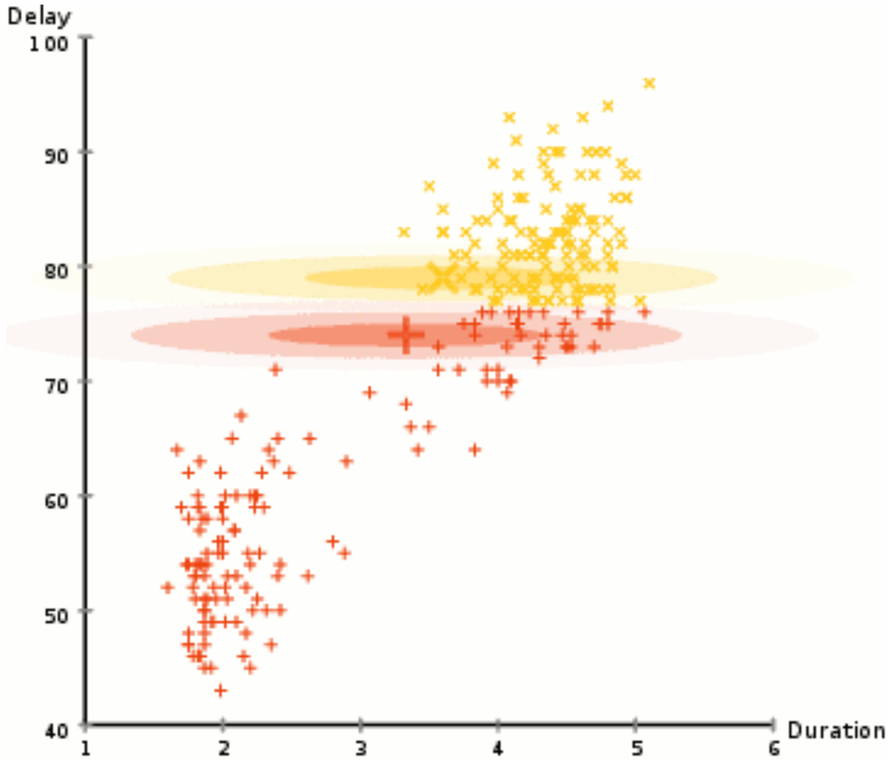
The Expectation-Maximization (EM) Algorithm

- Dempster, Laird and Rubin (1977)
- One of the most widely used algorithms in machine learning
- Many applications, e.g. clustering
- Main idea - start randomly, and iterate until convergence:



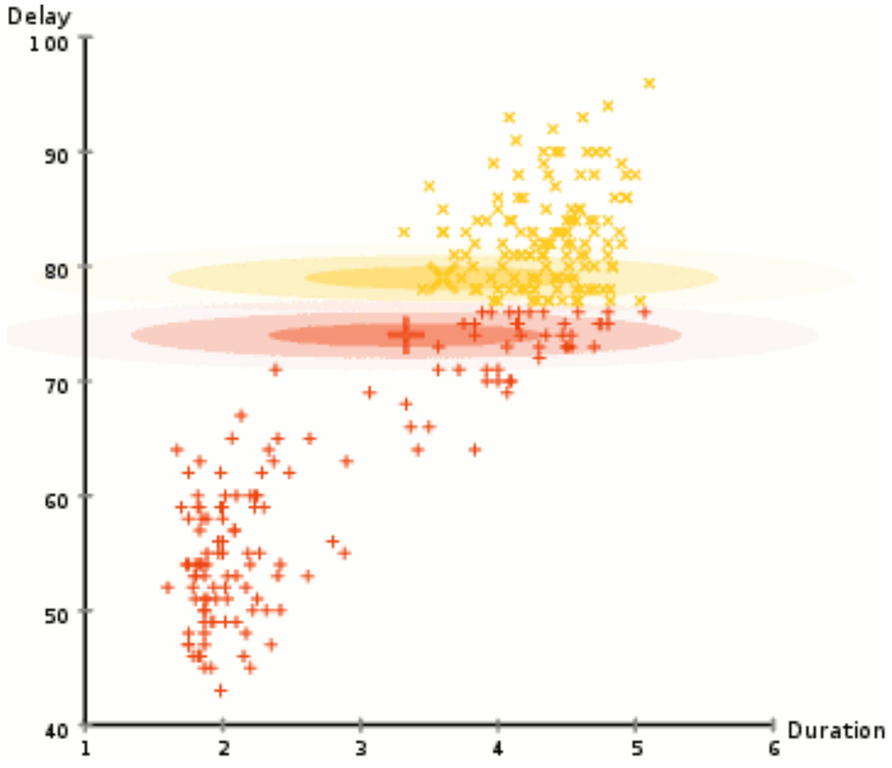
The Expectation-Maximization (EM) Algorithm

- Dempster, Laird and Rubin (1977)
- One of the most widely used algorithms in machine learning
- Many applications, e.g. clustering
- Main idea - start randomly, and iterate until convergence:
  - Calculate expected counts for missing data (expectation, or E-step) using current model



The Expectation-Maximization (EM) Algorithm

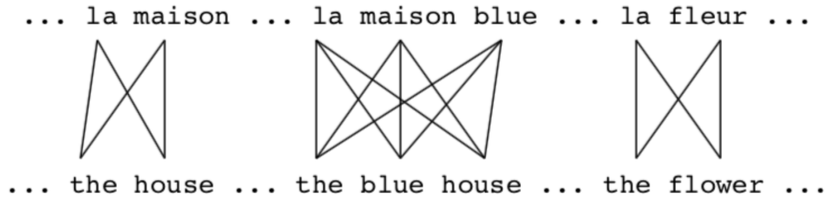
- Dempster, Laird and Rubin (1977)
- One of the most widely used algorithms in machine learning
- Many applications, e.g. clustering
- Main idea - start randomly, and iterate until convergence:
  - Calculate expected counts for missing data (expectation, or E-step) using current model
  - Find new model parameters that maximize the data likelihood (maximization, or M-step)



EM for IBM Model1 - Overview

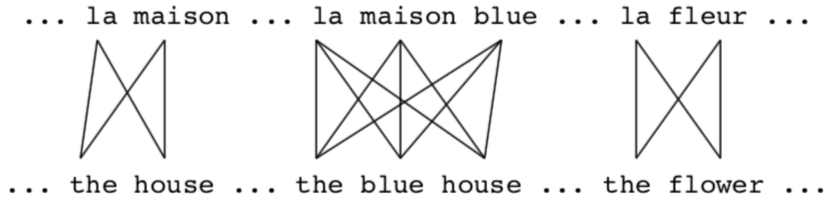
EM for IBM Model1 - Overview

- Start with all alignments equally likely



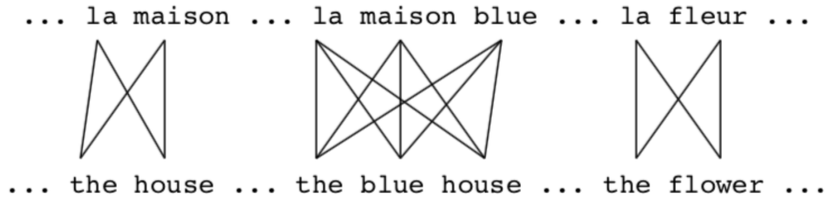
EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:



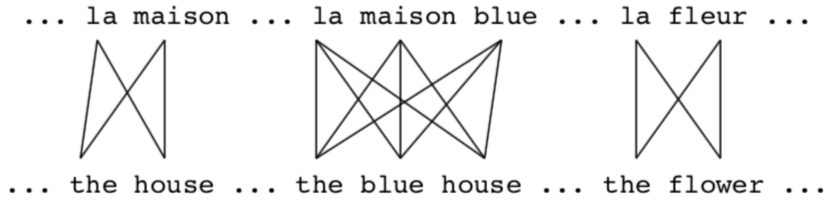
EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:
  - look at the entire dataset and sum the (expected) alignments we saw for each word and it’s possible translations (E-step)



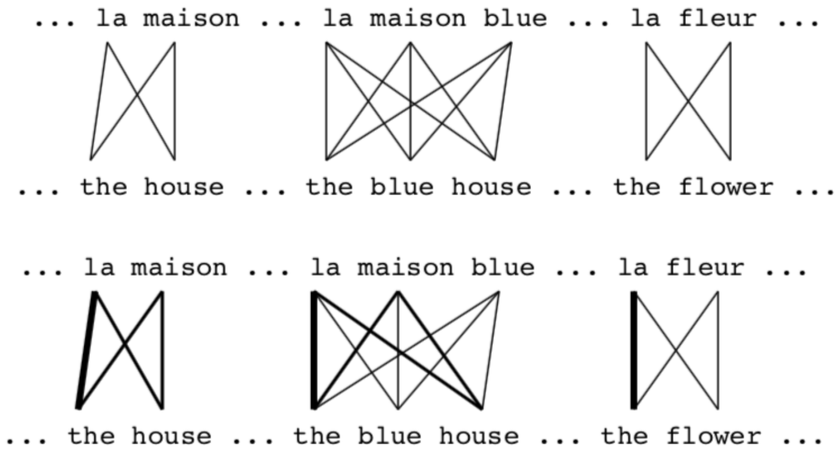
EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:
  - look at the entire dataset and sum the (expected) alignments we saw for each word and it’s possible translations (E-step)
  - Update the translation probabilities according to those global counts (M-step)...



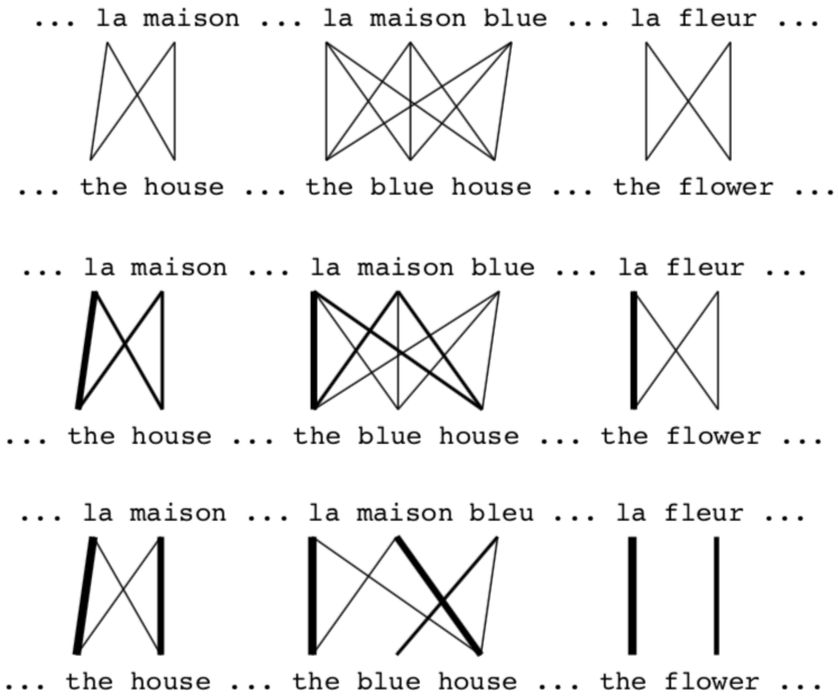
EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:
  - look at the entire dataset and sum the (expected) alignments we saw for each word and it's possible translations (E-step)
  - Update the translation probabilities according to those global counts (M-step)...
  - ...which will update the alignment counts



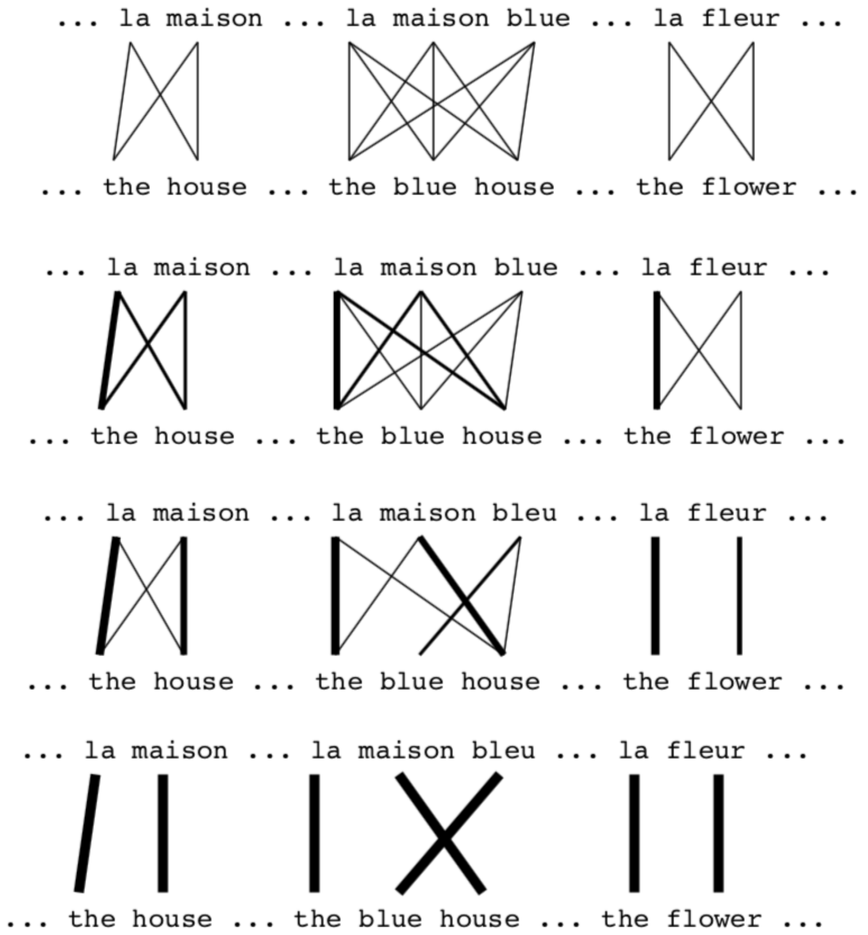
EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:
  - look at the entire dataset and sum the (expected) alignments we saw for each word and it’s possible translations (E-step)
  - Update the translation probabilities according to those global counts (M-step)...
  - ...which will update the alignment counts
- Repeat steps above



EM for IBM Model1 - Overview

- Start with all alignments equally likely
- In each iteration:
  - look at the entire dataset and sum the (expected) alignments we saw for each word and it's possible translations (E-step)
  - Update the translation probabilities according to those global counts (M-step)...
  - ...which will update the alignment counts
- Repeat steps above
- Until convergence



EM for IBM Model1 - Computation

EM for IBM Model1 - Computation

- We need to compute:

EM for IBM Model1 - Computation

- We need to compute:
  - Expectation step: probability of alignments

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

EM for IBM Model1 - Computation

- We need to compute:
  - Expectation step: probability of alignments

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

  - Maximization step: probability of word translations

$$t(e|f;\mathbf{e},\mathbf{f})$$

EM for IBM Model 1 - Expectation Step

EM for IBM Model 1 - Expectation Step

- Apply the chain rule

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

EM for IBM Model 1 - Expectation Step

- Apply the chain rule
- Numerator - IBM Model 1 definition

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

EM for IBM Model 1 - Expectation Step

- Apply the chain rule
- Numerator - IBM Model 1 definition
- Denominator:

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

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

$$= \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e},a|\mathbf{f})$$

$$= \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

EM for IBM Model 1 - Expectation Step

- Apply the chain rule
- Numerator - IBM Model 1 definition
- Denominator:
  - Marginalize over all possible alignments

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

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

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e},a|\mathbf{f})$$

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

EM for IBM Model 1 - Expectation Step

- Apply the chain rule
- Numerator - IBM Model 1 definition
- Denominator:
  - Marginalize over all possible alignments
  - IBM model 1 definition

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

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

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e},a|\mathbf{f})$$

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

EM for IBM Model 1 - Expectation Step

$$p(\mathbf{e}|\mathbf{f}) = \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)$$

EM for IBM Model 1 - Expectation Step

- We want to get rid of the exponential number of multiplications

$$p(\mathbf{e}|\mathbf{f}) = \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)$$

EM for IBM Model 1 - Expectation Step

- We want to get rid of the exponential number of multiplications
- Move the constants out

$$p(\mathbf{e}|\mathbf{f}) = \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)$$

EM for IBM Model 1 - Expectation Step

- We want to get rid of the exponential number of multiplications
- Move the constants out
- Last trick - change sum of products to product of sums

$$p(\mathbf{e}|\mathbf{f}) = \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)$$

EM for IBM Model 1 - Expectation Step

EM for IBM Model 1 - Expectation Step

- So finally we got:

EM for IBM Model 1 - Expectation Step

- So finally we got:

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

$$= \frac{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)}$$

$$= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

EM for IBM Model 1 - Expectation Step

- So finally we got:
  - Probability of alignment given a sentence pair

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

$$= \frac{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)}$$

$$= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

EM for IBM Model 1 - Expectation Step

- So finally we got:
  - Probability of alignment given a sentence pair
  - Based on the translation probabilities alone

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

$$= \frac{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)}$$

$$= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

EM for IBM Model 1 - Expectation Step

- So finally we got:
  - Probability of alignment given a sentence pair
  - Based on the translation probabilities alone
  - We will use this to get the expected alignment counts

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

$$= \frac{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)}$$

$$= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

EM for IBM Model 1 - Maximization Step

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation

$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_{a} p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation

$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_{a} p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{i=0}^{l_f} \delta(f,f_i)$$

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation

$$p(\text{the}|\text{la}) = 0.7 \qquad p(\text{house}|\text{la}) = 0.05$$
$$p(\text{the}|\text{maison}) = 0.1 \qquad p(\text{house}|\text{maison}) = 0.8$$

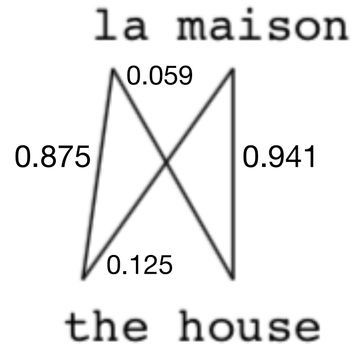
$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_a p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{i=0}^{l_f} \delta(f,f_i)$$

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation

$$p(\text{the}|\text{la}) = 0.7 \quad p(\text{house}|\text{la}) = 0.05$$
$$p(\text{the}|\text{maison}) = 0.1 \quad p(\text{house}|\text{maison}) = 0.8$$



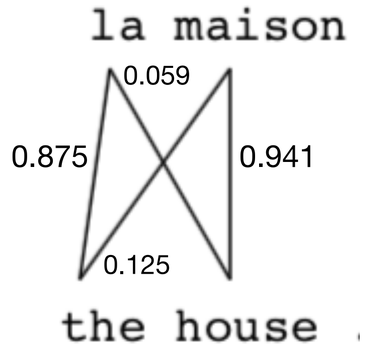
$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_a p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{i=0}^{l_f} \delta(f,f_i)$$

EM for IBM Model 1 - Maximization Step

- We want to collect alignment counts to compute the new translation parameters, using the **current** translation parameters
- For each possible pair (e,f) in each example (**e,f**) sum the expected counts of this translation
- Maximization: after we do this over the entire corpus, sum and normalize to get the **new** parameters

$p(\text{the}|\text{la}) = 0.7$ $p(\text{house}|\text{la}) = 0.05$
$p(\text{the}|\text{maison}) = 0.1$ $p(\text{house}|\text{maison}) = 0.8$



$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_a p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e,e_j) \sum_{i=0}^{l_f} \delta(f,f_i)$$

$$t(e|f;\mathbf{e},\mathbf{f}) = \frac{\sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}{\sum_f \sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}$$

EM for IBM Model 1 - Pseudo Code

EM for IBM Model 1 - Pseudo Code

- Finally:

EM for IBM Model 1 - Pseudo Code

- Finally:

Input: set of sentence pairs (**e**, **f**)
Output: translation prob. $t(e|f)$

initialize $t(e|f)$ uniformly
while not converged **do**
  // *initialize*
  count$(e|f) = 0$ **for all** e, f
  total$(f) = 0$ **for all** f
  for all sentence pairs (**e**,**f**) **do**
    // *compute normalization*
    for all words e in **e do**
      s-total$(e) = 0$
      for all words f in **f do**
        s-total(e) += $t(e|f)$
      end for
    end for
    // *collect counts*
    for all words e in **e do**
      for all words f in **f do**
        count$(e|f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
        total(f) += $\frac{t(e|f)}{\text{s-total}(e)}$
      end for
    end for
  end for
  // *estimate probabilities*
  for all foreign words f **do**
    for all English words e **do**
      $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
    end for
  end for
end while

EM for IBM Model 1 - Pseudo Code

- Finally:
- You will implement this in Exercise 1

Input: set of sentence pairs $(\mathbf{e}, \mathbf{f})$
Output: translation prob. $t(e|f)$

initialize $t(e|f)$ uniformly
while not converged **do**
  // initialize
  $\text{count}(e|f) = 0$ **for all** e, f
  $\text{total}(f) = 0$ **for all** f
  for all sentence pairs (**e**,**f**) **do**
    // compute normalization
    for all words e in **e** **do**
      $\text{s-total}(e) = 0$
      for all words f in **f** **do**
        $\text{s-total}(e)$ += $t(e|f)$
      end for
    end for
    // collect counts
    for all words e in **e** **do**
      for all words f in **f** **do**
        $\text{count}(e|f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
        $\text{total}(f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
      end for
    end for
  end for
  // estimate probabilities
  for all foreign words f **do**
    for all English words e **do**
      $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
    end for
  end for
end while

Alignment Error Rate (AER)

Alignment Error Rate (AER)

- How can we measure the alignment quality?

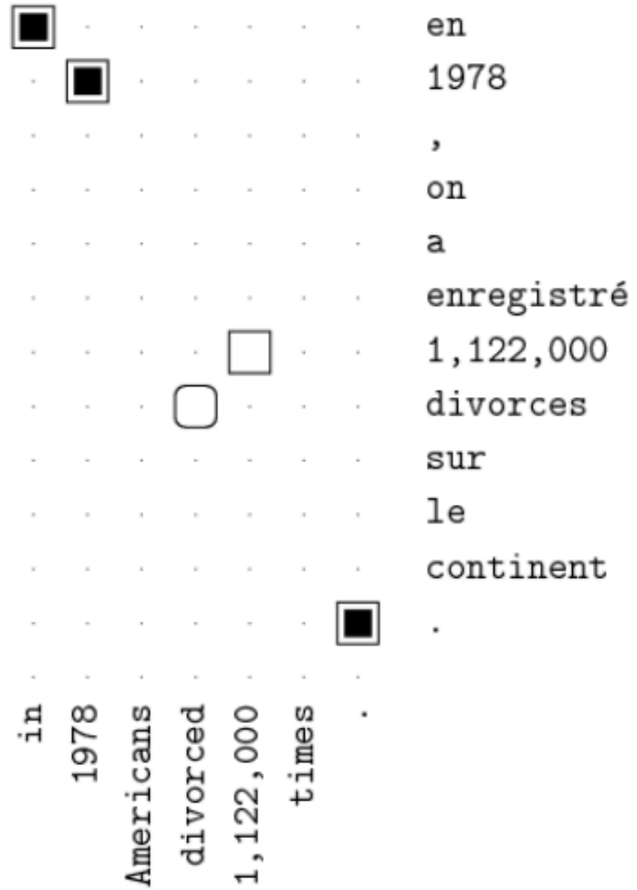
Alignment Error Rate (AER)

- How can we measure the alignment quality?
- AER - Och and Ney, 2000

= Sure

= Possible

= Predicted

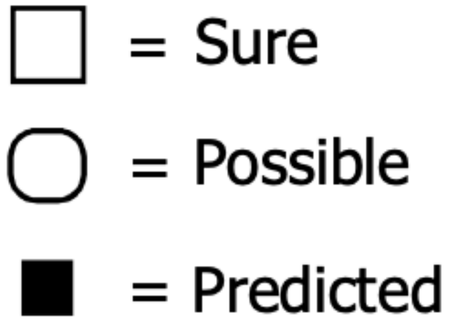


$$AER(A,S,P) = \left(1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}\right)$$

$$= \left(1 - \frac{3+3}{3+4}\right) = \frac{1}{7}$$

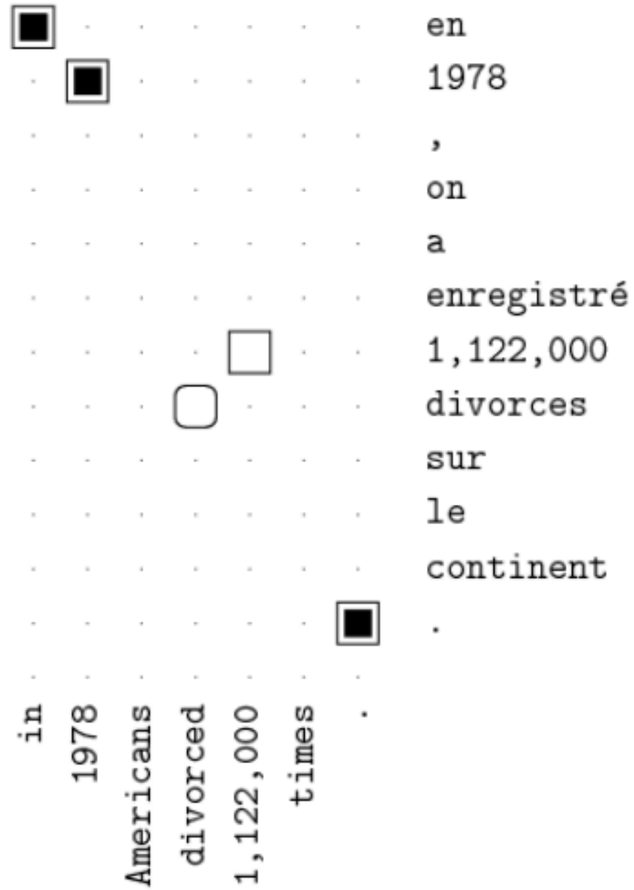
Alignment Error Rate (AER)

- How can we measure the alignment quality?
- AER - Och and Ney, 2000
- Possible contains Sure



$$AER(A,S,P) = \left(1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}\right)$$

$$= \left(1 - \frac{3+3}{3+4}\right) = \frac{1}{7}$$



Alignment Error Rate (AER)

- How can we measure the alignment quality?
- AER - Och and Ney, 2000
- Possible contains Sure
- Must hit all Sure to be perfect, ok to not cover all probable

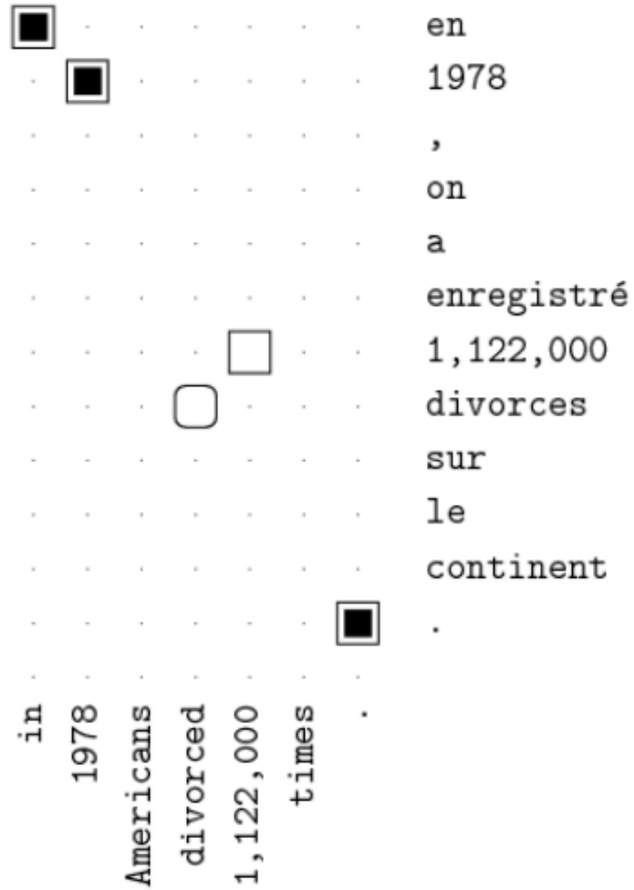
= Sure

= Possible

= Predicted

$$AER(A,S,P) = \left(1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|}\right)$$

$$= \left(1 - \frac{3+3}{3+4}\right) = \frac{1}{7}$$



Summary

Summary

- IBM model 1

Summary

- IBM model 1
  - Generative model, based on:

Summary

- IBM model 1
  - Generative model, based on:
  - Alignments (latent variables)

Summary

- IBM model 1
  - Generative model, based on:
    - nearly parallel order of morphemes and concepts
  - Alignments (latent variables)
  - Which are used to calculate translation parameters

Summary

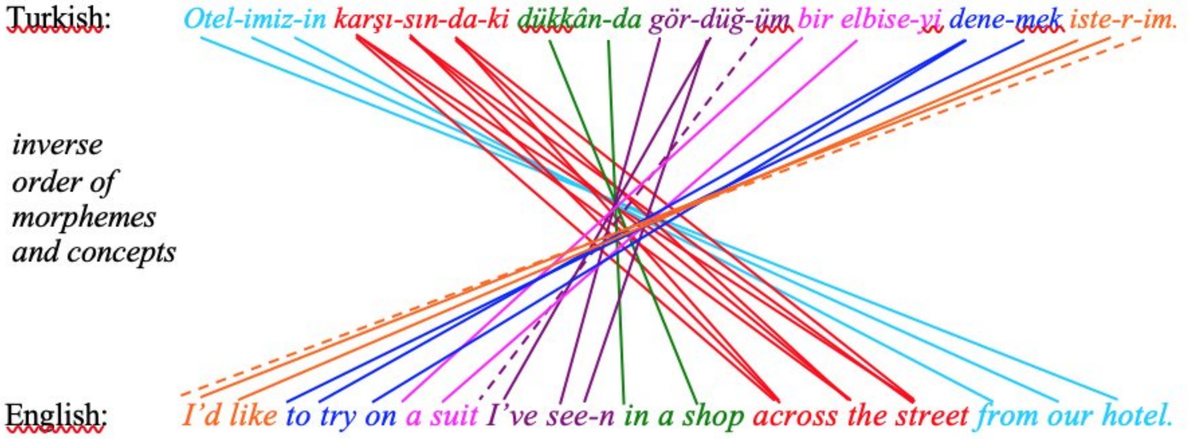
- IBM model 1
  - Generative model, based on:
  - Alignments (latent variables)
  - Which are used to calculate translation parameters
  - Using Expectation Maximization

Summary

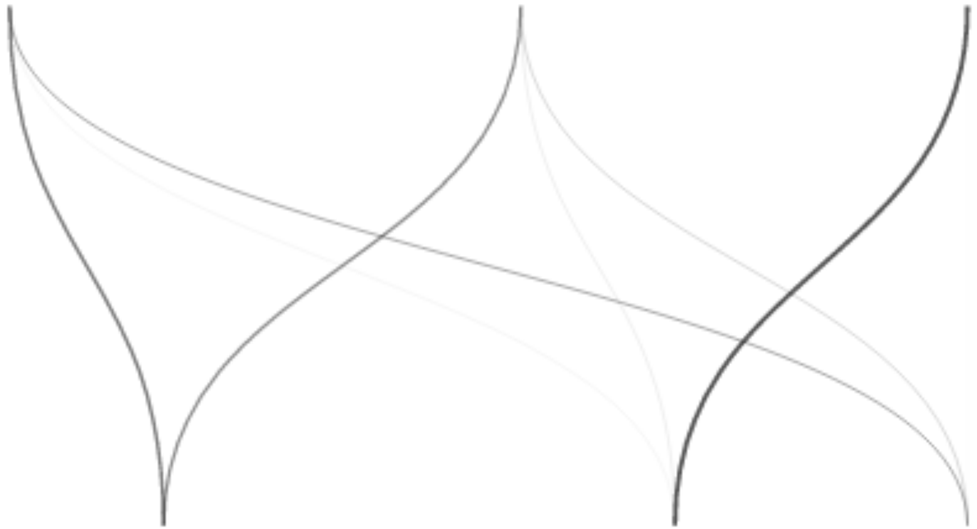
- IBM model 1
  - Generative model, based on:
  - Alignments (latent variables)
  - Which are used to calculate translation parameters
  - Using Expectation Maximization
- Exercise 1 - May 18th

Summary

- IBM model 1
  - Generative model, based on:
  - Alignments (latent variables)
  - Which are used to calculate translation parameters
  - Using Expectation Maximization
- Exercise 1 - May 18th



Any Questions ?



Questions diverses ?