machine translation I:
IBM Model 1 and EM

CS 585, Fall 2018
Introduction to Natural Language Processing
http://people.cs.umass.edu/~miyyer/cs585/

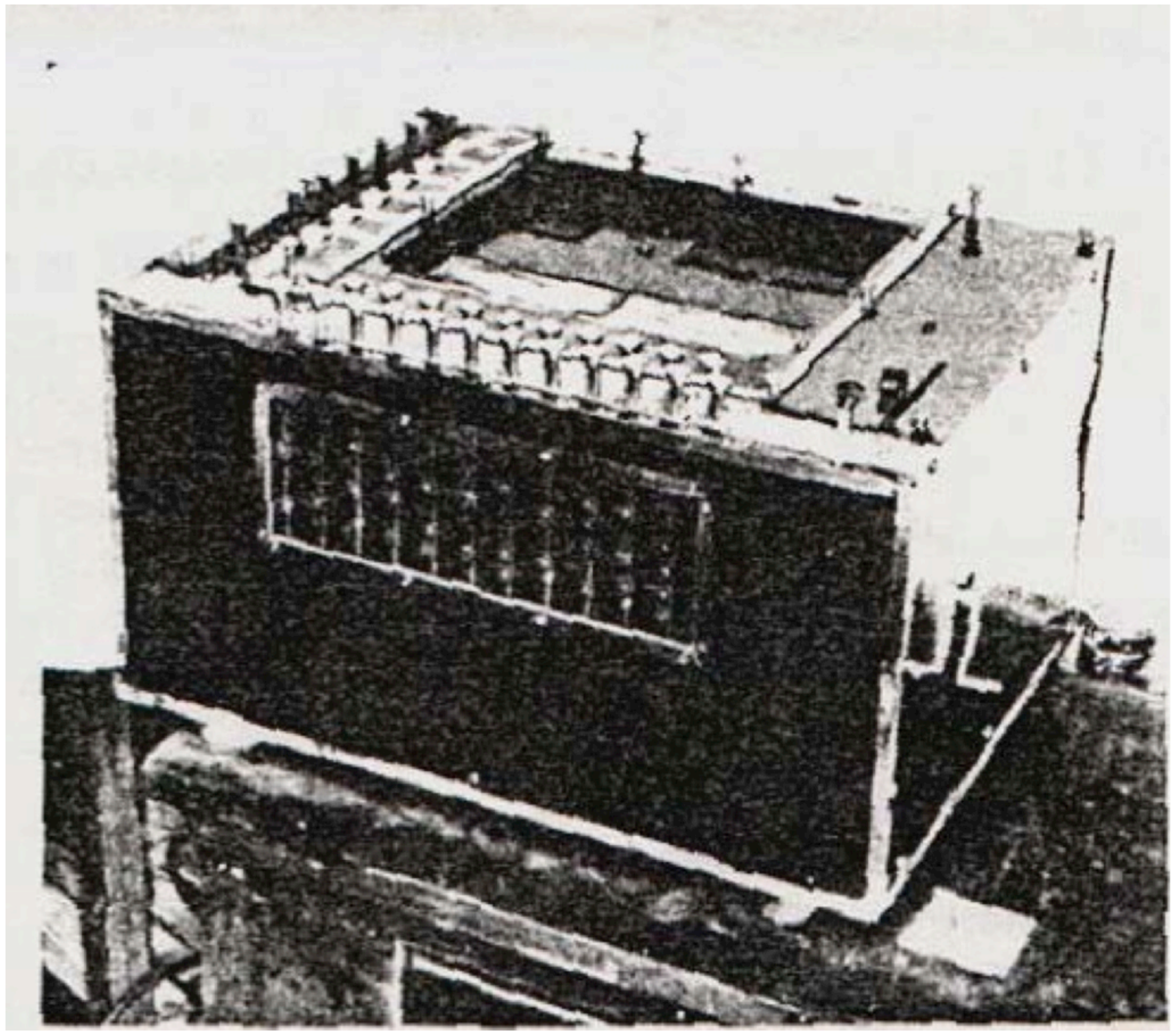
Mohit Iyyer
College of Information and Computer Sciences
University of Massachusetts Amherst

slides adapted from Brendan O'Connor, Philip Koehn, and Michael Collins

questions from last time…

- my office hours tmrw rescheduled to Monday 2-3PM
- i need a final project group / we need one more person for our group / etc.
  - use "search for teammates" post on Piazza!
- What was that google GPU resource?
  - https://colab.research.google.com
- Is implementing an existing model for project okay?
  - yes, as a replication study or comparison between multiple models (*and you must implement it yourself*)
- can i share NN and NLP course projects?
  - ok w/ me, but check with NN instructor

Machine translation

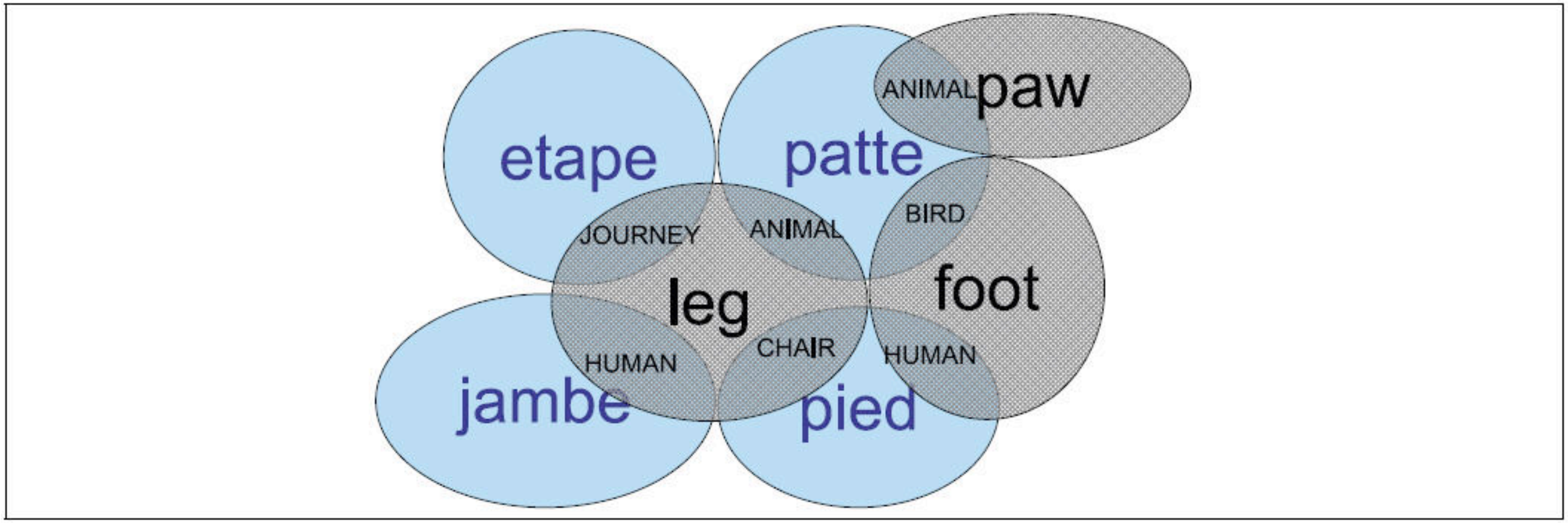


Georges Artrouni's "mechanical brain", a translation device patented in France in 1933. (Image from Corbé by way of John Hutchins)

The memory was the core of the device. It consisted of a paper band 40 cm wide, which could be up to 40 meters in length, moving over two rolling drums and held in position by perforations on the edges. The dictionary entries were recorded in normal orthographic form (i.e. not coded) line by line in five columns. The first column was for the source language word (or term), the other columns for equivalents in other languages and for other useful information.

MT is hard

- Word meaning:
  many-to-many and context dependent



- *Translation* itself is hard: metaphors, cultural references, etc.

MT goals

- Motivation: Human translation is expensive
- Rough translation vs. none
- Interactive assistance for human translators
  - e.g. Lilt
    - https://www.youtube.com/watch?v=YZ7G3gQgpfI
    - https://lilt.com/app/projects/details/1887/edit-document/2306
  - [compare to bilingual dictionary]

MT paradigms

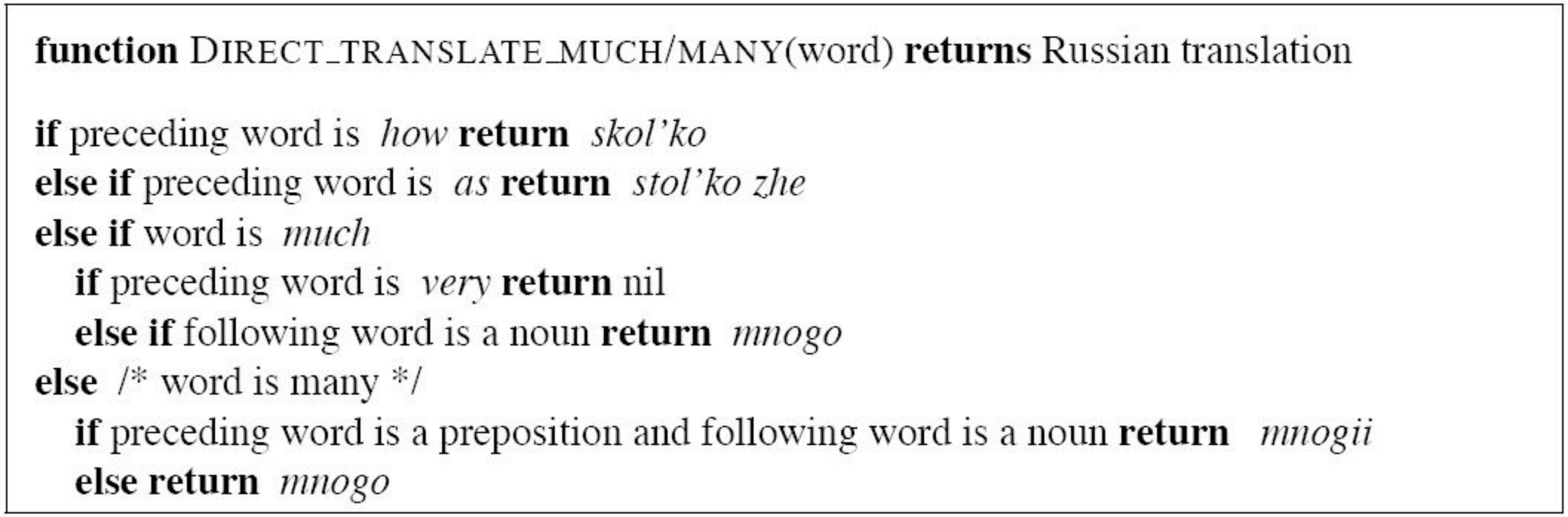
- Rule-based *transfer rules*
  - Manually program lexicons/rules
  - SYSTRAN (AltaVista Babelfish; originally from 70s)

- Statistical MT
  - Word-to-word, phrase-to-phrase probs
  - Learn phrase- or syntax-tree translation rules from data, search for high-scoring translation outputs
  - Key research in the early 90s
  - Google Translate (mid 00s)
  - Open-source: Moses

- Neural MT next lecture!
  - Research in early 10s; very recently deployed
  - Latent representations of words/phrases

Rules are hard

- Coverage
- Complexity (context dependence)
- Maintenance



```
function DIRECT_TRANSLATE_MUCH/MANY(word) returns Russian translation

if preceding word is how return skol'ko
else if preceding word is as return stol'ko zhe
else if word is much
   if preceding word is very return nil
   else if following word is a noun return mnogo
else /* word is many */
   if preceding word is a preposition and following word is a noun return mnogii
   else return mnogo
```

Interlingua

“Mary did not slap the green witch”

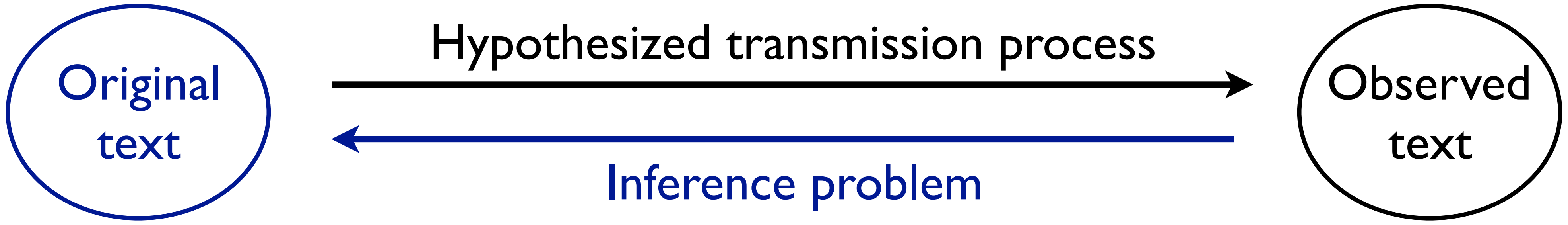
$$\begin{bmatrix} \text{EVENT} & \text{SLAPPING} \\ \text{AGENT} & \text{MARY} \\ \text{TENSE} & \text{PAST} \\ \text{POLARITY} & \text{NEGATIVE} \\ \text{THEME} & \begin{bmatrix} \text{WITCH} & \\ \text{DEFINITENESS} & \text{DEF} \\ \text{ATTRIBUTES} & \begin{bmatrix} \text{HAS-COLOR} & \text{GREEN} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

- More like classic logic-based AI
- Works in narrow domains
- Broad domain currently fails
  - Coverage: Knowledge representation for all possible semantics?
  - Can you parse to it?
  - Can you generate from it?

Machine learning for MT

- MT as ML: Translation is something people do naturally. Learn rules from data?
- Parallel data: (source, target) text pairs
  - E.g. 20 million words of European Parliament proceedings http://www.statmt.org/europarl/
- Training: learn parameters to predict {source => target}
- Test time: given source sentence, search for high-scoring target (e.g. beam search)

Noisy channel model



One naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography. When I look at an article in Russian, I say: 'This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.'

-- Warren Weaver (1955)

Machine translation

$$P(\text{target text} \mid \text{source text}) \propto P(\text{source text} \mid \text{target text})\, P(\text{target text})$$

Example from Koehn & Knight

Translation from Spanish to English, candidate translations based on $p(Spanish \mid English)$ alone:

Que hambre tengo yo
$\rightarrow$

What hunger have	$p(s\|e) = 0.000014$
Hungry I am so	$p(s\|e) = 0.000001$
I am so hungry	$p(s\|e) = 0.0000015$
Have i that hunger	$p(s\|e) = 0.000020$

$\ldots$

Example from Koehn & Knight

With $p(Spanish \mid English) \times p(English)$:

Que hambre tengo yo

$\rightarrow$

What hunger have	$p(s\|e)p(e) = 0.000014 \times 0.000001$
Hungry I am so	$p(s\|e)p(e) = 0.000001 \times 0.0000014$
I am so hungry	$p(s\|e)p(e) = 0.0000015 \times 0.0001$
Have i that hunger	$p(s\|e)p(e) = 0.000020 \times 0.00000098$

. . .

Example: learning with parallel data

1a. Garcia and associates.
1b. Garcia y asociados.

2a. Carlos Garcia has three associates.
2b. Carlos Garcia tiene tres asociados.

3a. his associates are not strong.
3b. sus asociados no son fuertes.

4a. Garcia has a company also.
4b. Garcia tambien tiene una empresa.

5a. its clients are angry.
5b. sus clientes están enfadados.

6a. the associates are also angry.
6b. los asociados tambien están enfadados.

7a. the clients and the associates are enemies.
7b. los clientes y los asociados son enemigos.

8a. the company has three groups.
8b. la empresa tiene tres grupos.

9a. its groups are in Europe.
9b. sus grupos están en Europa.

10a. the modern groups sell strong pharmaceuticals.
10b. los grupos modernos venden medicinas fuertes.

11a. the groups do not sell zanzanine.
11b. los grupos no venden zanzanina.

12a. the small groups are not modern.
12b. los grupos pequeños no son modernos.

this lecture: lexical translation

- we translate a single word by… looking it up in a dictionary (basically)!
    - Haus ==> house, building, home, etc.
- Multiple translations possible! some are more frequent than others
    - house & building most common for Haus
    - some special cases: Haus of snail is its shell
- For our lectures, we always will translate from a foreign language to English

Recap: The Noisy Channel Model

- Goal: translation system from French to English
- Have a model $p(e \mid f)$ which estimates conditional probability of any English sentence e given the French sentence f. Use the training corpus to set the parameters.
- A Noisy Channel Model has two components:

$p(e)$	**the language model**
$p(f \mid e)$	**the translation model**

- Giving:

$$p(e \mid f) = \frac{p(e, f)}{p(f)} = \frac{p(e)p(f \mid e)}{\sum_e p(e)p(f \mid e)}$$

and

$$\text{argmax}_e p(e \mid f) = \text{argmax}_e p(e)p(f \mid e)$$

rest of lecture will focus on the translation model!

Collect Statistics

Look at a parallel corpus (German text along with English translation)

Translation of Haus	**Count**
house	8,000
building	1,600
home	200
household	150
shell	50

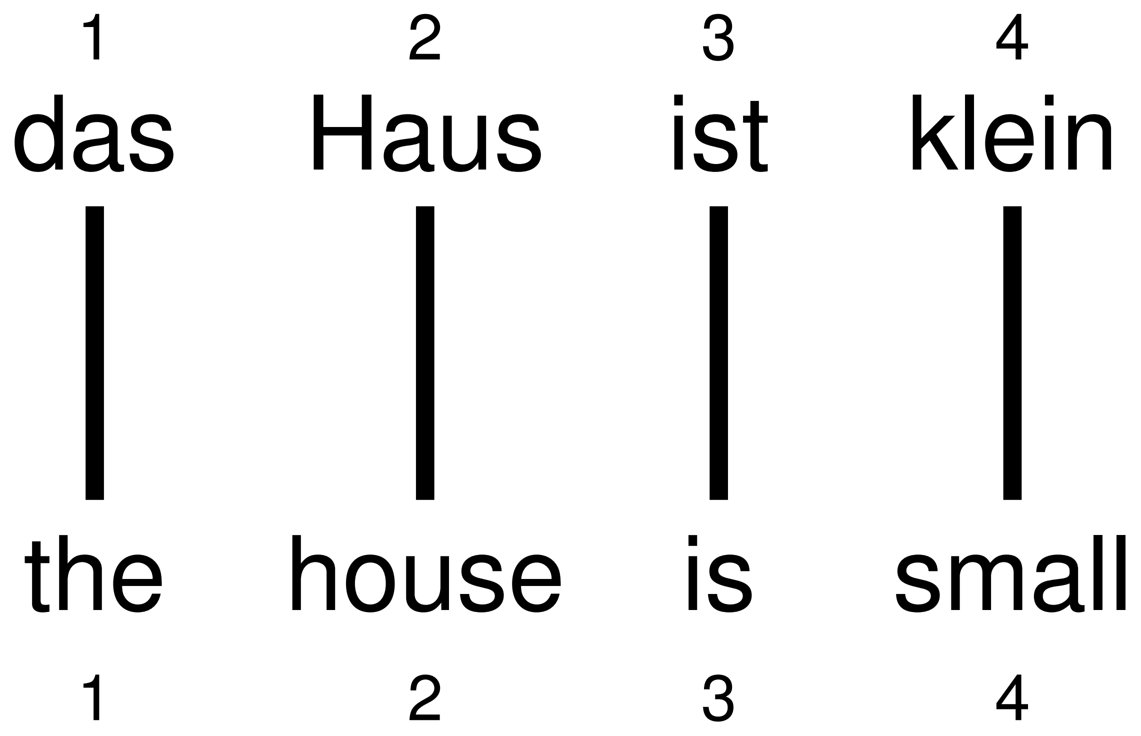
Estimate Translation Probabilities

Maximum likelihood estimation

$$p_f(e) = \begin{cases} 0.8 & \text{if } e = \text{house}, \\ 0.16 & \text{if } e = \text{building}, \\ 0.02 & \text{if } e = \text{home}, \\ 0.015 & \text{if } e = \text{household}, \\ 0.005 & \text{if } e = \text{shell}. \end{cases}$$

Alignment

- In a parallel text (or when we translate), we align words in one language with the words in the other



- Word positions are numbered 1–4

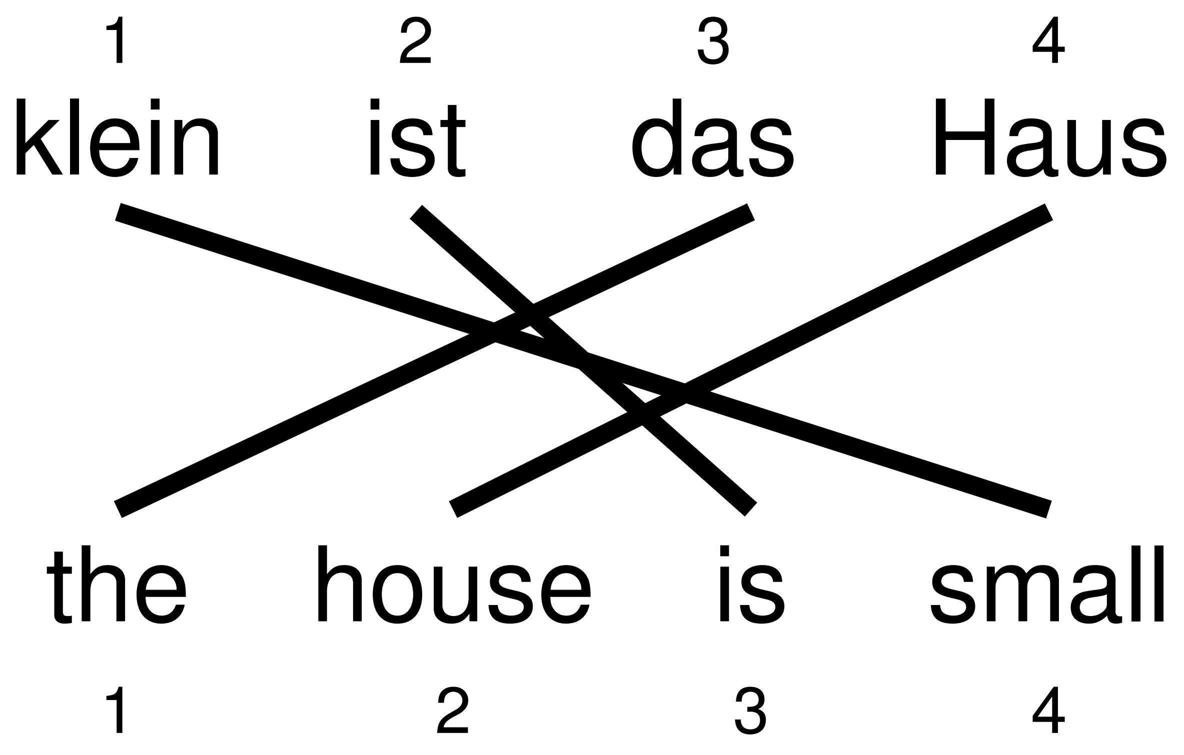
Alignment Function

- Formalizing alignment with an alignment function
- Mapping an English target word at position i to a German source word at position j with a function $a : i \rightarrow j$
- Example

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

Reordering

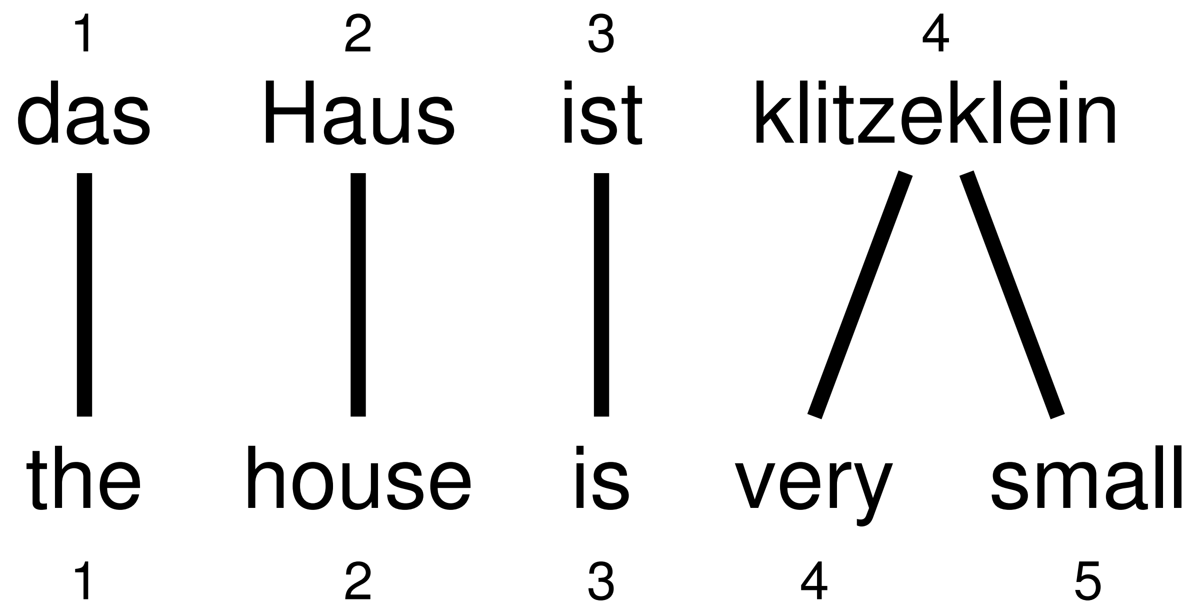
Words may be reordered during translation



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

One-to-Many Translation

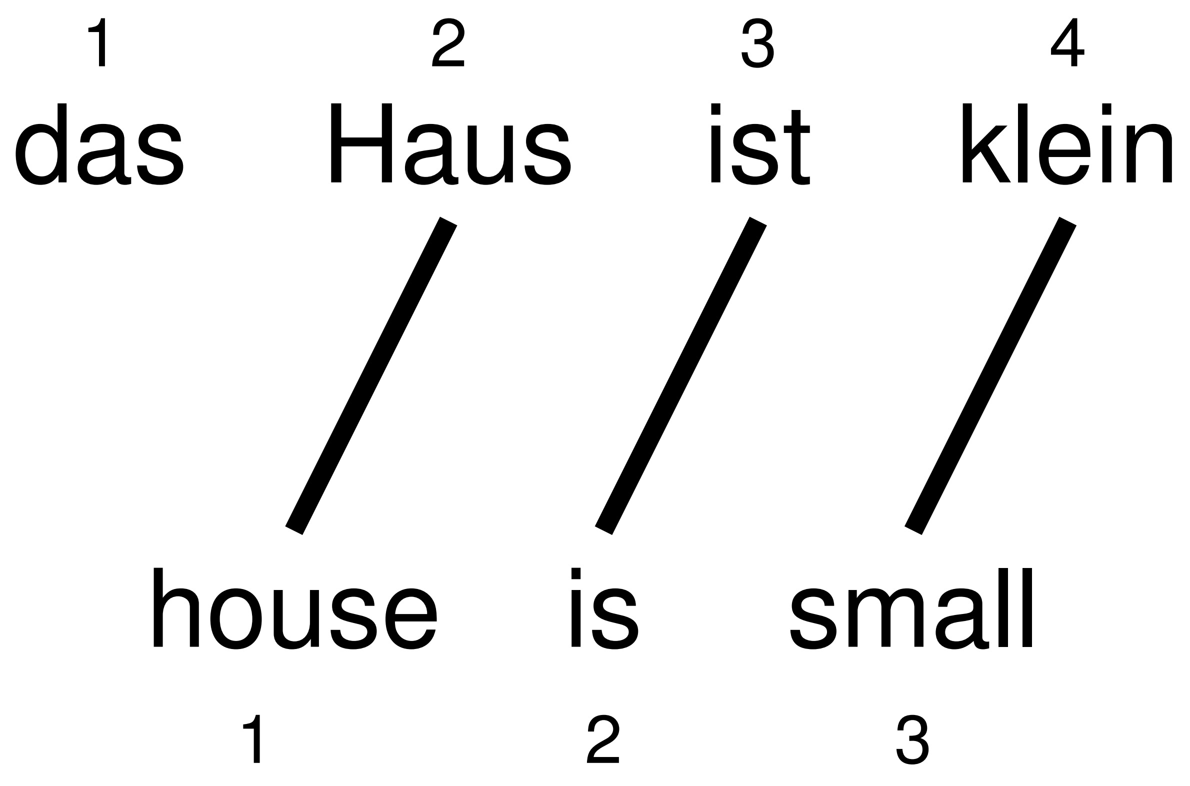
A source word may translate into multiple target words



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

Dropping Words

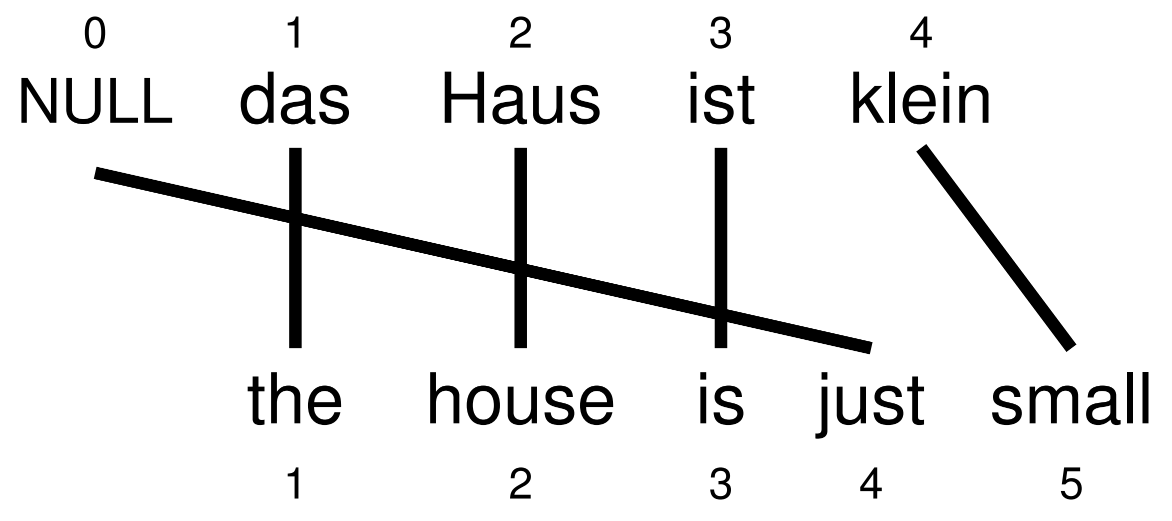
Words may be dropped when translated
(German article das is dropped)



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

Inserting Words

- Words may be added during translation
  - The English just does not have an equivalent in German
  - We still need to map it to something: special NULL token



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

A family of lexical translation models

- A family translation models
- Uncreatively named: Model 1, Model 2, . . .
- Foundation of all modern translation algorithms
- First up: Model 1

IBM Model 1: Alignments

- How do we model $p(f \mid e)$?

translation model
in noisy channel

- English sentence e has l words $e_1 \ldots e_l$,
  French sentence f has m words $f_1 \ldots f_m$.

- An alignment a identifies which English word each French word originated from

- Formally, an alignment a is $\{a_1, \ldots a_m\}$, where each $a_j \in \{0 \ldots l\}$.

how many possible alignments are there?

IBM Model 1: Alignments

- How do we model $p(f \mid e)$?

translation model in noisy channel

- English sentence e has l words $e_1 \dots e_l$,
  French sentence f has m words $f_1 \dots f_m$.
- An alignment a identifies which English word each French word originated from
- Formally, an alignment a is $\{a_1, \dots a_m\}$, where each $a_j \in \{0 \dots l\}$.
- There are $(l+1)^m$ possible alignments.

Alignments in the IBM Models

- We'll define models for $p(a \mid e, m)$ and $p(f \mid a, e, m)$, giving

$$p(f, a \mid e, m) = p(a \mid e, m)p(f \mid a, e, m)$$

chain rule

- Also,

$$p(f \mid e, m) = \sum_{a \in \mathcal{A}} p(a \mid e, m)p(f \mid a, e, m)$$

where $\mathcal{A}$ is the set of all possible alignments

here we *marginalize out* the alignments!

A By-Product: Most Likely Alignments

- Once we have a model $p(f, a \mid e, m) = p(a \mid e)p(f \mid a, e, m)$ we can also calculate

$$p(a \mid f, e, m) = \frac{p(f, a \mid e, m)}{\sum_{a \in \mathcal{A}} p(f, a \mid e, m)}$$

  for any alignment a this will be very useful for EM!

- For a given f, e pair, we can also compute the most likely alignment,

$$a^* = \arg\max_a p(a \mid f, e, m)$$

- Nowadays, the original IBM models are rarely (if ever) used for translation, but they are used for recovering alignments

IBM Model 1: Alignments

$$p(f, a \mid e, m) = \boxed{p(a \mid e, m)} p(f \mid a, e, m)$$

- In IBM model 1 all allignments a are equally likely:

$$p(a \mid e, m) = \frac{1}{(l+1)^m}$$

is this a reasonable assumption?

IBM Model 1: Translation Probabilities

$$p(f, a \mid e, m) = p(a \mid e, m)\boxed{p(f \mid a, e, m)}$$

- Next step: come up with an estimate for

$$p(f \mid a, e, m)$$

- In model 1, this is:

here, f_j is the french word aligned to e_{a_j}

$$p(f \mid a, e, m) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

what is the independence assumption here?

IBM Model 1: Translation Probabilities

$$p(f, a \mid e, m) = p(a \mid e, m)\boxed{p(f \mid a, e, m)}$$

- Next step: come up with an estimate for

$$p(f \mid a, e, m)$$

- In model 1, this is:

here, f_j is the french word aligned to e_{a_j}

$$p(f \mid a, e, m) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

what is the independence assumption here?

given the alignment, each translation decision is conditionally independent of all others and depends only on the aligned source word

IBM Model 1: The Generative Process

To generate a French string f from an English string e:

- **Step 1:** Pick an alignment a with probability $\frac{1}{(l+1)^m}$
- **Step 2:** Pick the French words with probability

$$p(f \mid a, e, m) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

The final result:

$$p(f, a \mid e, m) = p(a \mid e, m) \times p(f \mid a, e, m) = \frac{1}{(l+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

example

- e.g., $l = 6$, $m = 7$

$$e = \text{And the program has been implemented}$$

$$f = \text{Le programme a ete mis en application}$$

- $a = \{2, 3, 4, 5, 6, 6, 6\}$

$$\begin{aligned}
p(f \mid a, e) \quad = \quad & t(Le \mid the) \times \\
& t(programme \mid program) \times \\
& t(a \mid has) \times \\
& t(ete \mid been) \times \\
& t(mis \mid implemented) \times \\
& t(en \mid implemented) \times \\
& t(application \mid implemented)
\end{aligned}$$

IBM Model 2

- Only difference: we now introduce **alignment** or **distortion** parameters

$\mathbf{q}(i \mid j, l, m)$ = Probability that j'th French word is connected to i'th English word, given sentence lengths of e and f are l and m respectively

- Define

$$p(a \mid e, m) = \prod_{j=1}^{m} \mathbf{q}(a_j \mid j, l, m)$$

where $a = \{a_1, \ldots a_m\}$

- Gives

$$p(f, a \mid e, m) = \prod_{j=1}^{m} \mathbf{q}(a_j \mid j, l, m)\mathbf{t}(f_j \mid e_{a_j})$$

The Parameter Estimation Problem

- Input to the parameter estimation algorithm: $(e^{(k)}, f^{(k)})$ for $k = 1 \ldots n$. Each $e^{(k)}$ is an English sentence, each $f^{(k)}$ is a French sentence

need this only for model 2

- Output: parameters $t(f|e)$ and $q(i|j,l,m)$
- A key challenge: **we do not have alignments on our training examples**, e.g.,

$$\begin{aligned} e^{(100)} &= \text{And the program has been implemented} \\ f^{(100)} &= \text{Le programme a ete mis en application} \end{aligned}$$

chicken & egg problem!

- if we had the alignments, we could estimate the parameters of our model (i.e., the lexical translation probabilities)
- if we had the parameters, we could estimate the alignments.

- we have neither! :(

Parameter Estimation if the Alignments are Observed

- First: case where alignments are observed in training data. E.g.,

$$\begin{aligned} e^{(100)} &= \text{And the program has been implemented} \\ f^{(100)} &= \text{Le programme a ete mis en application} \\ a^{(100)} &= \langle 2,3,4,5,6,6,6 \rangle \end{aligned}$$

- Training data is $(e^{(k)}, f^{(k)}, a^{(k)})$ for $k = 1 \ldots n$. Each $e^{(k)}$ is an English sentence, each $f^{(k)}$ is a French sentence, each $a^{(k)}$ is an alignment
- Maximum-likelihood parameter estimates in this case are trivial:

$$t_{ML}(f|e) = \frac{\text{Count}(e,\ f)}{\text{Count}(e)}$$

EM algorithm

- Expectation maximization (EM) in a nutshell:
    1. initialize model parameters (trans. probs) using some method (e.g., uniform)
    2. assign probabilities to missing data (alignments)
    3. estimate model parameters from the completed data
    4. iterate steps 2-3 until convergence

J+M example:

- ignore NULL word
- ignore alignments where English word doesn’t align with any Spanish words
- compute simplified probability

$$p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

instead of

$$p(f, a \mid e, m) = p(a \mid e, m) \times p(f \mid a, e, m) = \frac{1}{(l+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

dataset:

green house	the house
casa verde	la casa

vocab:

{green, house, the}

{casa, la, verde}

dataset:

green house / casa verde

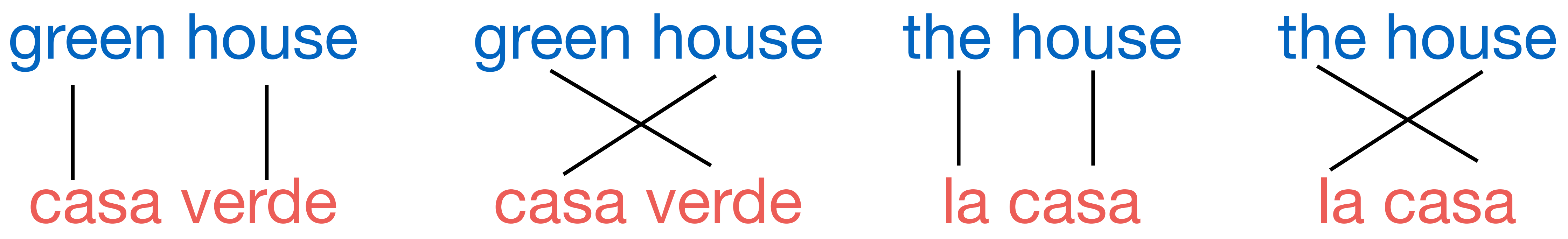
the house / la casa

initialize translation probabilities uniformly:

t(casa|green) = 1/3	t(verde|green) = 1/3	t(la|green) = 1/3
t(casa|house) = 1/3	t(verde|house) = 1/3	t(la|house) = 1/3
t(casa|the) = 1/3	t(verde|the) = 1/3	t(la|the) = 1/3

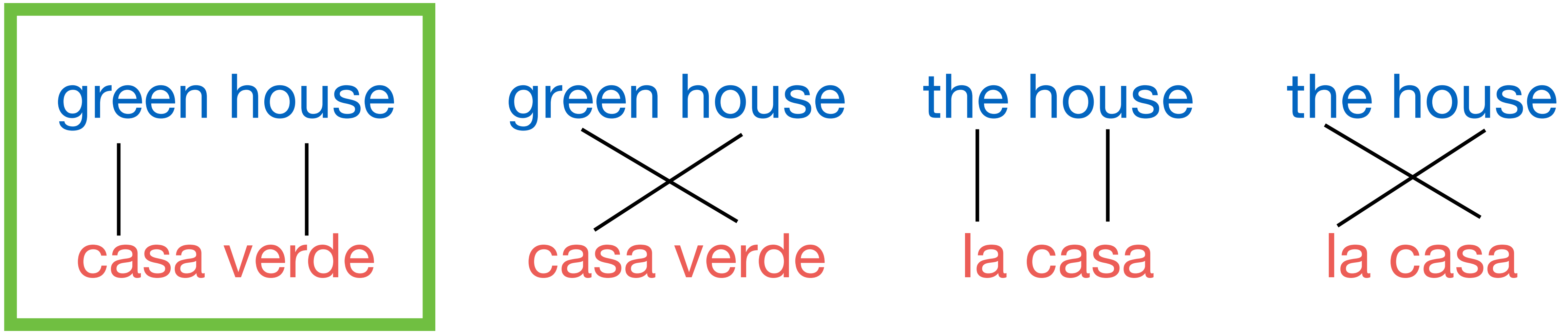
E-Step 1: compute expected counts E[*count(t(f,e))*]

first, for all alignments, let's compute $p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$



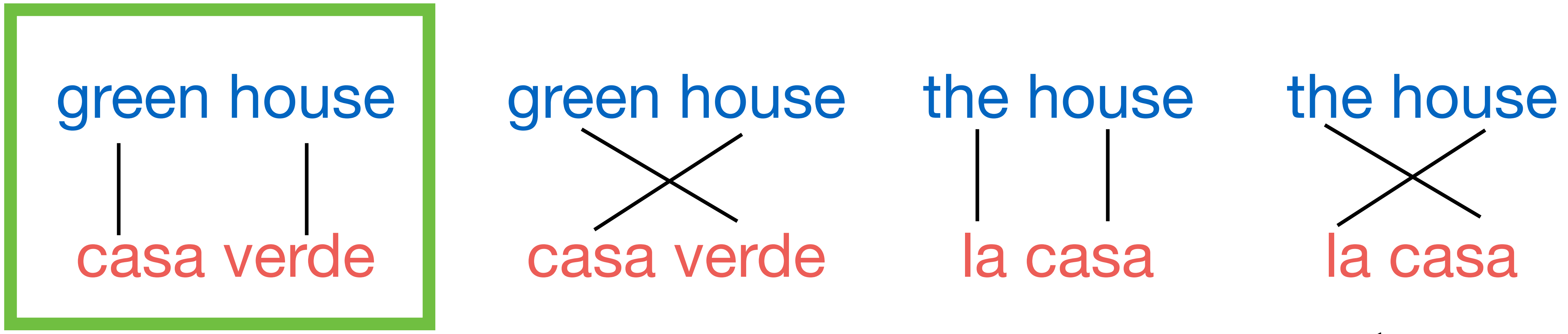
E-Step 1: compute expected counts E[*count(t(f,e)*]

first, for all alignments, let's compute $p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$



E-Step 1: compute expected counts E[*count(t(f,e)*)]

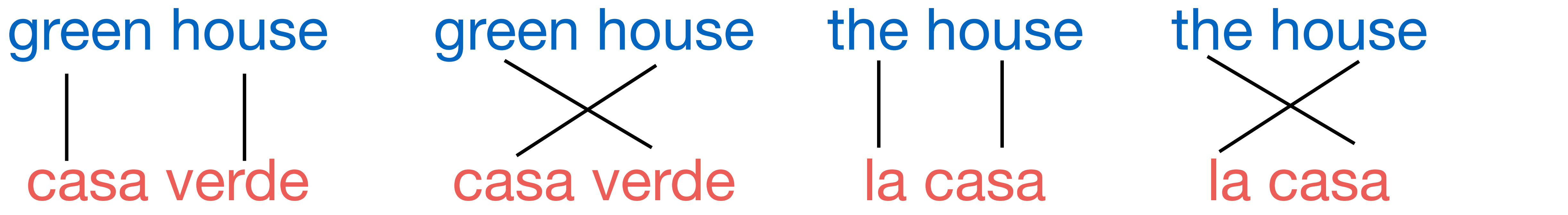
first, for all alignments, let's compute $p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$



$$p(f, a \mid e) = t(\text{casa} \mid \text{green}) \times t(\text{verde} \mid \text{house}) = \frac{1}{9}$$

E-Step 1: compute expected counts E[*count(t(f,e)*]

first, for all alignments, let's compute $p(f,a|e) = \prod_{j=1}^{m} t(f_j|e_{a_j})$

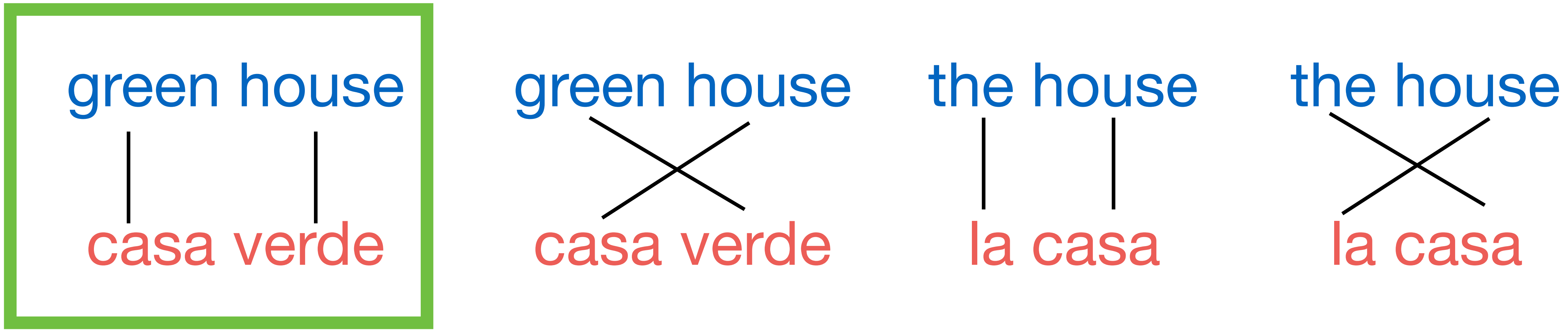


$p(f,a|e) = \frac{1}{9}$ $p(f,a|e) = \frac{1}{9}$ $p(f,a|e) = \frac{1}{9}$ $p(f,a|e) = \frac{1}{9}$

E-Step 1: compute expected counts E[*count(t(f,e)*]

next, let's compute *alignment probabilities* by normalizing:

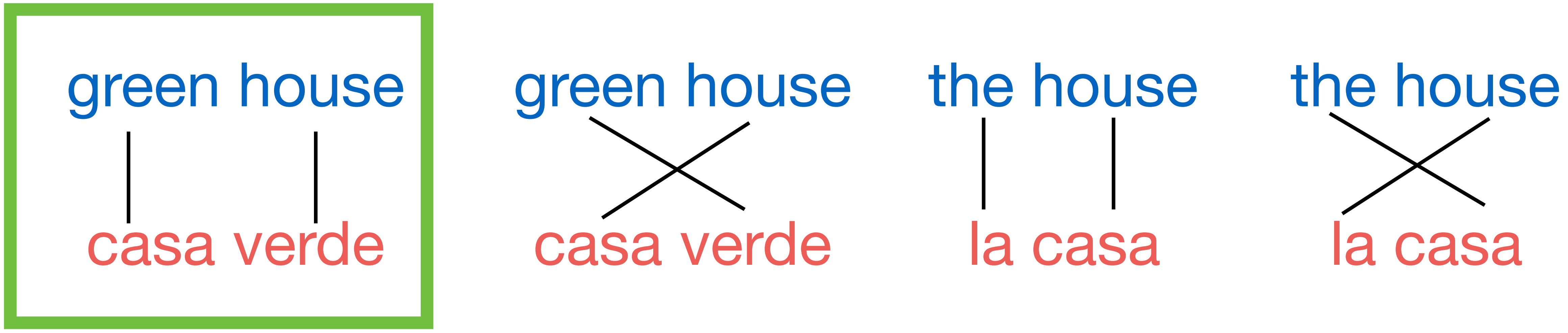
$$p(a\,|\,f,e) = \frac{p(a,f\,|\,e)}{\sum_a p(a,f\,|\,e)}$$



E-Step 1: compute expected counts E[*count(t(f,e)*]

next, let's compute *alignment probabilities* by normalizing:

$$p(a \mid f, e) = \frac{p(a, f \mid e)}{\sum_a p(a, f \mid e)}$$

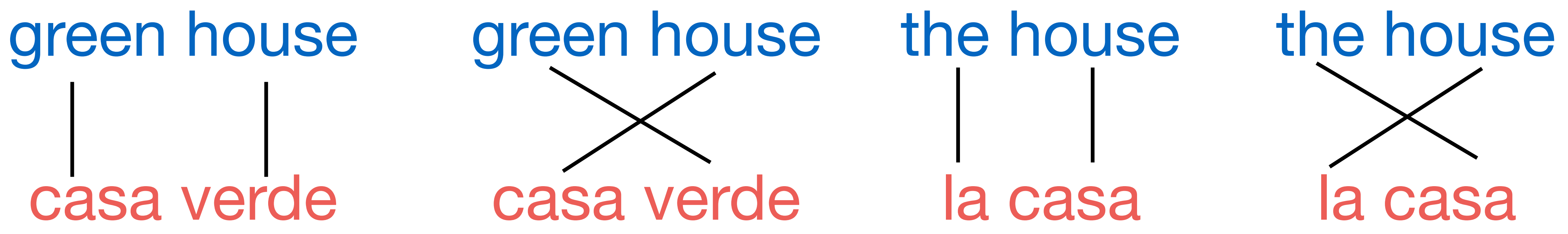


$$p(a \mid f, e) = \frac{\frac{1}{9}}{\frac{2}{9}} = \frac{1}{2}$$

E-Step 1: compute expected counts E[*count(t(f,e)*]

next, let’s compute *alignment probabilities* by normalizing:

$$p(a \mid f, e) = \frac{p(a, f \mid e)}{\sum_a p(a, f \mid e)}$$



$$p(a \mid f, e) = \frac{1}{2} \qquad p(a \mid f, e) = \frac{1}{2} \qquad p(a \mid f, e) = \frac{1}{2} \qquad p(a \mid f, e) = \frac{1}{2}$$

E-Step 1: compute expected counts E[*count(t(f,e)*]

now let's finally compute expected (fractional) counts for each (f,e) pair

t(casa|green) = ???	t(verde|green) =	t(la|green) =	total(green) =
t(casa|house) =	t(verde|house) =	t(la|house) =	total(house) =
t(casa|the) =	t(verde|the) =	t(la|the) =	total(the) =

E-Step 1: compute expected counts E[*count(t(f,e)*]

now let's finally compute expected (fractional) counts for each (f,e) pair

there is exactly one casa—green alignment with prob. 1/2

t(casa|green) = 1/2	t(verde|green) =	t(la|green) =	total(green) =
t(casa|house) =	t(verde|house) =	t(la|house) =	total(house) =
t(casa|the) =	t(verde|the) =	t(la|the) =	total(the) =

E-Step 1: compute expected counts E[*count(t(f,e)*]

now let's finally compute expected counts for each (f,e) pair

t(casa|green) = 1/2	t(verde|green) = 1/2	t(la|green) = 0	total(green) = 1
t(casa|house) =	t(verde|house) =	t(la|house) =	total(house) =
t(casa|the) =	t(verde|the) =	t(la|the) =	total(the) =

E-Step 1: compute expected counts E[*count(t(f,e)*)]

now let's finally compute expected counts for each (f,e) pair

there are two one casa—house alignments each with prob. 1/2

		(la|green) = 0	total(green) = 1
t(casa|house) = 1/2+1/2 =1	t(verde|house) =	t(la|house) =	total(house) =
t(casa|the) =	t(verde|the) =	t(la|the) =	total(the) =

E-Step 1: compute expected counts E[*count(t(f,e)*]

now let's finally compute expected counts for each (f,e) pair

t(casa|green) = 1/2	t(verde|green) = 1/2	t(la|green) = 0	total(green) = 1
t(casa|house) = 1/2+1/2 =1	t(verde|house) = 1/2	t(la|house) = 1/2	total(house) = 2
t(casa|the) = 1/2	t(verde|the) = 0	t(la|the) = 1/2	total(the) = 1

M-Step 1: compute MLE probs by normalizing

easy! just normalize each row to sum to 1

t(casa|green) = 1/2	t(verde|green) = 1/2	t(la|green) = 0
t(casa|house) = 1/2	t(verde|house) = 1/4	t(la|house) = 1/4
t(casa|the) = 1/2	t(verde|the) = 0	t(la|the) = 1/2

M-Step 1: compute MLE counts by normalizing

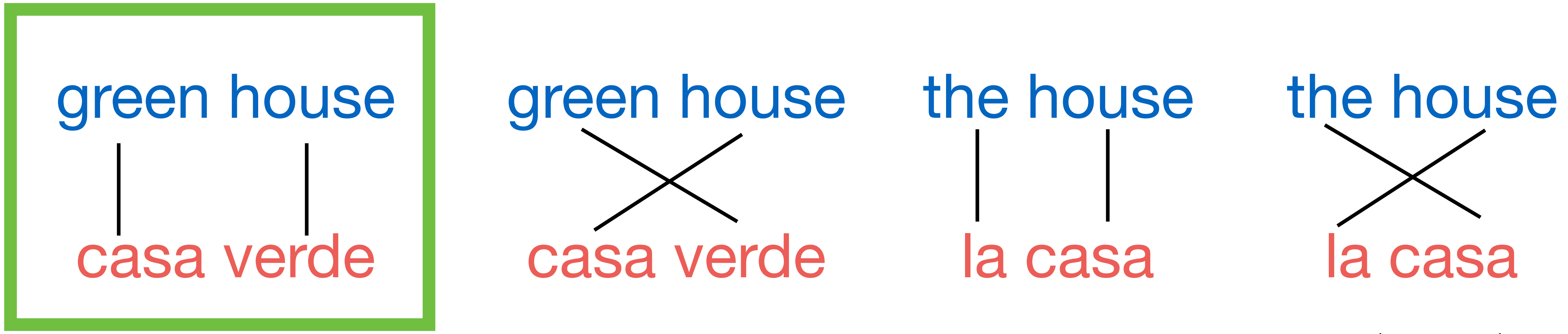
easy! just normalize each row to sum to 1

t(casa|green) = 1/2	t(verde|green) = 1/2	t(la|green) = 0
t(casa|house) = 1/2	t(verde|house) = 1/4	t(la|house) = 1/4
t(casa|the) = 1/2	t(verde|the) = 0	t(la|the) = 1/2

note that each of the correct translations have increased in probability! t(casa|house) is now 1/2 instead of 1/3

E-Step 2: repeat with new translation probabilities

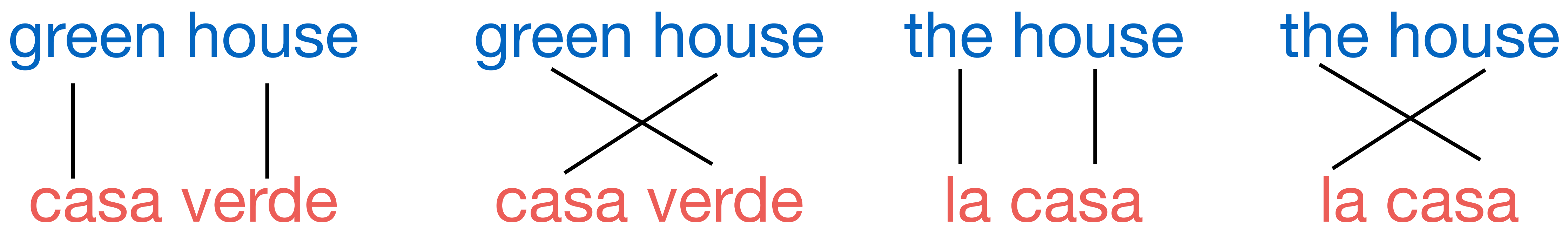
first, for all alignments, let's compute $p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$



$$p(f, a \mid e) = t(\text{casa} \mid \text{green}) \times t(\text{verde} \mid \text{house}) = \frac{1}{2} \times \frac{1}{4} = \frac{1}{8}$$

E-Step 2: repeat with new translation probabilities

first, for all alignments, let’s compute $p(f, a \mid e) = \prod_{j=1}^{m} t(f_j \mid e_{a_j})$



$p(f, a \mid e) = \frac{1}{8}$ $p(f, a \mid e) = \frac{1}{4}$ $p(f, a \mid e) = \frac{1}{4}$ $p(f, a \mid e) = \frac{1}{8}$

and keep doing this for some number of iterations!

next time… neural MT + decoding + evaluation