Machine translation: Word-based models and the EM algorithm

Chris Callison-Burch
(slides borrowed from Philipp Koehn)

December 3, 2007

JOHNS HOPKINS UNIVERSITY

Machine translation

- Task: make sense of foreign text like

毒品

本册子爲家長們提供實際和有用的關于毒品
的信息，包括如何減少使用非法毒品的危險。
它有助於您和您的家人討論有關毒品的問題。
這本小册子的主要内容已錄在磁帶上。如果您
想索取一盒免費的磁帶(中文)，請在下面的

- One of the oldest problems in Artificial Intelligence
- Solutions may many encompass many other NLP applications: parsing, generation, word sense disambiguation, named entity recognition, transliteration, pronoun resolution, etc.

The Rosetta stone

- Egyptian language was a mystery for centuries
- 1799 a stone with Egyptian text and its translation into Greek was found

$\Rightarrow$ Allowed people to *learn* how to translate Egyptian

Modern day Rosetta stone

what is more , the relevant cost dynamic is completely under control.	im übrigen ist die diesbezügliche kostenentwicklung völlig unter kontrolle .
sooner or later we will have to be sufficiently progressive in terms of own resources as a basis for this fair tax system .	früher oder später müssen wir die notwendige progressivität der eigenmittel als grundlage dieses gerechten steuersystems zur sprache bringen .
we plan to submit the first accession partnership in the autumn of this year .	wir planen , die erste beitrittspartnerschaft im herbst dieses jahres vorzulegen .
it is a question of equality and solidarity .	hier geht es um gleichberechtigung und solidarität .
the recommendation for the year 1999 has been formulated at a time of favourable developments and optimistic prospects for the european economy .	die empfehlung für das jahr 1999 wurde vor dem hintergrund günstiger entwicklungen und einer für den kurs der europäischen wirtschaft positiven perspektive abgegeben .
that does not , however , detract from the deep appreciation which we have for this report .	im übrigen tut das unserer hohen wertschätzung für den vorliegenden bericht keinen abbruch .

Parallel data

- Lots of translated text available: 100s of million words of translated text for some language pairs
  - a book has a few 100,000s words
  - an educated person may read 10,000 words a day

  → 3.5 million words a year

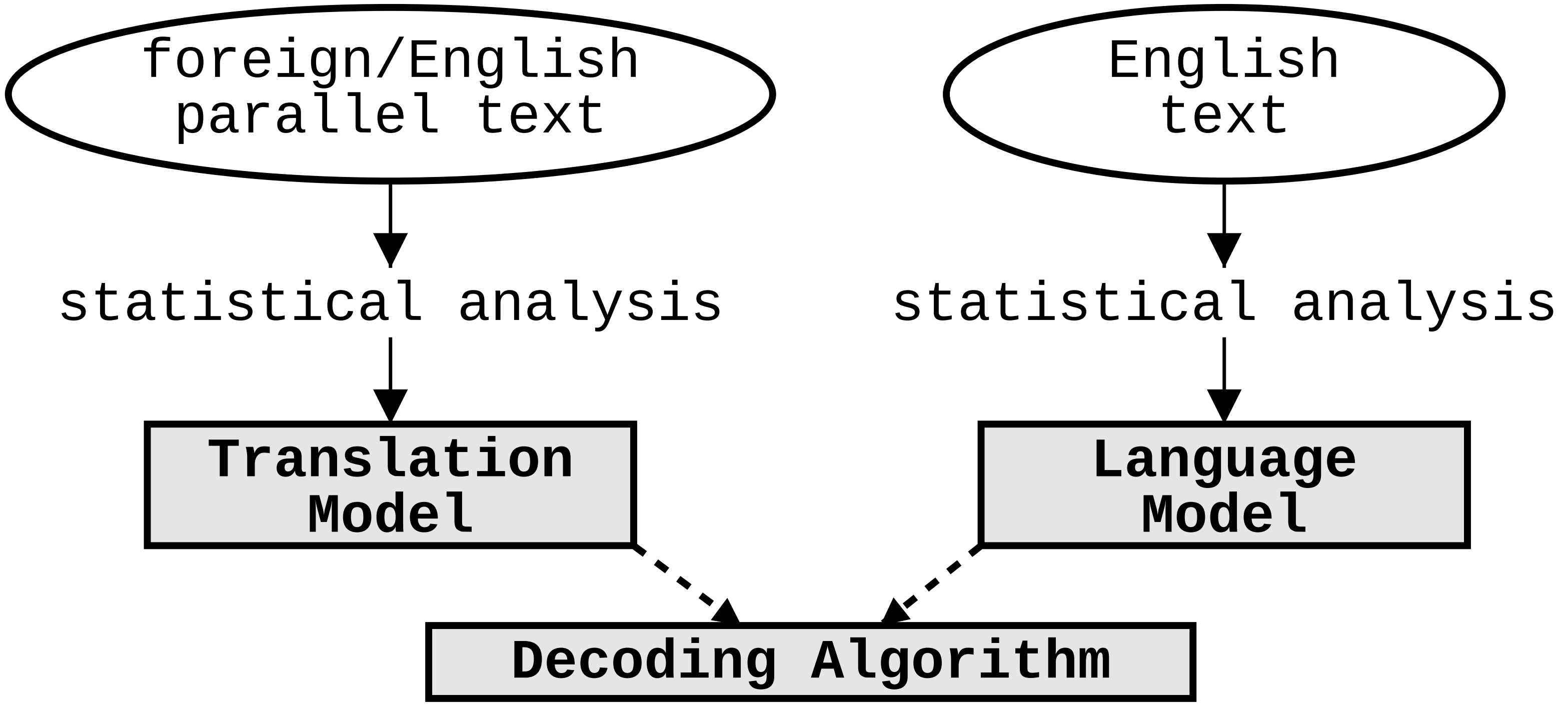
  → *300 million a lifetime*

  → soon computers will be able to see more translated text than humans read in a lifetime

⇒ Machines *can learn* how to translated foreign languages

Statistical Machine Translation

- Components: Translation model, language model, decoder



Lexical translation

- How to translate a word → look up in dictionary

  Haus — *house, building, home, household, shell.*

- *Multiple translations*
  - some more frequent than others
  - for instance: *house*, and *building* most common
  - special cases: *Haus* of a *snail* is its *shell*

Collect statistics

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

Translation of $\boldsymbol{Haus}$	**Count**
house	8,000
building	1,600
home	200
household	150
shell	50

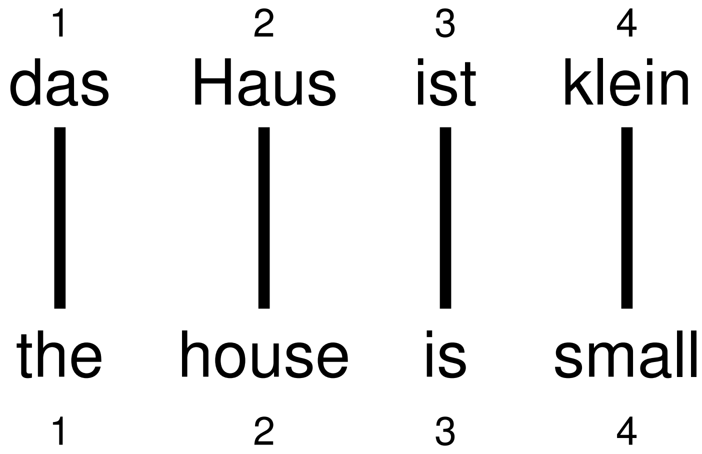
Estimate translation probabilities

- *Maximum likelihood estimation*

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

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

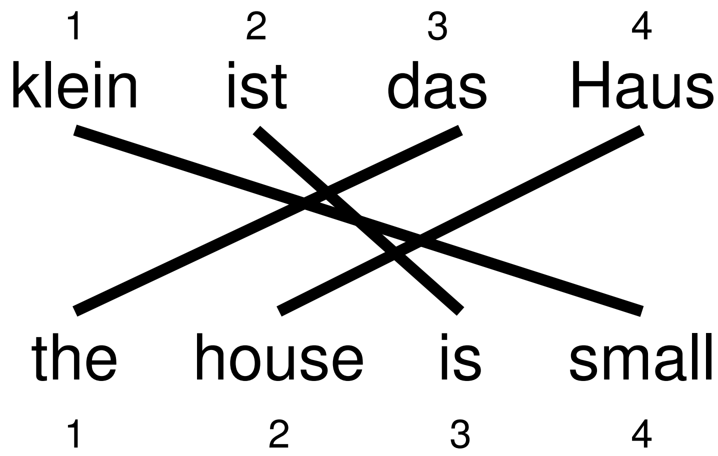
Alignment function

- 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

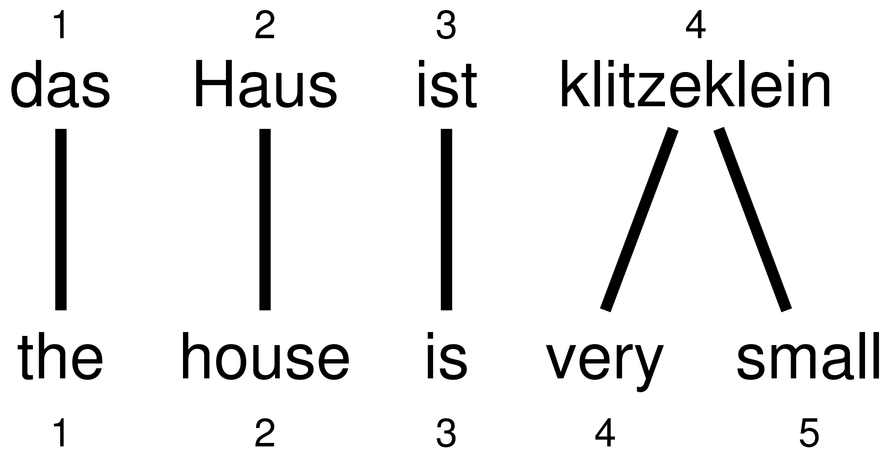


$$a : \{1 \rightarrow 3, 2 \rightarrow 4, 3 \rightarrow 2, 4 \rightarrow 1\}$$

JOHNS HOPKINS UNIVERSITY

One-to-many translation

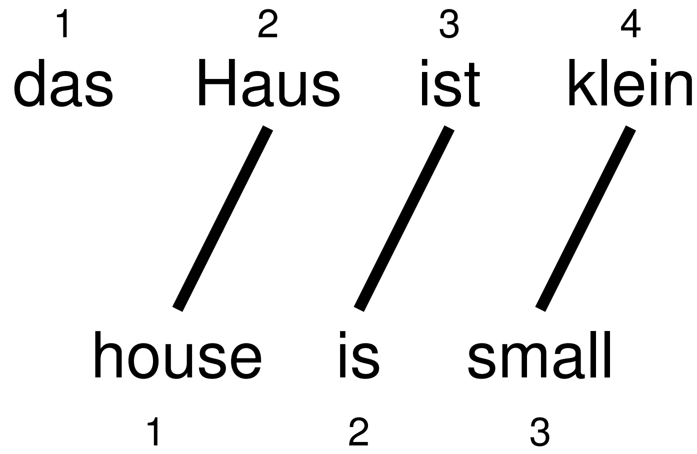
- A source word may translate into **multiple** target words



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

Dropping words

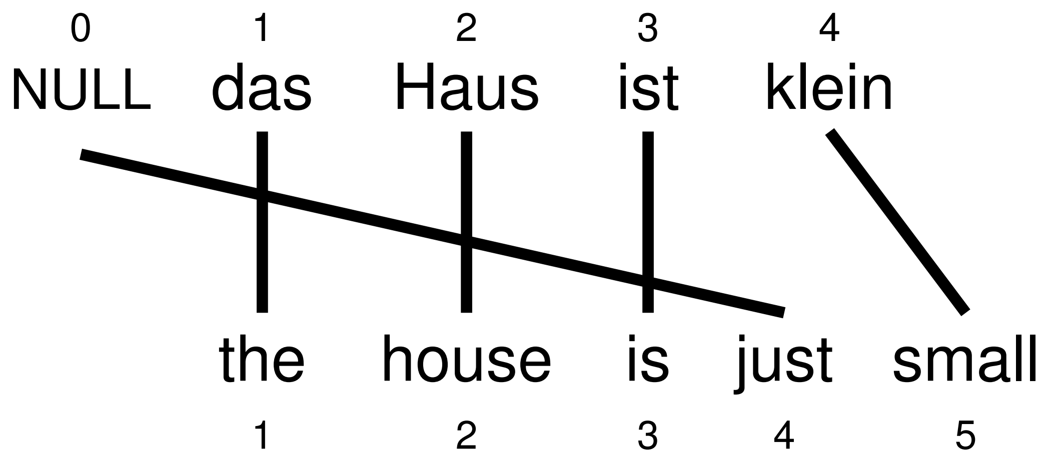
- Words may be **dropped** when translated
  - The German article *das* is dropped



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

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



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

IBM Model 1

- *Generative model*: break up translation process into smaller steps
  - **IBM Model 1** only uses *lexical translation*
- Translation probability
  - for a foreign sentence **f** $= (f_1, ..., f_{l_f})$ of length l_f
  - to an English sentence **e** $= (e_1, ..., e_{l_e})$ of length l_e
  - with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \rightarrow i$

$$p(\mathbf{e}, a|\mathbf{f}) = \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

Example

das

e	$t(e\|f)$
the	0.7
that	0.15
which	0.075
who	0.05
this	0.025

Haus

e	$t(e\|f)$
house	0.8
building	0.16
home	0.02
household	0.015
shell	0.005

ist

e	$t(e\|f)$
is	0.8
's	0.16
exists	0.02
has	0.015
are	0.005

klein

e	$t(e\|f)$
small	0.4
little	0.4
short	0.1
minor	0.06
petty	0.04

$$
\begin{aligned}
p(e,a|f) &= t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein}) \\
&= 0.7 \times 0.8 \times 0.8 \times 0.4 \\
&= 0.0028
\end{aligned}
$$

Learning lexical translation models

- We would like to *estimate* the lexical translation probabilities $t(e|f)$ from a parallel corpus
- ... but we do not have the alignments
- **Chicken and egg problem**
  - if we had the *alignments*,
    $\rightarrow$ we could estimate the *parameters* of our generative model
  - if we had the *parameters*,
    $\rightarrow$ we could estimate the *alignments*

EM algorithm

- **Incomplete data**
  - if we had *complete data*, would could estimate *model*
  - if we had *model*, we could fill in the *gaps in the data*
- **Expectation Maximization (EM)** in a nutshell
  - initialize model parameters (e.g. uniform)
  - assign probabilities to the missing data
  - estimate model parameters from completed data
  - iterate

IBM Model 1 and EM

- EM Algorithm consists of two steps
- **Expectation-Step**: Apply model to the data
  - parts of the model are hidden (here: alignments)
  - using the model, assign probabilities to possible values
- **Maximization-Step**: Estimate model from data
  - take assign values as fact
  - collect counts (weighted by probabilities)
  - estimate model from counts
- Iterate these steps until **convergence**

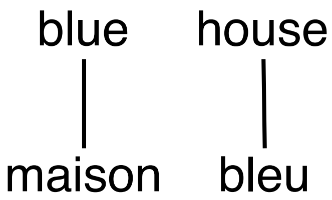
IBM Model 1 and EM

Step 1. Set parameter values uniformly.

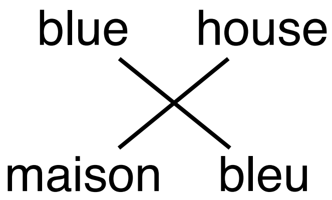
$t(bleu|house) = \frac{1}{2}$
$t(maison|house) = \frac{1}{2}$
$t(bleu|blue) = \frac{1}{2}$
$t(maison|blue) = \frac{1}{2}$

IBM Model 1 and EM

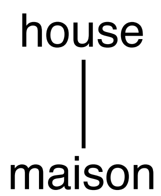
Compute $P(a, f|e)$ for all alignments.



$$P(a, f|e) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$$



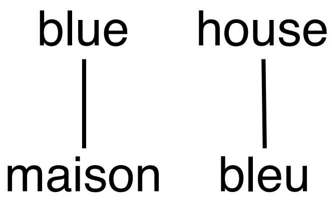
$$P(a, f|e) = \frac{1}{2} * \frac{1}{2} = \frac{1}{4}$$



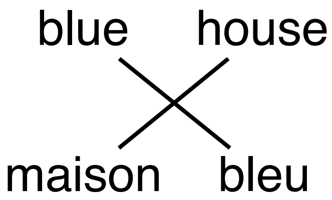
$$P(a, f|e) = \frac{1}{2}$$

IBM Model 1 and EM

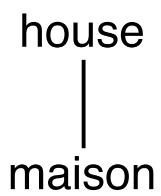
Step 3. Normalize $P(a, f|e)$ values to yield $P(a|e, f)$ values.



$$P(a|e, f) = \tfrac{1}{4} / \tfrac{2}{4} = \tfrac{1}{2}$$



$$P(a|e, f) = \tfrac{1}{4} / \tfrac{2}{4} = \tfrac{1}{2}$$



$P(a|e, f) = \frac{1}{2} / \frac{1}{2} = 1$ There's only one alignment, so $P(a|e, f)$ will be 1 always.

IBM Model 1 and EM

Step 4. Collect fractional counts.

$tc(bleu|house) = \frac{1}{2}$
$tc(maison|house) = \frac{1}{2} + 1 = \frac{3}{2}$
$tc(bleu|blue) = \frac{1}{2}$
$tc(maison|blue) = \frac{1}{2}$

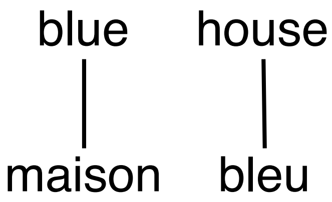
IBM Model 1 and EM

Step 5. Normalize fractional counts to get revised parameter values.

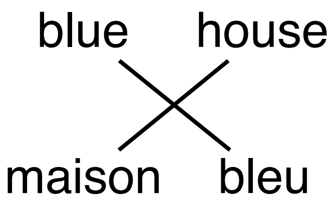
$t(bleu|house) = \frac{1}{2}/\frac{4}{2} = \frac{1}{4}$
$t(maison|house) = \frac{3}{2}/\frac{4}{2} = \frac{3}{4}$
$t(bleu|blue) = \frac{1}{2}/1 = \frac{1}{2}$
$t(maison|blue) = \frac{1}{2}/1 = \frac{1}{2}$

IBM Model 1 and EM

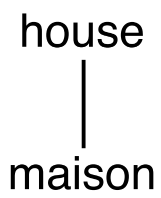
Repeat Step 2. Compute $P(a, f|e)$ for all alignments.



$$P(a, f|e) = \frac{1}{4} * \frac{1}{2} = \frac{1}{8}$$



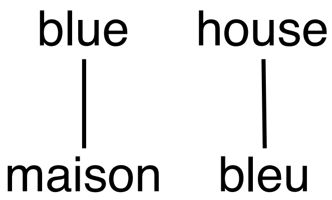
$$P(a, f|e) = \frac{3}{4} * \frac{1}{2} = \frac{3}{8}$$



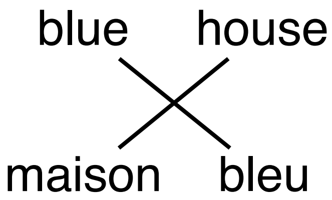
$$P(a, f|e) = \frac{3}{4}$$

IBM Model 1 and EM

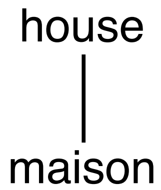
Repeat Step 3. Normalize $P(a, f|e)$ values to yield $P(a|e, f)$ values.



$$P(a|e, f) = \frac{1}{4}$$



$$P(a|e, f) = \frac{1}{4}/\frac{2}{4} = \frac{3}{4}$$



$$P(a|e, f) = 1$$

IBM Model 1 and EM

Repeat Step 4. Collect fractional counts.

$tc(bleu|house) = \frac{1}{4}$
$tc(maison|house) = \frac{3}{4} + 1 = \frac{7}{4}$
$tc(bleu|blue) = \frac{3}{4}$
$tc(maison|blue) = \frac{1}{4}$

IBM Model 1 and EM

Repeat Step 5. Normalize fractional counts to get revised parameter values.

$t(bleu|house) = \frac{1}{8}$
$t(maison|house) = \frac{7}{8}$
$t(bleu|blue) = \frac{3}{4}$
$t(maison|blue) = \frac{1}{4}$

IBM Model 1 and EM

Repeating steps 2-5 many times yields:

$$t(bleu|house) = 0.0001$$
$$t(maison|house) = 0.9999$$
$$t(bleu|blue) = 0.9999$$
$$t(maison|blue) = 0.0001$$

IBM Model 1 and EM: Pseudocode

```
initialize t(e|f) uniformly
do
  set count(e|f) to 0 for all e,f
  set total(f) to 0 for all f
  for all sentence pairs (e_s,f_s)
    for all words e in e_s
      total_s = 0
      for all words f in f_s
        total_s += t(e|f)
    for all words e in e_s
      for all words f in f_s
        count(e|f) += t(e|f) / total_s
        total(f)   += t(e|f) / total_s
  for all f in domain( total(.) )
    for all e in domain( count(.|f) )
      t(e|f) = count(e|f) / total(f)
until convergence
```

Higher IBM Models

IBM Model 1	lexical translation
IBM Model 2	adds absolute **reordering model**
IBM Model 3	adds **fertility model**
IBM Model 4	relative reordering model

- Only IBM Model 1 has *global maximum*
  - training of a higher IBM model builds on previous model
- Compuationally biggest change in Model 3
  $\rightarrow$ *exhaustive* count collection becomes computationally too expensive
  - **sampling** over high probability alignments is used instead

Next up: Decoding

Derivations for the Expectation Step

- We need to compute $p(a|\mathbf{e},\mathbf{f})$
- Applying the *chain rule*:

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

- We already have the formula for $p(\mathbf{e},\mathbf{a}|\mathbf{f})$ (definition of Model 1)
- We need to compute $p(\mathbf{e}|\mathbf{f})$

Derivations for the Expectation Step

$$
\begin{aligned}
p(\mathbf{e}|\mathbf{f}) &= \sum_a p(\mathbf{e}, a|\mathbf{f}) \\
&= \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e}, a|\mathbf{f}) \\
&= \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\
&= \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)
\end{aligned}
$$

Trick in the last line removes the need for an *exponential* number of products. This makes IBM Model 1 estimation **tractable**

Derivations for the Expectation Step

- Combine what we have:

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

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

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

Derivations for the Maximization Step

- Now we have to *collect counts*
- Evidence from a sentence pair **e,f** that word e is a translation of word f:

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

- With the same simplication as before:

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

Derivations for the Maximization Step

- After collecting these counts over a corpus, we can estimate the model:

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