MT is Hard

Ambiguities
- words
- morphology
- syntax
- semantics
- pragmatics

In der Innenstadt explodierte eine Autobombe

A car bomb exploded downtown

In the downtown, a car bomb exploded
Levels of Transfer

Interlingua

"meaning"

report_event[
  factivity=true
  explode(e, bomb, car)
  loc(e, downtown)
]

explodieren
  :arg0 Bombe
  :arg1 Auto
  :loc Innenstadt
  :tempus imperfect

In der Innenstadt explodierte eine Autobombe

detonate
  :arg0 bomb
  :arg1 car
  :loc downtown
  :time past

A car bomb exploded downtown

In der Innenstadt explodierte eine Autobombe

A car bomb exploded downtown
Two Views of Statistical MT

- **Direct modeling** (aka pattern matching)
  - I have *really good learning algorithms* and a bunch of *example inputs* (source language sentences) and *outputs* (target language translations)

- **Code breaking** (aka the noisy channel, Bayes rule)
  - I know the *target language*
  - I have example *translations texts* (example enciphered data)
MT as Direct Modeling

\[ \hat{e} = \arg \max_e p_\lambda(e \mid f) \]

- one model does everything
- trained to reproduce a corpus of translations
Noisy Channel Model

\[ \hat{e} = \arg \max_e p_\varphi(e) \times p_\theta(f \mid e) \]

- **Sent transmission**
  - English
- **Received transmission**
  - “French”
- **Recovered message**
  - English’

- **Language model**
- **Translation model**
Which is better?

- **Noisy channel** - \( p_\theta(e) \times p_\varphi(f \mid e) \)
  - easy to use monolingual target language data
  - search happens under a product of two models (individual models can be simple, product can be powerful)
  - obtaining probabilities requires renormalizing

- **Direct model** - \( p_\lambda(e \mid f) \)
  - directly model the process you care about
  - model must be very powerful
<table>
<thead>
<tr>
<th>Centauri-Arcturan Parallel Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a. ok-voon ororok sprok .</td>
</tr>
<tr>
<td>1b. at-voon bichat dat .</td>
</tr>
<tr>
<td>2a. ok-drubel ok-voon anok plok sprok .</td>
</tr>
<tr>
<td>2b. at-drubel at-voon pippat rrat dat .</td>
</tr>
<tr>
<td>3a. erok sprok izok hihok ghirok .</td>
</tr>
<tr>
<td>3b. totat dat arrat vat hilat .</td>
</tr>
<tr>
<td>4a. ok-voon anok drok brok jok .</td>
</tr>
<tr>
<td>4b. at-voon krat pippat sat lat .</td>
</tr>
<tr>
<td>5a. wiwok farok izok stok .</td>
</tr>
<tr>
<td>5b. totat jjat quat cat .</td>
</tr>
<tr>
<td>6a. lalok sprok izok jok stok .</td>
</tr>
<tr>
<td>6b. wat dat krat quat cat .</td>
</tr>
</tbody>
</table>

Translation challenge: **farok crrrok hihok yorok clok kantok ok-yurp**

(from Knight (1997): Automating Knowledge Acquisition for Machine Translation)
Noisy Channel Model: Phrase-Based MT

Translation Model: $P(f|e)$
- Source phrase
- Target phrase
- Translation features

Reranking Model
- Feature weights

Language Model: $P(e)$

Parallel corpus
- $f$
- $e$

Monolingual corpus
- $e$

Held-out parallel corpus
- $f$
- $e$

$\text{argmax}_e P(f|e)P(e)$
Phrase-Based MT

Translation Model $P(f|e)$

- Source phrase
- Target phrase
- Translation features

Reranking Model

Language Model $P(e)$

- Feature weights

$arg\max_e P(f|e)P(e)$
В этом смысле подобные действия частично дискредитируют систему американской демократии.
Phrase-Based System Overview

Morgen | fliege | ich | nach Kanada | zur Konferenz

Sentence-aligned corpus

Word alignments

Phrase table (translation model)

cat ||| chat ||| 0.9
cat ||| le chat ||| 0.8
dog ||| chien ||| 0.8
house ||| maison ||| 0.6
my house ||| ma maison ||| 0.9
language ||| langue ||| 0.9

…
Lexical Translation

- How do we translate a word? Look it up in the dictionary
  
  *Haus* — *house, building, home, household, shell*
  
- Multiple translations
  - some more frequent than others
  - different word senses, different registers, different inflections (?)
    - *house, home* are common
  
- *shell* is specialized (the Haus of a snail is a shell)
How common is each?

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

<table>
<thead>
<tr>
<th>Translation of Haus</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>house</td>
<td>8,000</td>
</tr>
<tr>
<td>building</td>
<td>1,600</td>
</tr>
<tr>
<td>home</td>
<td>200</td>
</tr>
<tr>
<td>household</td>
<td>150</td>
</tr>
<tr>
<td>shell</td>
<td>50</td>
</tr>
</tbody>
</table>
Maximum likelihood estimation

\[ \hat{p}_{\text{MLE}}(e \mid \text{Haus}) = \begin{cases} 
0.8 & \text{if } e = \text{house,} \\
0.16 & \text{if } e = \text{building,} \\
0.02 & \text{if } e = \text{home,} \\
0.015 & \text{if } e = \text{household,} \\
0.005 & \text{if } e = \text{shell.} 
\end{cases} \]
- Goal: a model $p(e | f, m)$
- where $e$ and $f$ are complete English and Foreign sentences

\[
e = \langle e_1, e_2, \ldots, e_m \rangle \quad f = \langle f_1, f_2, \ldots, f_n \rangle
\]
Alignment Function

- In a parallel text (or when we translate), we align words in one language with the words in the other.
- Alignments are represented as vectors of positions:

\[
\mathbf{f} = \langle f_1, f_2, \ldots, f_n \rangle
\]

\[
\mathbf{e} = \langle e_1, e_2, \ldots, e_m \rangle
\]

\[
\mathbf{a} = (1, 2, 3, 4)
\]
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 $a : i \to j$
- Example

$$a = (1, 2, 3, 4)$$
Reordering

- Words may be reordered during translation.

\[ a = (3, 4, 2, 1) \]
One-to-many Translation

- A source word may translate into more than one target word

\[ a = (1, 2, 3, 4, 4) \]
Word Dropping

- A source word may not be translated at all

```
das  Haus  ist  klein
1    2     3     4
```

```
house  is  small
1     2     3
```

\[ a = (2, 3, 4) \]
Word Insertion

- Words may be inserted during translation
  - English *just* does not have an equivalent
  - But it must be explained - we typically assume every source sentence contains a *NULL* token

\[ a = (1, 2, 3, 0, 4) \]
Many-to-one Translation

- More than one source word may not translate as a unit in lexical translation.

```
a = ???
a = (1, 2, (3, 4))
```
Generative Story

\[ p(e \mid f, m) \]

Mary did not slap the green witch
Mary did not slap the green witch
Generative Story

Mary did not slap the green witch

fertility

Mary not slap slap slap the green witch n(3|slap)
Generative Story

Mary did not slap the green witch

fertility

Mary not slap slap slap the green witch

Mary not slap slap slap NULL the green witch

n(3|slap)
Generative Story

Mary did not slap the green witch

fertility

NULL insertion

n(3|slap)
P(NULL)
Generative Story

Mary did not slap the green witch

fertility

NULL insertion

Mary not slap slap slap the green witch

Mary not slap slap slap NULL the green witch

Mary no daba una botefada a la verde bruja

n(3|slap)
P(NULL)
Mary did not slap the green witch

- **Fertility**
  - Mary not slap slap slap the green witch
  - Mary not slap slap null the green witch
  - Mary no daba una botefada a la verde bruja

- **NULL Insertion**
  - n(3|slap)
  - P(NULL)

- **Lexical Translation**
  - t(la|the)
Generative Story

Mary did not slap the green witch

- Mary not slap slap slap the green witch
- Mary not slap slap slap NULL the green witch
- Mary no daba una botefada a la verde bruja

fertility
NULL insertion
lexical translation

n(3|slap)
P(NULL)
t(la|the)
Generative Story

Mary did not slap the green witch

fertility

NULL insertion

lexical translation

distortion

Mary not slap slap slap the green witch

Mary not slap slap slap NULL the green witch

Mary no daba una botefada a la verde bruja

n(3|slap)

P(NULL)

t(la|the)

d(j|i)
The IBM Models 1--5 (Brown et al. 93)

Mary did not slap the green witch

- Mary not slap slap slap the green witch
- Mary not slap slap slap slap NULL the green witch
- Mary no daba una botefada a la verde bruja
- Mary no daba una botefada a la bruja verde

fertility

NULL insertion

lexical translation

distortion

n(3|slap)
P(NULL)
t(la|the)
d(j|i)

[from Al-Onaizan and Knight, 1998]
Alignment Models

- IBM Model 1: lexical translation
- IBM Model 2: alignment model, global monotonicity
- HMM model: local monotonicity
- fastalign: efficient reparametrization of Model 2
- IBM Model 3: fertility
- IBM Model 4: relative alignment model
- IBM Model 5: deficiency
- +many more
$P(e, a | f) = \prod p_f \prod p_t \prod p_d$

Mary did not slap the green witch

Mary not slap slap slap the green witch

Mary not slap slap slap NULL the green witch

Mary no daba una botefada a la verde bruja

Mary no daba una botefada a la bruja verde

$P(NULL)$

$t(la|the)$

$d(j|i)$

$fertility$

$NULL$

$lexical$

$translation$

$distortion$
The diagram illustrates the computation of the probability of an event \( P(e|f) \) as:

\[
P(e|f) = \sum_{\text{all_possible_alignments}} \prod p_f \prod p_t \prod p_d
\]
IBM Model 1

- Generative model: break up translation process into smaller steps
- Simplest possible **lexical translation model**
- Additional assumptions
  - All alignment decisions are independent
  - The alignment distribution for each $a_i$ is uniform over all source words and NULL
IBM Model 1

- **Translation probability**
  - for a foreign sentence \( f = (f_1, ..., f_{lf}) \) of length \( l_f \)
  - to an English sentence \( e = (e_1, ..., e_{le}) \) of length \( l_e \)
  - with an alignment of each English word \( e_j \) to a foreign word \( f_i \) according to the alignment function \( a : j \rightarrow i \)

\[
p(e, a | f) = \frac{\epsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j | f_{a(j)})
\]

- parameter \( \epsilon \) is a normalization constant
### Example

<table>
<thead>
<tr>
<th>das</th>
<th>Haus</th>
<th>ist</th>
<th>klein</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>e</em></td>
<td>*t(e</td>
<td>f)*</td>
<td><em>e</em></td>
</tr>
<tr>
<td>the</td>
<td>0.7</td>
<td>house</td>
<td>0.8</td>
</tr>
<tr>
<td>that</td>
<td>0.15</td>
<td>building</td>
<td>0.16</td>
</tr>
<tr>
<td>which</td>
<td>0.075</td>
<td>home</td>
<td>0.02</td>
</tr>
<tr>
<td>who</td>
<td>0.05</td>
<td>household</td>
<td>0.015</td>
</tr>
<tr>
<td>this</td>
<td>0.025</td>
<td>shell</td>
<td>0.005</td>
</tr>
</tbody>
</table>

\[
p(e, a|f) = \frac{\epsilon}{4^3} \times t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein})
\]

\[
= \frac{\epsilon}{4^3} \times 0.7 \times 0.8 \times 0.8 \times 0.4
\]

\[
= 0.0028 \epsilon
\]
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,
    - → we could estimate the parameters of our generative model (MLE)
  - if we had the parameters,
    - → we could estimate the alignments
Incomplete data
- if we had complete data, we could estimate the model
- if we had the model, we could fill in the gaps in the data

Expectation Maximization (EM) in a nutshell

1. initialize model parameters (e.g. uniform, random)
2. assign probabilities to the missing data
3. estimate model parameters from completed data
4. iterate steps 2–3 until convergence
EM Algorithm

- Initial step: all alignments equally likely
- Model learns that, e.g., la is often aligned with the
EM Algorithm

- After one iteration
- Alignments, e.g., between *la* and *the* are more likely
After another iteration
It becomes apparent that alignments, e.g., between *fleur* and *flower* are more likely (pigeon hole principle)
EM Algorithm

- Convergence
- Inherent hidden structure revealed by EM
EM Algorithm

... la maison ... la maison bleu ... la fleur ...

... the house ... the blue house ... the flower ...

\[
\begin{align*}
p(\text{la}|\text{the}) &= 0.453 \\
p(\text{le}|\text{the}) &= 0.334 \\
p(\text{maison}|\text{house}) &= 0.876 \\
p(\text{bleu}|\text{blue}) &= 0.563
\end{align*}
\]

- Parameter estimation from the aligned corpus
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 assigned values as fact
  - collect counts (weighted by lexical translation probabilities)
  - estimate model from counts

- Iterate these steps until convergence
We need to be able to compute:

- Expectation-Step: probability of alignments
- Maximization-Step: count collection
IBM Model 1 and EM

t-table Probabilities

\[ p(\text{the}|\text{la}) = 0.7 \quad p(\text{house}|\text{la}) = 0.05 \]
\[ p(\text{the}|\text{maison}) = 0.1 \quad p(\text{house}|\text{maison}) = 0.8 \]
IBM Model 1 and EM

Probabilities

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

Alignments

- \text{la} \rightarrow \text{the}
- \text{maison} \rightarrow \text{house}
- \text{la} \rightarrow \text{the}
- \text{maison} \rightarrow \text{house}
- \text{la} \rightarrow \text{the}
- \text{maison} \rightarrow \text{house}
- \text{la} \rightarrow \text{the}
- \text{maison} \rightarrow \text{house}
IBM Model 1 and EM

**Probabilities**

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

**Alignments**

- $p(e, a|f) = 0.56$
- $p(e, a|f) = 0.035$
- $p(e, a|f) = 0.08$
- $p(e, a|f) = 0.005$
IBM Model 1 and EM

**Probabilities**

\[
\begin{align*}
    p(\text{the}|\text{la}) &= 0.7 & p(\text{house}|\text{la}) &= 0.05 \\
    p(\text{the}|\text{maison}) &= 0.1 & p(\text{house}|\text{maison}) &= 0.8
\end{align*}
\]

**Alignments**

- \(\text{la} \rightarrow \text{the}\) \(\text{maison} \rightarrow \text{house}\)\( p(e, a|f) = 0.56 \)
- \(\text{la} \rightarrow \text{the}\) \(\text{maison} \rightarrow \text{house}\)\( p(e, a|f) = 0.035 \)
- \(\text{la} \rightarrow \text{the}\) \(\text{maison} \rightarrow \text{house}\)\( p(e, a|f) = 0.08 \)
- \(\text{la} \rightarrow \text{the}\) \(\text{maison} \rightarrow \text{house}\)\( p(e, a|f) = 0.005 \)

**Applying the chain rule:**

\[
p(a|e, f) = \frac{p(e, a|f)}{p(e|f)}
\]

\[
p(e, a) = p(e)p(a|e)
\]
We need to compute $p(e|f)$

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

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

$$= \sum_{a(1)=0}^{l_f} \ldots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f + 1)^l_e} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$
IBM Model 1 and EM: Expectation Step

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

- Note the trick in the last line
  - removes the need for an exponential number of products
  → this makes IBM Model 1 estimation tractable
The Trick

\[(\text{case } l_e = l_f = 2)\]

\[\sum_{a(1)=0}^{2} \sum_{a(2)=0}^{2} \epsilon = \frac{3^2}{2} \prod_{j=1}^{2} t(e_j | f_{a(j)}) = \]

\[= t(e_1 | f_0) t(e_2 | f_0) + t(e_1 | f_0) t(e_2 | f_1) + t(e_1 | f_0) t(e_2 | f_2) + \]
\[+ t(e_1 | f_1) t(e_2 | f_0) + t(e_1 | f_1) t(e_2 | f_1) + t(e_1 | f_1) t(e_2 | f_2) + \]
\[+ t(e_1 | f_2) t(e_2 | f_0) + t(e_1 | f_2) t(e_2 | f_1) + t(e_1 | f_2) t(e_2 | f_2) = \]

\[= t(e_1 | f_0) (t(e_2 | f_0) + t(e_2 | f_1) + t(e_2 | f_2)) + \]
\[+ t(e_1 | f_1) (t(e_2 | f_1) + t(e_2 | f_1) + t(e_2 | f_2)) + \]
\[+ t(e_1 | f_2) (t(e_2 | f_2) + t(e_2 | f_1) + t(e_2 | f_2)) = \]

\[= (t(e_1 | f_0) + t(e_1 | f_1) + t(e_1 | f_2)) (t(e_2 | f_2) + t(e_2 | f_1) + t(e_2 | f_2)) \]
Combine what we have:

\[ p(a|e,f) = \frac{p(e,a|f)}{p(e|f)} \]

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

\[ = \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)} \]
IBM Model 1 and EM: Expectation Step

**t-table**

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

**Alignments**

- \( p(e,a|f) = 0.56 \)
- \( p(e,a|f) = 0.035 \)
- \( p(e,a|f) = 0.08 \)
- \( p(e,a|f) = 0.005 \)

**E-step**

- \( p(a|e,f) = 0.824 \)
- \( p(a|e,f) = 0.052 \)
- \( p(a|e,f) = 0.118 \)
- \( p(a|e,f) = 0.007 \)

\[
p(a|e,f) = \frac{p(e,a|f)}{p(e|f)}
\]
Now we have to collect counts

Evidence from a sentence pair $e, f$ that word $e$ is a translation of word $f$:

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

With the same simplication as before:

$$c(e|f; e, f) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e, e_j) \sum_{i=0}^{l_f} \delta(f, f_i)$$
IBM Model 1 and EM: Maximization Step

Probabilities

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

Alignments

- $p(e, a|f) = 0.56$
- $p(e, a|f) = 0.035$
- $p(e, a|f) = 0.08$
- $p(e, a|f) = 0.005$

E-step

- $p(a|e, f) = 0.824$
- $p(a|e, f) = 0.052$
- $p(a|e, f) = 0.118$
- $p(a|e, f) = 0.007$

M-step Counts

- $c(\text{the}|\text{la}) = 0.824 + 0.052$
- $c(\text{house}|\text{la}) = 0.052 + 0.007$
- $c(\text{the}|\text{maison}) = 0.118 + 0.007$
- $c(\text{house}|\text{maison}) = 0.824 + 0.118$
After collecting these counts over a corpus, we can estimate the model:

\[
t(e|f; e, f) = \frac{\sum_{(e,f)} c(e|f; e, f))}{\sum_e \sum_{(e,f)} c(e|f; e, f))}
\]
IBM Model 1 and EM: Maximization Step

**t-table**

Probabilities

\[
\begin{align*}
    p(\text{the}|\text{la}) &= 0.7 \\
    p(\text{the}|\text{maison}) &= 0.1 \\
    p(\text{house}|\text{la}) &= 0.05 \\
    p(\text{house}|\text{maison}) &= 0.8
\end{align*}
\]

**E-step**

Alignments

\[
\begin{align*}
    p(a|e, f) &= 0.824 \\
    p(a|e, f) &= 0.052 \\
    p(a|e, f) &= 0.118 \\
    p(a|e, f) &= 0.007
\end{align*}
\]

**M-step**

Counts

\[
\begin{align*}
    c(\text{the}|\text{la}) &= 0.824 + 0.052 \\
    c(\text{the}|\text{maison}) &= 0.118 + 0.007 \\
    c(\text{house}|\text{la}) &= 0.052 + 0.007 \\
    c(\text{house}|\text{maison}) &= 0.824 + 0.118
\end{align*}
\]

Update t-table:

\[
p(\text{the}|\text{la}) = \frac{c(\text{the}|\text{la})}{c(\text{la})}
\]
IBM Model 1 and EM: Pseudocode

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

1: initialize \(t(e|f)\) uniformly
2: while not converged do
3:   // initialize
4:   count\((e|f)\) = 0 for all \(e, f\)
5:   total\((f)\) = 0 for all \(f\)
6:   for all sentence pairs \((e, f)\) do
7:     // compute normalization
8:     for all words \(e\) in \(e\) do
9:        s-total\((e)\) = 0
10:    for all words \(f\) in \(f\) do
11:        s-total\((e)\) += \(t(e|f)\)
12:        end for
13:    end for
14:   end for
15:   // collect counts
16:   for all words \(e\) in \(e\) do
17:      for all words \(f\) in \(f\) do
18:         count\((e|f)\) += \(\frac{t(e|f)}{s-total(e)}\)
19:         total\((f)\) += \(\frac{t(e|f)}{s-total(e)}\)
20:      end for
21:   end for
22:   end for
23:   // estimate probabilities
24:   for all foreign words \(f\) do
25:      for all English words \(e\) do
26:        \(t(e|f) = \frac{count(e|f)}{total(f)}\)
27:      end for
28:   end for
29: end while
### Convergence

<table>
<thead>
<tr>
<th></th>
<th>f</th>
<th>initial</th>
<th>1st it.</th>
<th>2nd it.</th>
<th>3rd it.</th>
<th>...</th>
<th>final</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>das</td>
<td>0.25</td>
<td>0.5</td>
<td>0.6364</td>
<td>0.7479</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>book</td>
<td>das</td>
<td>0.25</td>
<td>0.25</td>
<td>0.1818</td>
<td>0.1208</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>house</td>
<td>das</td>
<td>0.25</td>
<td>0.25</td>
<td>0.1818</td>
<td>0.1313</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>the</td>
<td>buch</td>
<td>0.25</td>
<td>0.25</td>
<td>0.1818</td>
<td>0.1208</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>book</td>
<td>buch</td>
<td>0.25</td>
<td>0.5</td>
<td>0.6364</td>
<td>0.7479</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>buch</td>
<td>0.25</td>
<td>0.25</td>
<td>0.1818</td>
<td>0.1313</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>book</td>
<td>ein</td>
<td>0.25</td>
<td>0.5</td>
<td>0.4286</td>
<td>0.3466</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>ein</td>
<td>0.25</td>
<td>0.5</td>
<td>0.5714</td>
<td>0.6534</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>the</td>
<td>haus</td>
<td>0.25</td>
<td>0.5</td>
<td>0.4286</td>
<td>0.3466</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>house</td>
<td>haus</td>
<td>0.25</td>
<td>0.5</td>
<td>0.5714</td>
<td>0.6534</td>
<td>...</td>
<td>1</td>
</tr>
</tbody>
</table>
Problems with IBM Model 1

- **Mary did not slap the green witch**
- **Mary not slap slap slap the green witch**
- **Mary not slap slap NULL the green witch**
- **Mary no daba una botefada a la verde bruja**
- **Mary no daba una botefada a la bruja verde**

** NULL insertion**

**fertility**

**lexical translation**

**distortion**

- \( n(3|\text{slap}) \)
- \( P(\text{NULL}) \)
- \( t(\text{la}|\text{the}) \)
- \( d(\text{j}|	ext{i}) \)
IBM Model 2

**Mary** did not slap the **green witch**

- **Mary** not slap slap slap the **green witch**
- **Mary** not slap slap slap **NULL** the **green witch**
- **Mary** no daba una botefada a la **verde bruja**
- **Mary** no daba una botefada a la **bruja verde**

- `fertility`
- `NULL insertion`
- `lexical translation`
- `monotonic alignment`
IBM Model 2

\[ p(e,a|f) = \epsilon \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) a(a(j)|j, l_e, l_f) \]

\[ p(e|f) = \epsilon \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_{a(j)}) a(a(j)|j, l_e, l_f) \]

- compare with Model 1:

\[ p(e,a|f) = \frac{\epsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \]
## Higher IBM Models

<table>
<thead>
<tr>
<th>IBM Model 1</th>
<th>lexical translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Model 2</td>
<td>adds absolute reordering model</td>
</tr>
<tr>
<td>IBM Model 3</td>
<td>adds fertility model</td>
</tr>
<tr>
<td>IBM Model 4</td>
<td>relative reordering model</td>
</tr>
<tr>
<td>IBM Model 5</td>
<td>fixes deficiency</td>
</tr>
</tbody>
</table>

Only IBM Model 1 has global maximum

- training of a higher IBM model builds on previous model

Computationally biggest change in Model 3

- trick to simplify estimation does not work anymore
  - exhaustive count collection becomes computationally too expensive
  - sampling over high probability alignments is used instead
The IBM Models 1--5 (Brown et al. 93)

fertility

NULL

insertion

lexical

translation

distortion

Mary did not slap the green witch

Mary not slap slap slap the green witch

Mary not slap slap slap NULL the green witch

Mary no daba una botefada a la verde bruja

Mary no daba una botefada a la bruja verde

n(3|slap)

P(NULL)

t(la|the)

d(j|i)

[from Al-Onaizan and Knight, 1998]
Given a sentence pair, which words correspond to each other?
Word Alignment?

Is the English word *does* aligned to the German *wohnt* (verb) or *nicht* (negation) or neither?
How do the idioms *kicked the bucket* and *biss ins grass* match up? Outside this exceptional context, *bucket* is never a good translation for *grass*. 
IBM Models create a many-to-one mapping
- words are aligned using an alignment function
- a function may return the same value for different input (one-to-many mapping)
- a function can not return multiple values for one input (no many-to-one mapping)

Real word alignments have many-to-many mappings
Symmetrization
Growing Heuristics

- Add alignment points from union based on heuristics
- Popular method: grow-diag-final-and
Evaluating Alignment Models

- How do we measure quality of a word-to-word model?
  - Method 1: use in an end-to-end translation system
    - Hard to measure translation quality
    - Option: human judges
    - Option: reference translations (NIST, BLEU)
    - Option: combinations (HTER)
    - Actually, no one uses word-to-word models alone as TMs
  - Method 2: measure quality of the alignments produced
    - Easy to measure
    - Hard to know what the gold alignments should be
    - Often does not correlate well with translation quality (like perplexity in LMs)
Alignment Error Rate
Alignment Error Rate

Possible links

P
Alignment Error Rate

Possible links $P$

Sure links $S$
Alignment Error Rate

Possible links $P$

Sure links $S$

Precision($\mathcal{A}$, $P$) = \[ \frac{|P \cap \mathcal{A}|}{|A|} \]

Recall($\mathcal{A}$, $S$) = \[ \frac{|S \cap \mathcal{A}|}{|S|} \]
Alignment Error Rate

Possible links $P$

Sure links $S$

\[
\text{Precision}(A, P) = \frac{|P \cap A|}{|A|}
\]

\[
\text{Recall}(A, S) = \frac{|S \cap A|}{|S'|}
\]

\[
\text{AER}(A, P, S) = 1 - \frac{|S \cap A| + |P \cap A|}{|S'| + |A|}
\]
Problems with Lexical Translation

- Complexity -- exponential in sentence length
- Weak reordering -- the output is not fluent
- Many local decisions -- error propagation
Phrase-Based Translation

\[ P(e, \text{alignment}|f) = p_{\text{segmentation}} p_{\text{translation}} p_{\text{reorderings}} \]
Phrase-Based MT

**Translation Model** \( P(f|e) \)

- **Source phrase**
- **Target phrase**
- **Translation features**

**Parallel corpus**

- \( e \)
- \( f \)

**Monolingual corpus**

- \( f \)

**Language Model** \( P(e) \)

**Reranking Model**

- **Feature weights**

\[ \arg\max_e P(f|e)P(e) \]