Lecture 14: Statistical Machine Translation

Julia Hockenmaier

juliahmr@illinois.edu
3324 Siebel Center
Lecture 14: Machine Translation II

Part 1: Word Alignment in the IBM models
Statistical Machine Translation

Given a Chinese input sentence (source)…

主席：各位議員，早晨。

…find the best English translation (target)

President: Good morning, Honourable Members.

We can formalize this as

\[ T^* = \arg\max_T P( T \mid S ) \]

Using **Bayes Rule** simplifies the modeling task, so this was the first approach for statistical MT (the so-called “noisy-channel model”):

\[ T^* = \arg\max_T P( T \mid S ) = \arg\max_T P( S \mid T )P(T) \]

where \( P( S \mid T ) \): translation model

\( P(T) \): language model
The noisy channel model

This is really just an application of Bayes’ rule:

\[
T^* = \arg\max_T P(T \mid S) \\
= \arg\max_T \frac{P(S \mid T)}{P(T)} \\
\text{Translation Model} \quad \text{Language Model}
\]

The translation model \( P(S \mid T) \) is intended to capture the faithfulness of the translation. [this is the noisy channel]

Since we only need \( P(S \mid T) \) to score \( S \), and don’t need it to generate a grammatical \( S \), it can be a relatively simple model.

\( P(S \mid T) \) needs to be trained on a parallel corpus

The language model \( P(T) \) is intended to capture the fluency of the translation.

\( P(T) \) can be trained on a (very large) monolingual corpus
IBM models

First statistical MT models, based on noisy channel:
Translate from (French/foreign) source $f$ to (English) target $e$ via a translation model $P(f \mid e)$ and a language model $P(e)$
The translation model goes from target $e$ to source $f$ via word alignments $a$: $P(f \mid e) = \sum_a P(f, a \mid e)$

Original purpose: Word-based translation models
Later: Were used to obtain word alignments, which are then used to obtain phrase alignments for phrase-based translation models

Sequence of 5 translation models
Model 1 is too simple to be used by itself, but can be trained very easily on parallel data.
IBM translation models: assumptions

The model “generates” the ‘foreign’ source sentence \( f \) conditioned on the ‘English’ target sentence \( e \) by the following stochastic process:

1. Generate the **length** of the source \( f \) with probability \( p = ... \)
2. Generate the **alignment** of the source \( f \) to the target \( e \) with probability \( p = ... \)
3. Generate the **words** of the source \( f \) with probability \( p = ... \)
Word alignment

John loves Mary.           … that John loves Mary.

Jean aime Marie.         … dass John Maria liebt.
Word alignment

<table>
<thead>
<tr>
<th></th>
<th>Maria</th>
<th>no</th>
<th>dió</th>
<th>una</th>
<th>bofetada</th>
<th>a</th>
<th>la</th>
<th>bruja</th>
<th>verde</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>did</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>not</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>slap</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>green</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>witch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Mary did not slap the green witch.
Word alignment

<table>
<thead>
<tr>
<th></th>
<th>Marie</th>
<th>a</th>
<th>traversé</th>
<th>le</th>
<th>lac</th>
<th>à</th>
<th>la</th>
<th>nage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>swim</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>across</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lake</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Word alignment

<table>
<thead>
<tr>
<th>Target</th>
<th>Marie</th>
<th>a</th>
<th>traversé</th>
<th>le</th>
<th>lac</th>
<th>à</th>
<th>la</th>
<th>nage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>swam</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>across</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lake</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One target word can be aligned to many source words.
Word alignment

One target word can be aligned to many source words. But each source word can only be aligned to one target word. This allows us to model $P(\text{source} \mid \text{target})$. 
Word alignment

<table>
<thead>
<tr>
<th>Target</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Marie traversé</td>
</tr>
<tr>
<td></td>
<td>a</td>
</tr>
<tr>
<td></td>
<td>le</td>
</tr>
<tr>
<td></td>
<td>lac</td>
</tr>
<tr>
<td></td>
<td>à</td>
</tr>
<tr>
<td></td>
<td>la</td>
</tr>
<tr>
<td></td>
<td>nage</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mary</th>
<th>swam</th>
</tr>
</thead>
<tbody>
<tr>
<td>across</td>
<td>the</td>
</tr>
<tr>
<td>lake</td>
<td></td>
</tr>
</tbody>
</table>

Some source words may not align to *any* target words.
Some source words may not align to *any* target words.
To handle this we assume a **NULL** word in the target sentence.
Representing word alignments

<table>
<thead>
<tr>
<th>Position</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreign</td>
<td>Marie</td>
<td>a</td>
<td>traversé</td>
<td>le</td>
<td>lac</td>
<td>à</td>
<td>la</td>
<td>nage</td>
</tr>
<tr>
<td>Alignment</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Every source word \( f[i] \) is aligned to one target word \( e[j] \) (incl. NULL). We represent alignments as a vector \( a \) (of the same length as the source) with \( a[i] = j \).
Lecture 14: Machine Translation II

Part 2: The IBM alignment models
The IBM models

Use the noisy channel (Bayes rule) to get the best (most likely) target translation $e$ for source sentence $f$:

$$\arg \max_e P(e|f) = \arg \max_e P(f|e)P(e)$$

The translation model $P(f|e)$ requires alignments $a$

$$P(f|e) = \sum_{a \in A(e,f)} P(f,a|e)$$

Generate $f$ and the alignment $a$ with $P(f,a|e)$:

$$P(f,a|e) = \underbrace{P(m|e)}_{\text{Length: } |f|=m} \prod_{j=1}^{m} \underbrace{P(a_j|a_{1..j-1},f_{1..j-1},m,e)}_{\text{Word alignment } a_j} \underbrace{P(f_j|a_{1..j}f_{1..j-1},e,m)}_{\text{Translation } f_j}$$

$m = \#\text{words in } f_j$

probability of alignment $a_j$

probability of word $f_j$
IBM model 1: Generative process

For each target sentence $e = e_1..e_n$ of length $n$:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>Mary</td>
<td>swam</td>
<td>across</td>
<td>the</td>
<td>lake</td>
<td></td>
</tr>
</tbody>
</table>

1. **Choose a length** $m$ for the source sentence (e.g. $m = 8$)

<table>
<thead>
<tr>
<th>Position</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

2. **Choose an alignment** $a = a_1...a_m$ for the source sentence

   Each $a_j$ corresponds to a word $e_i$ in $e$: $0 \leq a_j \leq n$

<table>
<thead>
<tr>
<th>Position</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

3. **Translate** each target word $e_{aj}$ into the source language

<table>
<thead>
<tr>
<th>Position</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alignment</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Translation</td>
<td>Marie</td>
<td>a traversé</td>
<td>le lac</td>
<td>à la nage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model parameters

Length probability \( P(m \mid n) \):
What’s the probability of generating a source sentence of length \( m \) given a target sentence of length \( n \)?
Count in training data, or use a constant

Alignment probability: \( P(a \mid m, n) \):
Model 1 assumes all alignments have the same probability:
For each position \( a_1...a_m \), pick one of the \( n+1 \) target positions uniformly at random

Translation probability: \( P(f_j = lac \mid a_j = i, e_i = lake) \):
In Model 1, these are the only parameters we have to learn.
IBM model 1: details

The **length probability** is constant:  
\[ P(m \mid e) = \varepsilon \]

The **alignment probability** is uniform  
\[ P(a_i \mid e) = 1/(n+1) \]

The **translation probability** depends only on \( e_{ai} \)  
\( \text{the corresponding target word}: \quad P(f_i \mid e_{ai}) \)

\[
P(f, a \mid e) = P(m \mid e) \prod_{j=1}^{m} P(a_j \mid a_{1..j-1}, f_{1..j-1}, m, e) \prod_{j=1}^{m} P(f_j \mid e_{a_j})
\]

All alignments have the same probability

Translation depends only on the aligned English word
Finding the best alignment

How do we find the best alignment between \( e \) and \( f \)?

\[
\hat{a} = \arg \max_a P(f, a|e)
\]

\[
= \arg \max_a \frac{\epsilon}{(n + 1)^m} \prod_{j=1}^{m} P(f_j|e_{a_j})
\]

\[
= \arg \max_a \prod_{j=1}^{m} P(f_j|e_{a_j})
\]

\[
\hat{a}_j = \arg \max_{a_j} P(f_j|e_{a_j})
\]
Learning translation probabilities

The only parameters that need to be learned are the **translation probabilities** $P(f | e)$

$$P(f_j = lac \mid e_i = lake)$$

If the training corpus had word alignments, we could simply count how often ‘lake’ is aligned to ‘lac’:

$$P(lac \mid lake) = \frac{\text{count}(lac, lake)}{\sum_w \text{count}(w, lake)}$$

But we don’t have gold word alignments.

So, instead of relative frequencies, we have to use expected relative frequencies:

$$P( lac \mid lake) = \langle \text{count}(lac, lake) \rangle / \langle \sum_w \text{count}(w, lake) \rangle$$
Training Model 1 with EM

The only parameters that need to be learned are the translation probabilities $P(f \mid e)$

We use the **EM algorithm** to estimate these parameters from a corpus with $S$ sentence pairs $s = \langle f^{(s)}, e^{(s)} \rangle$ with alignments $A(f^{(s)}, e^{(s)})$

**Initialization:** guess $P(f \mid e)$

**Expectation step:** compute expected counts

$$\langle c(f, e) \rangle = \sum_{s \in S} \langle c(f, e \mid e^{(s)}, f^{(s)}) \rangle$$

**Maximization step:** recompute probabilities $P(f \mid e)$

$$\hat{P}(f \mid e) = \frac{\langle c(f, e) \rangle}{\sum_{f'} \langle c(f', e) \rangle}$$
Expectation-Maximization (EM)

1. Initialize a first model, $M^{(0)}$

2. **Expectation (E) step:**
   Go through training data to gather expected counts
   $\langle \text{count}(lac, lake) \rangle$

3. **Maximization (M) step:**
   Use expected counts to compute a new model $M^{(i+1)}$
   
   \[
P^{(i+1)}(lac \mid lake) = \langle \text{count}(lac, lake) \rangle / \langle \sum_w \text{count}(w, lake) \rangle
   \]

4. **Check for convergence:**
   Compute log-likelihood of training data with $M_{i+1}$
   If the difference between new and old log-likelihood smaller than a threshold, stop. Else go to 2.
The E-step

Compute the expected count $\langle c(f, e|f, e) \rangle$:

$$\langle c(f, e|f, e) \rangle = \sum_{a \in A(f,e)} P(a|f,e) \cdot c(f,e|a,e,f)$$

How often are $f,e$ aligned in $a$?

$$P(a|f,e) = \frac{P(a,f|e)}{P(f|e)} = \frac{P(a,f|e)}{\sum_{a'} P(a',f|e)}$$

$$P(a,f|e) = \prod_j P(f_j|e_{a_j})$$

$$\langle c(f,e|f,e) \rangle = \sum_{a \in A(f,e)} \frac{\prod_j P(f_j|e_{a_j})}{\sum_{a'} \prod_j P(f_j|e_{a_j})} \cdot c(f,e|a,e,f)$$

We need to know $P(f_j|e_{a_j})$, the probability that word $f_j$ is aligned to word $e_{a_j}$ under the alignment $a$. 
Other translation models

Model 1 is a very simple (and not very good) translation model.

IBM models 2-5 are more complex. They take into account:

– “fertility”: the number of foreign words generated by each target word
– the word order and string position of the aligned words
Part 3: Phrase-based translation models

Lecture 14: Machine Translation II
Phrase-based translation models

Assumption: fundamental units of translation are phrases:

Chair: 各位議員，早晨。

President (in Cantonese): Good morning, Honourable Members.

Phrase-based model of $P(F \mid E)$:
1. Split target sentence deterministically into phrases $ep_1...ep_n$
2. Translate each target phrase $ep_i$ into source phrase $fp_i$ with translation probability $\varphi(fp_i \mid ep_i)$
3. Reorder foreign phrases with distortion probability $d(a_i - b_{i-1}) = c|a_i - b_{i-1} - 1|$
   
   $a_i$ = start position of source phrase generated by $e_i$
   
   $b_{i-1}$ = end position of source phrase generated by $e_{i-1}$
Phrase-based models of $P(f|e)$

Split target sentence $e = e_1..n$ into phrases $e_{p1}..e_{pN}$:

[[The green witch] [is] [at home] [this week]]

Translate each target phrase $e_{pi}$ into source phrase $f_{pi}$ with translation probability $P(f_{pi}|e_{pi})$:

[[The green witch] = [die grüne Hexe], ...]

Arrange the set of source phrases $\{f_{pi}\}$ to get $s$ with distortion probability $P(f|\{f_{pi}\})$:

[[Diese Woche] [ist] [die grüne Hexe] [zuhause]]

$$P(f|e = \langle e_{p1}, ..., e_{pl} \rangle) = \prod_{i} P(f_{pi}|e_{pi}) P(f|\{f_{pi}\})$$
Translation probability $P( fp_i \mid ep_i )$

Phrase translation probabilities can be obtained from a phrase table:

<table>
<thead>
<tr>
<th>EP</th>
<th>FP</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>green witch</td>
<td>grüne Hexe</td>
<td>…</td>
</tr>
<tr>
<td>at home</td>
<td>zuhause</td>
<td>10534</td>
</tr>
<tr>
<td>at home</td>
<td>daheim</td>
<td>9890</td>
</tr>
<tr>
<td>is</td>
<td>ist</td>
<td>598012</td>
</tr>
<tr>
<td>this week</td>
<td>diese Woche</td>
<td>…</td>
</tr>
</tbody>
</table>

This requires phrase alignment
### Word alignment

<table>
<thead>
<tr>
<th>Deutsche</th>
<th>Woche</th>
<th>ist</th>
<th>die</th>
<th>grüne</th>
<th>Hexe</th>
<th>zuhause</th>
</tr>
</thead>
<tbody>
<tr>
<td>The</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>green</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>witch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>is</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>at</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>this</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>week</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- "Diese Woche ist die grüne Hexe zuhause" translates to "This green witch is at home this week."
### Phrase alignment

<table>
<thead>
<tr>
<th></th>
<th>Diese</th>
<th>Woche</th>
<th>ist</th>
<th>die</th>
<th>grüne</th>
<th>Hexe</th>
<th>zuhause</th>
</tr>
</thead>
<tbody>
<tr>
<td>The</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>green</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>witch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>is</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>at</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>home</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>this</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>week</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Obtaining phrase alignments

We’ll skip over details, but here’s the basic idea:

For a given parallel corpus (F—E)
1. Train **two word aligners**, (F→E and E→F)
2. Take the **intersection** of these alignments
to get a **high-precision** word alignment
3. **Grow** these high-precision alignments
   until all words in both sentences are included in the alignment.
   Consider any pair of words in the **union** of the alignments, and incrementally add them to the existing alignments
4. Consider all phrases that are **consistent** with this improved word alignment
Part 4: Decoding (for phrase-based MT)
Phrase-based models of $P(f | e)$

Split target sentence $e = e_1 ... e_n$ into phrases $e_{p1} ... e_{pN}$:

*The green witch* [is] *at home* [this week]

Translate each target phrase $e_{pi}$ into source phrase $f_{pi}$ with **translation probability** $P(f_{pi} | e_{pi})$:

*The green witch* = *die grüne Hexe*, ...

Arrange the set of source phrases $\{f_{pi}\}$ to get $s$ with **distortion probability** $P(f | \{f_{pi}\})$:

*Diese Woche* [ist] *die grüne Hexe* [zuhause]

$$P(f | e = \langle e_{p1}, ..., e_{pl} \rangle) = \prod_i P(f_{pi} | e_{pi}) P(f | \{f_{pi}\})$$
Translating

How do we translate a foreign sentence (e.g. “Diese Woche ist die grüne Hexe zuhause”) into English?

- We need to find $\hat{e} = \arg\max_e P(f \mid e)P(e)$
- There is an exponential number of candidate translations $e$
- But we can look up phrase translations $ep$ and $P(fp \mid ep)$ in the phrase table:

<table>
<thead>
<tr>
<th>diese</th>
<th>Woche</th>
<th>ist</th>
<th>die</th>
<th>grüne</th>
<th>Hexe</th>
<th>zuhause</th>
</tr>
</thead>
<tbody>
<tr>
<td>this 0.2</td>
<td>week 0.7</td>
<td>is 0.8</td>
<td>the 0.3</td>
<td>green 0.3</td>
<td>witch 0.5</td>
<td>home 1.00</td>
</tr>
<tr>
<td>these 0.5</td>
<td></td>
<td></td>
<td>the green 0.4</td>
<td>sorceress 0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>this week 0.6</td>
<td></td>
<td></td>
<td>green witch 0.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>is this week 0.4</td>
<td></td>
<td></td>
<td>the green witch 0.7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Generating a (random) translation

1. Pick the first Target phrase $ep_1$ from the candidate list.
   
   \[
P := P_{LM}(<s> ep_1)P_{Trans}(fp_1 | ep_1)
   \]
   
   $E = \text{the, } F = <\ldots\text{die}\ldots>$

2. Pick the next target phrase $ep_2$ from the candidate list
   
   \[
P := P \times P_{LM}(ep_2 | ep_1)P_{Trans}(fp_2 | ep_2)
   \]
   
   $E = \text{the green witch, } F = <\ldots\text{die grüne Hexe}\ldots>$

3. Keep going: pick target phrases $ep_i$ until the entire source sentence is translated
   
   \[
P := P \times P_{LM}(ep_i | ep_{1\ldots i-1})P_{Trans}(fp_i | ep_i)
   \]
   
   $E = \text{the green witch is, } F = <\ldots\text{ist die grüne Hexe}\ldots>$
Finding the best translation

How can we find the best translation efficiently?

There is an exponential number of possible translations.

We will use a heuristic search algorithm

We cannot guarantee to find the best (= highest-scoring) translation, but we’re likely to get close.

We will use a “stack-based” decoder

(If you’ve taken Intro to AI: this is A* (“A-star”) search)

We will score partial translations based on how good we expect the corresponding completed translation to be.

Or, rather: we will score partial translations on how bad we expect the corresponding complete translation to be.

That is, our scores will be costs (high=bad, low=good)
Scoring partial translations

Assign expected costs to partial translations \((E, F)\):

\[
expected\_cost(E,F) = current\_cost(E,F) + future\_cost(E,F)
\]

The current cost is based on the score of the partial translation \((E, F)\)

\(\text{e.g. } current\_cost(E,F) = \log P(E)P(F \mid E)\)

The (estimated) future cost is a lower bound on the actual cost of completing the partial translation \((E, F)\):

\[
true\_cost(E,F) = current\_cost(E,F) + actual\_future\_cost(E,F)
\]

\[\geq expected\_cost(E,F) = current\_cost(E,F) + est\_future\_cost(E,F)\]

because \(actual\_future\_cost(E,F) \geq est\_future\_cost(E,F)\)

(The estimated future cost ignores the distortion cost)
Stack-based decoding

Maintain a priority queue (=’stack’) of partial translations (hypotheses) with their expected costs.
Each element on the stack is open (we haven’t yet pursued this hypothesis) or closed (we have already pursued this hypothesis).

At each step:

— **Expand** the best open hypothesis (the open translation with the lowest expected cost) in all possible ways.
— These new translations become new open elements on the stack.
— **Close** the best open hypothesis.

**Additional Pruning** \((n\text{-best / beam search})\):
Only keep the \(n\) best open hypotheses around
Stack-based decoding

- Current translation: these
cost: 852

- Current translation: the
cost: 500

- Current translation: at home
cost: 993

Question: Which words in F have we covered?
Stack-based decoding

We’re done with this node now (all continuations have a lower cost)
Stack-based decoding

Expand one of these new yellow nodes next
Stack-based decoding

Expand the yellow node with the lowest cost

- E: these
  F: d******
  Cost: 852
- E: the
  F: ***d***
  Cost: 500
- E: at home
  F: ******z
  Cost: 993
- E: the witch
  F: ***d*H*
  Cost: 700
- E: the green witch
  F: ***dgH*
  Cost: 560
- E: the at home
  F: ***d*H*
  Cost: 983
Stack-based decoding

Expand the next node with the lowest cost