Lexical Translation

• How do we translate a word? Look it up in the dictionary

  \textit{Haus: house, home, shell, household}

• Multiple translations
  • Different word senses, different registers, different inflections (?)
  • \textit{house, home} are common
  • \textit{shell} is specialized (the Haus of a snail is a shell)
How common is each?

<table>
<thead>
<tr>
<th>Translation</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>house</td>
<td>5000</td>
</tr>
<tr>
<td>home</td>
<td>2000</td>
</tr>
<tr>
<td>shell</td>
<td>100</td>
</tr>
<tr>
<td>household</td>
<td>80</td>
</tr>
</tbody>
</table>
\[ \hat{p}_{\text{MLE}}(e \mid \text{Haus}) = \begin{cases} 
0.696 & \text{if } e = \text{house} \\
0.279 & \text{if } e = \text{home} \\
0.014 & \text{if } e = \text{shell} \\
0.011 & \text{if } e = \text{household} \\
0 & \text{otherwise} 
\end{cases} \]
Lexical Translation

• Goal: a model \( p(e \mid f, m) \)

• where \( e \) and \( f \) are complete English and Foreign sentences

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

- Goal: a model \( p(e \mid f, m) \)

- where \( e \) and \( f \) are complete English and Foreign sentences

- Lexical translation makes the following assumptions:
  - Each word in \( e_i \) in \( e \) is generated from exactly one word in \( f \)
  - Thus, we have an alignment \( a_i \) that indicates which word \( e_i \) “came from”, specifically it came from \( f_{a_i} \).
  - Given the alignments \( a \), translation decisions are conditionally independent of each other and depend only on the aligned source word \( f_{a_i} \).
Putting our assumptions together, we have:

\[
p(e \mid f, m) = \sum_{a \in [0, n]^m} p(a \mid f, m) \times \prod_{i=1}^{m} p(e_i \mid f_{a_i})
\]
Lexical Translation

\[ p(e_i | f a_i) \]

\[ p(\text{house} | \text{Haus}) \quad p(\text{shell} | \text{Haus}) \]

\[ p(\text{declaration} | \text{Unabhaenigkeitserkaerung}) \]

Remember bigram models...
Lexical Translation

• Putting our assumptions together, we have:

\[
p(e \mid f, m) = \sum_{a \in [0, n]^m} p(a \mid f, m) \times \prod_{i=1}^{m} p(e_i \mid f_{a_i})
\]
Alignment

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

Most of the action for the first 10 years of MT was here. Words weren’t the problem, word order was hard.
Alignment

- Alignments can be visualized by drawing links between two sentences, and they are represented as vectors of positions:

\[ \mathbf{a} = (1, 2, 3, 4) \]
Reordering

• Words may be reordered during translation.

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

- A source word may not be translated at all

\[ a = (2, 3, 4) \]
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.

\[
a = (1, 2, 3, 0, 4)^\top
\]
One-to-many Translation

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

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

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

\[ a = ??? \quad a = (1, 2, (3, 4)^T)^T \]
IBM Model 1

• Simplest possible lexical translation model
• Additional assumptions
  • The $m$ alignment decisions are independent
  • The alignment distribution for each $a_i$ is uniform over all source words and NULL

\[
\begin{align*}
  &\text{for each } i \in [1, 2, \ldots, m] \\
  &a_i \sim \text{Uniform}(0, 1, 2, \ldots, n) \\
  &e_i \sim \text{Categorical}(\theta_{f_{a_i}})
\end{align*}
\]
IBM Model I

for each \( i \in [1, 2, \ldots, m] \)

\[ a_i \sim \text{Uniform}(0, 1, 2, \ldots, n) \]

\[ e_i \sim \text{Categorical}(\boldsymbol{\theta}_{f_{a_i}}) \]

\[
p(e, a \mid f, m) = \prod_{i=1}^{m}
\]
IBM Model I

for each $i \in [1, 2, \ldots, m]$

$\begin{align*}
    a_i &\sim \text{Uniform}(0, 1, 2, \ldots, n) \\
e_i &\sim \text{Categorical}(\theta_{f_{a_i}})
\end{align*}$

$p(e, a \mid f, m) = \prod_{i=1}^{m} \frac{1}{1 + n}$
IBM Model I

for each $i \in [1, 2, \ldots, m]$

$\begin{align*}
a_i &\sim \text{Uniform}(0, 1, 2, \ldots, n) \\
e_i &\sim \text{Categorical}(\theta_{f_{a_i}})
\end{align*}$

$p(e, a | f, m) = \prod_{i=1}^{m} \frac{1}{1 + n} p(e_i | f_{a_i})$
IBM Model 1

for each $i \in [1, 2, \ldots, m]$

$$a_i \sim \text{Uniform}(0, 1, 2, \ldots, n)$$

$$e_i \sim \text{Categorical}(\theta_{f_{a_i}})$$

$$p(e, a \mid f, m) = \prod_{i=1}^{m} \frac{1}{1 + n} p(e_i \mid f_{a_i})$$

$$p(e_i, a_i \mid f, m) = \frac{1}{1 + n} p(e_i \mid f_{a_i})$$

$$p(e, a \mid f, m) = \prod_{i=1}^{m} p(e_i, a_i \mid f, m)$$
Marginal probability

\[ p(e_i, a_i | f, m) = \frac{1}{1 + n} p(e_i | f_{a_i}) \]

\[ p(e_i | f, m) = \sum_{a_i=0}^{n} \frac{1}{1 + n} p(e_i | f_{a_i}) \]

Recall our independence assumption: all alignment decisions are independent of each other, and given alignments all translation decisions are independent of each other, so all translation decisions are independent of each other.

\[ p(a, b, c, d) = p(a)p(b)p(c)p(d) \]

\[ p(e | f, m) = \prod_{i=1}^{m} p(e_i | f, m) \]
Marginal probability

\[
p(e_i, a_i \mid f, m) = \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e_i \mid f, m) = \sum_{a_i = 0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e \mid f, m) = \prod_{i=1}^{m} p(e_i \mid f, m)
\]
Marginal probability

\[
p(e_i, a_i \mid f, m) = \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e_i \mid f, m) = \sum_{a_i = 0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e \mid f, m) = \prod_{i=1}^{m} p(e_i \mid f, m)
\]

\[
= \prod_{i=1}^{m} \sum_{a_i = 0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]
Marginal probability

\[
p(e_i, a_i \mid f, m) = \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e_i \mid f, m) = \sum_{a_i=0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
p(e \mid f, m) = \prod_{i=1}^{m} p(e_i \mid f, m)
\]

\[
= \prod_{i=1}^{m} \sum_{a_i=0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i})
\]

\[
= \frac{1}{(1 + n)^m} \prod_{i=1}^{m} \sum_{a_i=0}^{n} p(e_i \mid f_{a_i})
\]
Example

Start with a foreign sentence and a target length.
Example

das Haus ist klein

the house is small
Example

das Haus ist klein

das Haus ist klein

house is small

the house is small
Finding the Viterbi Alignment

\[ a^* = \arg \max_{a \in [0,1,\ldots,n]^m} p(a \mid e, f) \]

\[ = \arg \max_{a \in [0,1,\ldots,n]^m} \frac{p(e, a \mid f)}{\sum_{a'} p(e, a' \mid f)} \]

\[ = \arg \max_{a \in [0,1,\ldots,n]^m} p(e, a \mid f) \]

\[ a_i^* = \arg \max_{a_i=0}^{n} \frac{1}{1 + n} p(e_i \mid f_{a_i}) \]

\[ = \arg \max_{a_i=0}^{n} p(e_i \mid f_{a_i}) \]
Finding the Viterbi Alignment

NULL 1 das 2 Haus 3 ist 4 klein

the 1 home 2 is 3 little 4
Finding the Viterbi Alignment
Finding the Viterbi Alignment

null 0

das 1

Haus 2

ist 3

klein 4

the 1

home 2

is 3

little 4
Finding the Viterbi Alignment

Finding the Viterbi Alignment
Learning Lexical Translation Models

• How do we learn the parameters $p(e \mid f)$

• “Chicken and egg” problem
  
  • If we had the alignments, we could estimate the parameters (MLE)
  
  • If we had parameters, we could find the most likely alignments
EM Algorithm

- pick some random (or uniform) parameters
- Repeat until you get bored (~ 5 iterations for lexical translation models)
  - using your current parameters, compute “expected” alignments for every target word token in the training data
    \[ p(a_i \mid e, f) \] (on board)
  - keep track of the expected number of times \( f \) translates into \( e \) throughout the whole corpus
  - keep track of the expected number of times that \( f \) is used as the source of any translation
  - use these expected counts as if they were “real” counts in the standard MLE equation
EM for Model I

... la maison ... la maison blue ... la fleur ...

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

- Initial step: all alignments equally likely

- Model learns that, e.g., la is often aligned with the
EM for Model I

... la maison ... la maison blue ... la fleur ... 

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

- After one iteration

- Alignments, e.g., between la and the are more likely
EM for Model 1

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

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

- After another iteration

- It becomes apparent that alignments, e.g., between fleur and flower are more likely (pigeon hole principle)
EM for Model 1

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

/ \ / \ / \ / \  

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

- Convergence

- Inherent hidden structure revealed by EM
EM for Model 1

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

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

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

- Parameter estimation from the aligned corpus
## Convergence

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>initial</td>
<td>1st it.</td>
<td>2nd it.</td>
<td>3rd it.</td>
<td>...</td>
</tr>
<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>
</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>
</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>
</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>
</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>
</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>
</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>
</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>
</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>
</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>
</tr>
</tbody>
</table>
Evaluation

- Since we have a probabilistic model, we can evaluate perplexity.

\[
PPL = 2^{-\frac{1}{|\mathcal{D}|} \sum_{(e,f) \in \mathcal{D}} \log \prod_{(e,f) \in \mathcal{D}} p(e|f)}
\]

<table>
<thead>
<tr>
<th></th>
<th>Iter 1</th>
<th>Iter 2</th>
<th>Iter 3</th>
<th>Iter 4</th>
<th>...</th>
<th>Iter ∞</th>
</tr>
</thead>
<tbody>
<tr>
<td>-log likelihood</td>
<td>-</td>
<td>7.66</td>
<td>7.21</td>
<td>6.84</td>
<td>...</td>
<td>-6</td>
</tr>
<tr>
<td>perplexity</td>
<td>-</td>
<td>2.42</td>
<td>2.3</td>
<td>2.21</td>
<td>...</td>
<td>2</td>
</tr>
</tbody>
</table>
Alignment Error Rate

Possible links \( P \)

Sure links \( S \)

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

Recall\((A, S) = \frac{|S \cap A|}{|S|} \)

\[ \text{AER}(A, P, S) = 1 - \frac{|S \cap A| + |P \cap A|}{|S| + |A|} \]
Announcements

• HW 1 will be posted tonight (due Feb. 18)
HOMEWORK 1

Due 11:59pm on Tuesday, Feb. 12, 2013

Word alignment is a fundamental task in statistical machine translation. This homework will give you an opportunity to try your hand at developing solutions to this challenging and interesting problem.

Getting started

Go to your clone of your course GitHub repository on the machine where you will be doing this assignment, and run the following command to obtain the code and data you will need:

./tools/get-new-assignments

You will obtain a very simple heuristic aligner written in Python and 100,000 German-English parallel sentences from the Europarl corpus, version 7. The heuristic aligner uses set similarity to determine which words are aligned to each other in a corpus of parallel sentences. The intuition is that if you look at the set of sentence pairs that contain an English word $x$, and that set is similar to the set of sentence pairs that contain a German word $y$, then these words are likely to be translations of each other. The set similarity measure we use is Dice's coefficient, defined in terms of sets $X$ and $Y$ as follows:

$$D(X, Y) = \frac{2 \times |X \cap Y|}{|X| + |Y|}$$

Dice's coefficient ranges in value from 0 to 1.

In our formulation, every pair of words $(e, g)$ in the parallel corpus receives a Dice “score” $\delta(e, g)$. The aligner goes through all pairs of sentences and aligns English word $e_i$ to German word $g_j$ if $\delta(e_i, g_j) > \tau$. By making $\tau$ closer to 1, fewer (hopefully, higher precision) points are aligned; by making it closer to 0, more points are aligned. By default, our aligner uses $\tau = 0.5$ as its threshold.

Run the baseline heuristic model 1,000 sentences using the command:

```sh
./align -n 1000 | ./check > dice.al
```