IBM Translation Models

Instructor: Yoav Artzi

Slides adapted from Michael Collins
The Noisy Channel Model

- **Goal:** translate from French to English
- Have a model $p(e|f)$ to estimate the probability of an English sentence $e$ given a French sentence $f$
- Estimate the parameters from training corpus
- A noisy channel model has two components:
  - $p(e)$: the language model
  - $p(f|e)$: the translation model
- Giving:
  
  $$p(e|f) = \frac{p(e, f)}{p(f)} = \frac{p(e)p(f|e)}{\sum_e p(e)p(f|e)}$$

  and

  $$\arg \max_e p(e|f) = \arg \max_e p(e)p(f|e)$$
Overview

- IBM Model 1
- IBM Model 2
- EM Training of Models 1 and 2
IBM Model 1: Alignments

- How do we model \( p(f|e) \)?
- 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 alignent \( a \) is:
  \[
  \{a_1, \ldots, a_m\} \quad \text{where} \quad a_j \in 0 \ldots l
  \]
- There are \((l + 1)^m\) possible alignments.
IBM Model 1: Alignments

\[ l = 6, \ m = 7 \]

\( e = \) And the program has been implemented

\[ f = \) Le programme a été mis en application
IBM Model 1: Alignments

\[ l = 6,\ m = 7 \]

\( e = \) And the program has been implemented

\( f = \) Le programme a été mis en application

- One alignment is

\[ \{2, 3, 4, 5, 6, 6, 6, 6\} \]
IBM Model 1: Alignments

\[ l = 6, \ m = 7 \]

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

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

- Another (bad!) alignment is

\[ \{1, 1, 1, 1, 1, 1, 1, 1\} \]
IBM Model 1: Alignments

\[ l = 6, \ m = 7 \]
\[ e = \text{And the program has been implemented} \]
\[ f = \text{Le programme a été mis en application} \]

*Another (bad!) alignment is*

\[ \{1, 1, 1, 1, 1, 1, 1, 1\} \]
Alignments in the IBM Models

• We define two models:

\[ p(a|e, m) \quad p(f|a, e, m) \]

• Giving:

\[ p(f, a|e, m) = p(a|e, m)p(f|a, e, m) \]

• Also:

\[ p(f|e, m) = \sum_{a \in A} p(a|e, m)p(f|a, e, m) \]

where \( A \) is a set of all possible alignments
Most Likely Alignments

\[ p(f, a|e, m) = p(a|e, m)p(f|a, e, m) \]

- We can also calculate:

\[ p(a|f, e, m) = \frac{p(f, a|e, m)}{\sum_{a \in A} p(f, a|e, m)} \]

for any alignment \( a \)

- For a given \( f, e \) pair, can also compute the most likely alignment (details in notes)

- The original IBM models are rarely used for translation, but still key for recovering alignments
Example Alignment

- **French:**
  le conseil a rendu son avis, et nous devons à présent adopter un nouvel avis sur la base de la première position.

- **English:**
  the council has stated its position, and now, on the basis of the first position, we again have to give our opinion.

- **Alignment:**
  the/le council/conseil has/à stated/rendu its/son position/avis ,/, and/et now/présent ,/NULL on/sur the/le basis/base of/de the/la first/première position/position ,/NULL we/nous again/NULL have/devons to/a give/adopter our/nouvel opinion/avis ./.
IBM Model 1: Alignments

• In IBM Model 1 all alignments \( a \) are equally likely:

\[
p(a|e, m) = \frac{1}{(1 + l)^m}
\]

• Reasonable assumption?
  – Simplifying assumption, but it gets things started …
IBM Model 1: Translation Probabilities

- Next step: come up with an estimate for $p(f|a,e,m)$

- In Model 1, this is:

$$p(f|a,e,m) = \prod_{j=1}^{m} t(f_j|e_{a_j})$$
IBM Model 1: Example

\[ l = 6, \quad m = 7 \]

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

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

\[ a = \{2, 3, 4, 5, 6, 6, 6\} \]
IBM Model 1: Example

| p(f|e)     | And | the | program | has    | been | implemented |
|-----------|-----|-----|---------|--------|-------|-------------|
| Le        | 0.2 | 0.6 | 0.1     | 0.025  | 0.05  | 0.025       |
| programme | 0.05| 0.2 | 0.45    | 0.1    | 0.1   | 0.1         |
| a         | 0.1 | 0.1 | 0.15    | 0.2    | 0.15  | 0.3         |
| ete       | 0.05| 0.05| 0.05    | 0.05   | 0.7   | 0.1         |
| mis       | 0.2 | 0.05| 0.05    | 0.05   | 0.25  | 0.4         |
| en        | 0.25| 0.1 | 0.25    | 0.25   | 0.1   | 0.05        |
| application | 0.01| 0.03| 0.01    | 0.02   | 0.03  | 0.9         |
IBM Model 1: Example

\[ l = 6, m = 7 \]
\[ e = \text{And the program has been implemented} \]
\[ f = \text{Le programme a ete mis en application} \]

\[ a = \{2, 3, 4, 5, 6, 6, 6\} \]

\[
p(f|a, e) = t(\text{Le}|\text{the}) \times t(\text{programme}|\text{program}) \times t(\text{a}|\text{has}) \times t(\text{ete}|\text{been}) \times t(\text{mis}|\text{implemented}) \times t(\text{en}|\text{implemented}) \times t(\text{application}|\text{implemented}) = 0.0006804
\]

\[
p(f, a \mid e, 7) = 8.26186E - 10
\]
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

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

The final result:

$$p(f, a|e, m) = p(a|e, m) \times p(f|a, e, m) = \frac{1}{(1 + l)^m} \prod_{j=1}^{m} t(f_j|e_{a_j})$$
... de la situation au niveau des négociations de l’ompi ...
... of the current position in the wipo negotiations ... 

nous ne sommes pas en mesure de décider, ...
we are not in position to decide ...

... Le point de vue de la commission face à ce problème complexe .
... the commission ‘s position on this complex problem .
Overview

• IBM Model 1
• IBM Model 2
• EM Training of Models 1 and 2
IBM Model 2

• Only difference: we now introduce alignment distortion parameters
  \[q(i | j, l, m)\]

• Probability that \( j \)'th French word is connected to \( i \)'th English word, given sentence length of \( e \) and \( f \) are \( l \) and \( m \)

• Define
  \[p(a | e, m) = \prod_{j=1}^{m} q(a_j | j, l, m)\]

  where \( a = \{a_1, \ldots, a_m\} \)

• Gives
  \[p(f, a | e, m) = \prod_{j=1}^{m} q(a_j | j, l, m) t(f_j | e_{a_j})\]
Example

\[ l = 6 \]
\[ m = 7 \]
\[ e = \text{And the program has been implemented} \]
\[ f = \text{Le programme a ete mis en application} \]
\[ a = \{2, 3, 4, 5, 6, 6, 6\} \]
Example

\[ l = 6 \]
\[ m = 7 \]
\[ e = \text{And the program has been implemented} \]
\[ f = \text{Le programme a ete mis en application} \]
\[ a = \{2, 3, 4, 5, 6, 6, 6\} \]

\[ p(a \mid e, 7) = q(2 \mid 1, 6, 7) \times q(3 \mid 2, 6, 7) \times q(4 \mid 3, 6, 7) \times q(5 \mid 4, 6, 7) \times q(6 \mid 5, 6, 7) \times q(6 \mid 6, 6, 7) \times q(6 \mid 7, 6, 7) \]
Example

\[
\begin{align*}
l &= 6 \\
m &= 7 \\
e &= \text{And the program has been implemented} \\
f &= \text{Le programme a été mis en application} \\
a &= \{2, 3, 4, 5, 6, 6, 6\} \\
p(f \mid a, e, 7) &= \mathbf{t}(Le \mid \text{the}) \times \\
&\quad \mathbf{t}(\text{programme} \mid \text{program}) \times \\
&\quad \mathbf{t}(a \mid \text{has}) \times \\
&\quad \mathbf{t}(ete \mid \text{been}) \times \\
&\quad \mathbf{t}(mis \mid \text{implemented}) \times \\
&\quad \mathbf{t}(en \mid \text{implemented}) \times \\
&\quad \mathbf{t}(\text{application} \mid \text{implemented})
\end{align*}
\]
IBM Model 2: The Generative Process

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

- **Step 1**: Pick an alignment $a = \{a_1, \ldots, a_m\}$ with probability
  
  $p(a|e, m) = \prod_{j=1}^{m} q(a_j|j, l, m)$

- **Step 2**: Pick the French words with probability

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

The final result:

$p(f, a|e, m) = p(a|e, m) \times p(f|a, e, m) = \prod_{j=1}^{m} q(a_j|j, l, m)t(f_j|e_{a_j})$
Recovering Alignments

- If we have parameters $q$ and $t$, we can easily recover the most likely alignment for any sentence pair.

Given a sentence pair $e_1, e_2, \ldots, e_l, f_1, f_2, \ldots, f_m$ define

$$a_j = \arg \max_{a \in \{0 \ldots l\}} q(a | j, l, m) \times t(f_j, e_a)$$

for $j = 1 \ldots m$

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

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

• IBM Model 1
• IBM Model 2
• EM Training of Models 1 and 2
The Parameter Estimation Problem

• Input:
  \[(e^{(k)}, f^{(k)}), k = 1 \ldots n\]
  Each \(e^{(k)}\) is an English sentence, each \(f^{(k)}\) is a French sentence

• Output: parameters for
  \[t(f|e) \quad q(i|j, l, m)\]

• A key challenge: we do not have alignments in our training examples

\(e^{(100)} = \) And the program has been implemented

\(f^{(100)} = \) Le programme a ete mis en application
Parameter Estimation if Alignments are Observed

- Assume alignments are observed in training data
  \( e^{(100)} \) = And the program has been implemented

  \( f^{(100)} \) = Le programme a ete mis en application
  \( a^{(100)} \) = \(<2,3,4,5,6,6,6>\)

- Training data is
  \( (e^{(k)}, f^{(k)}, a^{(k)}), k = 1 \ldots n \)
  
  Each \( e^{(k)} \) is an English sentence, each \( f^{(k)} \) is a French sentence, each \( a^{(k)} \) is an alignment

- Maximum-likelihood parameter estimates are trivial:

\[
t_{ML}(f|e) = \frac{\text{count}(e, f)}{\text{count}(e)} \quad q_{ML}(j|i, l, m) = \frac{\text{count}(j, i, l, m)}{\text{count}(i, l, m)}
\]
Input: A training corpus \((f^{(k)}, e^{(k)}, a^{(k)})\) for \(k = 1 \ldots n\), where
\[ f^{(k)} = f_1^{(k)} \ldots f_{m_k}^{(k)}, \quad e^{(k)} = e_1^{(k)} \ldots e_{l_k}^{(k)}, \quad a^{(k)} = a_1^{(k)} \ldots a_{m_k}^{(k)}. \]

Algorithm:

- Set all counts \(c(\ldots) = 0\)
- For \(k = 1 \ldots n\)
  - For \(i = 1 \ldots m_k\), For \(j = 0 \ldots l_k\),
    \[
    c(e_j^{(k)}, f_i^{(k)}) \leftarrow c(e_j^{(k)}, f_i^{(k)}) + \delta(k, i, j)
    \]
    \[
    c(e_j^{(k)}) \leftarrow c(e_j^{(k)}) + \delta(k, i, j)
    \]
    \[
    c(j|i, l, m) \leftarrow c(j|i, l, m) + \delta(k, i, j)
    \]
    \[
    c(i, l, m) \leftarrow c(i, l, m) + \delta(k, i, j)
    \]

where \(\delta(k, i, j) = 1\) if \(a_i^{(k)} = j\), 0 otherwise.

Output: \(t_{ML}(f|e) = \frac{c(e,f)}{c(e)}\), \(q_{ML}(j|i, l, m) = \frac{c(j|i,l,m)}{c(i,l,m)}\).
Parameter Estimation with the EM Algorithm

- Input: \((e^{(k)}, f^{(k)}), k = 1 \ldots n\)
  
  Each \(e^{(k)}\) is an English sentence, each \(f^{(k)}\) is a French sentence.

- The algorithm is related to algorithm with observed alignments, but with two key differences:
  - Iterative: start with initial (e.g., random) choice of \(q\) and \(t\) parameters, at each iteration: compute some “counts” base on data and parameters, and re-estimate parameters.
  - The definition of the delta function is different:

\[
\delta(k, i, j) = \frac{q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}{\sum_{j=0}^{l_k} q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}
\]
**Input:** A training corpus \((f^{(k)}, e^{(k)})\) for \(k = 1 \ldots n\), where 

\[ f^{(k)} = f_1^{(k)} \ldots f_{m_k}^{(k)}, \quad e^{(k)} = e_1^{(k)} \ldots e_{l_k}^{(k)}. \]

**Initialization:** Initialize \(t(f|e)\) and \(q(j|i, l, m)\) parameters (e.g., to random values).
For $s = 1 \ldots S$

- Set all counts $c(\ldots) = 0$
- For $k = 1 \ldots n$
  - For $i = 1 \ldots m_k$, For $j = 0 \ldots l_k$

$$c(e_j^{(k)}, f_i^{(k)}) \leftarrow c(e_j^{(k)}, f_i^{(k)}) + \delta(k, i, j)$$

$$c(e_j^{(k)}) \leftarrow c(e_j^{(k)}) + \delta(k, i, j)$$

$$c(j|i, l, m) \leftarrow c(j|i, l, m) + \delta(k, i, j)$$

$$c(i, l, m) \leftarrow c(i, l, m) + \delta(k, i, j)$$

where

$$\delta(k, i, j) = \frac{q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}{\sum_{j=0}^{l_k} q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}$$

- Recalculate the parameters:

$$t(f|e) = \frac{c(e, f)}{c(e)} \quad q(j|i, l, m) = \frac{c(j|i, l, m)}{c(i, l, m)}$$
\[
\delta(k, i, j) = \frac{q(j | i, l_k, m_k) t(f_i^{(k)} | e_j^{(k)})}{\sum_{j=0}^{l_k} q(j | i, l_k, m_k) t(f_i^{(k)} | e_j^{(k)})}
\]

\(e^{(100)} = \) And the program has been implemented

\(f^{(100)} = \) Le programme a ete mis en application
For \( s = 1 \ldots S \)

- Set all counts \( c(\ldots) = 0 \)
- For \( k = 1 \ldots n \)
  - For \( i = 1 \ldots m_k \), For \( j = 0 \ldots l_k \)

\[
c(e_j^{(k)}, f_i^{(k)}) \leftarrow c(e_j^{(k)}, f_i^{(k)}) + \delta(k, i, j)
\]

\[
c(e_j^{(k)}) \leftarrow c(e_j^{(k)}) + \delta(k, i, j)
\]

\[
c(j|i, l, m) \leftarrow c(j|i, l, m) + \delta(k, i, j)
\]

\[
c(i, l, m) \leftarrow c(i, l, m) + \delta(k, i, j)
\]

where

\[
\delta(k, i, j) = \frac{q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}{\sum_{j=0}^{l_k} q(j|i, l_k, m_k)t(f_i^{(k)}|e_j^{(k)})}
\]

- Recalculate the parameters:

\[
t(f|e) = \frac{c(e, f)}{c(e)} \quad q(j|i, l, m) = \frac{c(j|i, l, m)}{c(i, l, m)}
\]
Justification for the Algorithm

- **Input:**
  \[ (e^{(k)}, f^{(k)}; k = 1 \ldots n) \]
  Each \( e^{(k)} \) is an English sentence, each \( f^{(k)} \) is a French sentence
- **The log-likelihood function:**
  \[
  L(t, q) = \sum_{k=1}^{n} \log p(f^{(k)}|e^{(k)}) = \sum_{k=1}^{n} \log \sum_{a} p(f^{(k)}, a|e^{(k)})
  \]
- **The maximum-likelihood estimates are:**
  \[
  \arg \max_{t, q} L(t, q)
  \]
- **The EM algorithm will converge to a local maximum of the log-likelihood function**
Summary

- Key ideas in the IBM translation models:
  - Alignment variables
  - Translation parameters, e.g., \( t(\text{chien}|\text{dog}) \)
  - Distortion parameters, e.g., \( q(2|1,6,7) \)
- The EM algorithm: an iterative algorithm for training the \( q \) and \( t \) parameters
- Once parameters are trained, can recover the most likely alignment on our training examples

\[
e^{(100)} = \text{And the program has been implemented}
\]
\[
f^{(100)} = \text{Le programme a été mis en application}
\]