SFU NatLangLab

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

October 28, 2014

Natural Language Processing

Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

Part 1: Generative Models for Word Alignment

Statistical Machine Translation

Noisy Channel Model

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} \underbrace{\Pr(\mathbf{e})}_{\textbf{Language Model}} \cdot \underbrace{\Pr(\mathbf{f} \mid \mathbf{e})}_{\textbf{Alignment Model}}$$

Alignment Task

$$\begin{matrix} \mathbf{e} \searrow & & & \\ & \text{Program} & \longrightarrow & \Pr(\mathbf{e} \mid \mathbf{f}) \\ \mathbf{f} \nearrow & & & \end{matrix}$$

Training Data

- **Alignment Model**: learn a mapping between **f**and **e**.
  Training data: lots of translation pairs between **f**and **e**.

Statistical Machine Translation

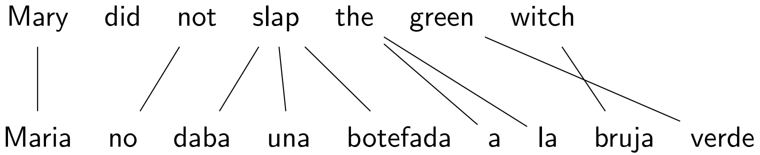
The IBM Models

- The first statistical machine translation models were developed at IBM Research (Yorktown Heights, NY) in the 1980s
- The models were published in 1993:
  Brown et. al. The Mathematics of Statistical Machine Translation.
  Computational Linguistics. 1993.
  `http://aclweb.org/anthology/J/J93/J93-2003.pdf`
- These models are the basic SMT models, called:
  IBM Model 1, IBM Model 2, IBM Model 3, IBM Model 4, IBM Model 5
  as they were called in the 1993 paper.
- We use **e**and **f** in the equations in honor of their system which translated from French to English.
  Trained on the Canadian Hansards (Parliament Proceedings)

Generative Model of Word Alignment

- English **e**: Mary did not slap the green witch
- "French" **f**: Maria no daba una botefada a la bruja verde
- Alignment **a**: $\{1, 3, 4, 4, 4, 5, 5, 7, 6\}$
  e.g. $(f_8, e_{a_8}) = (f_8, e_7) =$ (bruja, witch)

Visualizing alignment **a**



Generative Model of Word Alignment

Data Set

- Data set $\mathcal{D}$ of N sentences:
$$\mathcal{D} = \{(\mathbf{f}^{(1)}, \mathbf{e}^{(1)}), \ldots, (\mathbf{f}^{(N)}, \mathbf{e}^{(N)})\}$$
- French $\mathbf{f}$: $(f_1, f_2, \ldots, f_I)$
- English $\mathbf{e}$: $(e_1, e_2, \ldots, e_J)$
- Alignment $\mathbf{a}$: $(a_1, a_2, \ldots, a_I)$
- $\text{length}(\mathbf{f}) = \text{length}(\mathbf{a}) = I$

Generative Model of Word Alignment

Find the best alignment for each translation pair

$$\mathbf{a}^* = \arg\max_{\mathbf{a}} \Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e})$$

Alignment probability

$$\begin{aligned}
\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) &= \frac{\Pr(\mathbf{f}, \mathbf{a}, \mathbf{e})}{\Pr(\mathbf{f}, \mathbf{e})} \\
&= \frac{\Pr(\mathbf{e}) \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{e}) \Pr(\mathbf{f} \mid \mathbf{e})} \\
&= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} \\
&= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}
\end{aligned}$$

Word Alignments: IBM Model 3

Generative "story" for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$

Word Alignments: IBM Model 3

Fertility parameter

$$n(\phi_j \mid e_j) : n(3 \mid slap); n(0 \mid did)$$

Translation parameter

$$t(f_i \mid e_{a_i}) : t(bruja \mid witch)$$

Distortion parameter

$$d(f_{pos} = i \mid e_{pos} = j, I, J) : d(8 \mid 7, 8, 6)$$

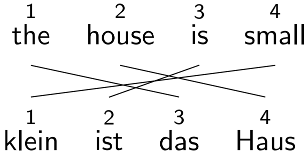
Word Alignments: IBM Model 3

Generative model for $P(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$

$$
\begin{aligned}
P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) &= \prod_{i=1}^{I} n(\phi_{a_i} \mid e_{a_i}) \\
&\times \; t(f_i \mid e_{a_i}) \\
&\times \; d(i \mid a_i, I, J)
\end{aligned}
$$

Word Alignments: IBM Model 3

Sentence pair with alignment $\mathbf{a} = (4, 3, 1, 2)$

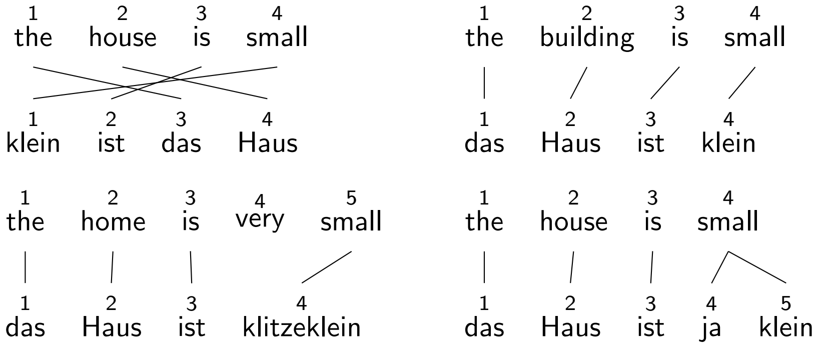


If we know the parameter values we can easily compute the probability of this aligned sentence pair.

$\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) =$

$$\begin{array}{lclcl}
n(1 \mid \text{the}) & \times & t(\text{das} \mid \text{the}) & \times & d(3 \mid 1, 4, 4) \times \\
n(1 \mid \text{house}) & \times & t(\text{Haus} \mid \text{house}) & \times & d(4 \mid 2, 4, 4) \times \\
n(1 \mid \text{is}) & \times & t(\text{ist} \mid \text{is}) & \times & d(2 \mid 3, 4, 4) \times \\
n(1 \mid \text{small}) & \times & t(\text{klein} \mid \text{small}) & \times & d(1 \mid 4, 4, 4)
\end{array}$$

Word Alignments: IBM Model 3



Parameter Estimation

- What is $n(1 \mid \text{very}) =$? and $n(0 \mid \text{very}) =$?
- What is $t(\text{Haus} \mid \text{house}) =$? and $t(\text{klein} \mid \text{small}) =$?
- What is $d(1 \mid 4, 4, 4) =$? and $d(1 \mid 1, 4, 4) =$?

Word Alignments: IBM Model 3

1	2	3	4
the	house	is	small
1	2	3	4
klein	ist	das	Haus

1	2	3	4
the	building	is	small
1	2	3	4
das	Haus	ist	klein

1	2	3	4	5
the	home	is	very	small
1	2	3	4	
das	Haus	ist	klitzeklein	

1	2	3	4	5
the	house	is	small	
1	2	3	4	5
das	Haus	ist	ja	klein

Parameter Estimation: Sum over all alignments

$$\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \sum_{\mathbf{a}} \prod_{i=1}^{I} n(\phi_{a_i} \mid e_{a_i}) \times t(f_i \mid e_{a_i}) \times d(i \mid a_i, I, J)$$

Word Alignments: IBM Model 3

Summary

- If we know the parameter values we can easily compute the probability $\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e})$ given an aligned sentence pair
- If we are given a corpus of sentence pairs with alignments we can easily learn the parameter values by using relative frequencies.
- If we do not know the alignments then perhaps we can produce all possible alignments each with a certain probability?

IBM Model 3 is too hard: Let us try learning only $t(f_i \mid e_{a_i})$

$$\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \sum_{\mathbf{a}} \prod_{i=1}^{I} n(\phi_{a_i} \mid e_{a_i}) \times t(f_i \mid e_{a_i}) \times d(i \mid a_i, I, J)$$

Word Alignments: IBM Model 1

Alignment probability

$$\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) = \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}$$

Example alignment

1	2	3	4
the	house	is	small
|	|	|	|
1	2	3	4
das	Haus	ist	klein

$$\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \prod_{i=1}^{I} t(f_i \mid e_{a_i})$$

$$\begin{aligned}\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \\ & t(\text{das} \mid \text{the}) \times \\ & t(\text{Haus} \mid \text{house}) \times \\ & t(\text{ist} \mid \text{is}) \times \\ & t(\text{klein} \mid \text{small})\end{aligned}$$

Word Alignments: IBM Model 1

Generative "story" for Model 1

$$\begin{array}{ccccc} \text{the} & \text{house} & \text{is} & \text{small} & \\ \downarrow & \downarrow & \downarrow & \downarrow & \\ \text{das} & \text{Haus} & \text{ist} & \text{klein} & \text{(translate)} \end{array}$$

$$\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \prod_{i=1}^{l} t(f_i \mid e_{a_i})$$

Finding the best word alignment: IBM Model 1

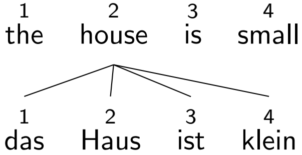
Compute the arg max word alignment

$$\hat{\mathbf{a}} = \arg\max_{\mathbf{a}} \Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f})$$

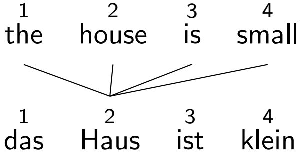
- For each f_i in $(f_1, \ldots, f_I)$ build $\mathbf{a} = (\hat{a}_1, \ldots, \hat{a}_I)$

$$\hat{a}_i = \arg\max_{a_i} t(f_i \mid e_{a_i})$$

Many to one alignment ✓



One to many alignment ✗



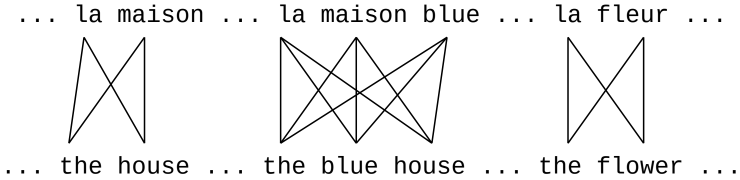
Learning parameters[from P.Koehn SMT book slides]

- 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 [from P.Koehn SMT book slides]

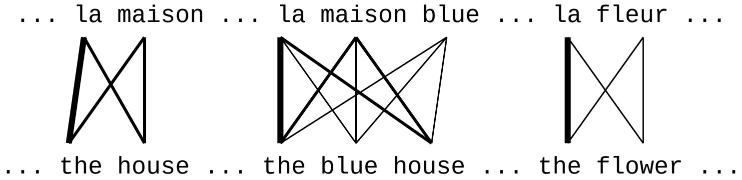
- Incomplete data
  - if we had *complete data*, we could estimate *model*
  - if we had *model*, we could fill in the *gaps in the data*
- Expectation Maximization (EM) in a nutshell
  1. initialize model parameters (e.g. uniform)
  2. assign probabilities to the missing data
  3. estimate model parameters from completed data
  4. iterate steps 2–3 until convergence

EM Algorithm[from P.Koehn SMT book slides]



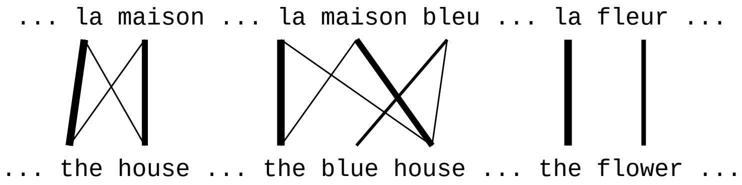
- Initial step: all alignments equally likely
- Model learns that, e.g., *la* is often aligned with *the*

EM Algorithm[from P.Koehn SMT book slides]



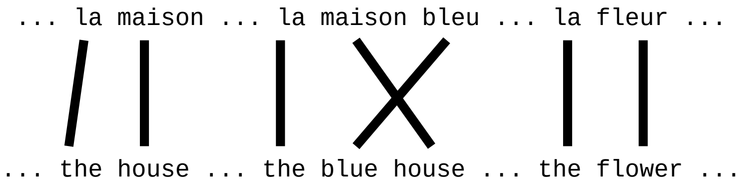
- After one iteration
- Alignments, e.g., between *la* and *the* are more likely

EM Algorithm[from P.Koehn SMT book slides]



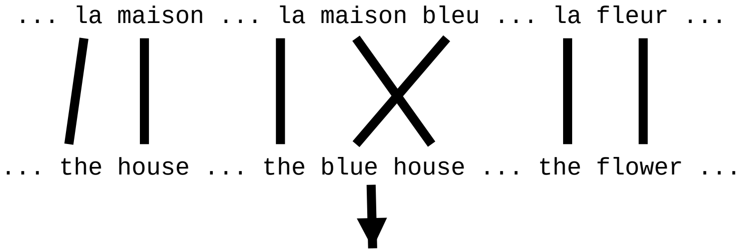
- After another iteration
- It becomes apparent that alignments, e.g., between *fleur* and *flower* are more likely (pigeon hole principle)

EM Algorithm[from P.Koehn SMT book slides]



- Convergence
- Inherent hidden structure revealed by EM

EM Algorithm[from P.Koehn SMT book slides]




```
p(la|the) = 0.453
p(le|the) = 0.334
p(maison|house) = 0.876
p(bleu|blue) = 0.563
...
```

- Parameter estimation from the aligned corpus

IBM Model 1 and the EM Algorithm [from P.Koehn SMT book slides]

- 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 the EM Algorithm [from P.Koehn SMT book slides]

- We need to be able to compute:
  - Expectation-Step: probability of alignments
  - Maximization-Step: count collection

Word Alignments: IBM Model 1

Alignment probability

$$
\begin{aligned}
\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) &= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} \\
&= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})} \\
&= \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i})}{\sum_{\mathbf{a}} \prod_{i=1}^{I} t(f_i \mid e_{a_i})}
\end{aligned}
$$

Computing the denominator

- The denominator above is summing over J^I alignments
- An interlude on how compute the denominator faster ...

Word Alignments: IBM Model 1

Sum over all alignments

$$\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \sum_{a_1=1}^{J} \sum_{a_2=1}^{J} \cdots \sum_{a_I=1}^{J} \prod_{i=1}^{I} t(f_i \mid e_{a_i})$$

Assume (f_1, f_2, f_3) and (e_1, e_2)

$$\sum_{a_1=1}^{2} \sum_{a_2=1}^{2} \sum_{a_3=1}^{2} t(f_1 \mid e_{a_1}) \times t(f_2 \mid e_{a_2}) \times t(f_3 \mid e_{a_3})$$

Word Alignments: IBM Model 1

Assume (f_1, f_2, f_3) and (e_1, e_2): $I = 3$ and $J = 2$

$$\sum_{a_1=1}^{2} \sum_{a_2=1}^{2} \sum_{a_3=1}^{2} t(f_1 \mid e_{a_1}) \times t(f_2 \mid e_{a_2}) \times t(f_3 \mid e_{a_3})$$

$J^I = 2^3$ terms to be added:

$$\begin{array}{lllllll}
t(f_1 \mid e_1) & \times & t(f_2 \mid e_1) & \times & t(f_3 \mid e_1) & + \\
t(f_1 \mid e_1) & \times & t(f_2 \mid e_1) & \times & t(f_3 \mid e_2) & + \\
t(f_1 \mid e_1) & \times & t(f_2 \mid e_2) & \times & t(f_3 \mid e_1) & + \\
t(f_1 \mid e_1) & \times & t(f_2 \mid e_2) & \times & t(f_3 \mid e_2) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_1) & \times & t(f_3 \mid e_1) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_1) & \times & t(f_3 \mid e_2) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_2) & \times & t(f_3 \mid e_1) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_2) & \times & t(f_3 \mid e_2) &
\end{array}$$

Word Alignments: IBM Model 1

Factor the terms:

$$
\begin{array}{lcll}
(t(f_1 \mid e_1) \times t(f_2 \mid e_1)) & \times & (t(f_3 \mid e_1) + t(f_3 \mid e_2)) & + \\
(t(f_1 \mid e_1) \times t(f_2 \mid e_2)) & \times & (t(f_3 \mid e_1) + t(f_3 \mid e_2)) & + \\
(t(f_1 \mid e_2) \times t(f_2 \mid e_1)) & \times & (t(f_3 \mid e_1) + t(f_3 \mid e_2)) & + \\
(t(f_1 \mid e_2) \times t(f_2 \mid e_2)) & \times & (t(f_3 \mid e_1) + t(f_3 \mid e_2)) &
\end{array}
$$

$$
(t(f_3 \mid e_1) + t(f_3 \mid e_2)) \left(\begin{array}{lcll}
t(f_1 \mid e_1) & \times & t(f_2 \mid e_1) & + \\
t(f_1 \mid e_1) & \times & t(f_2 \mid e_2) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_1) & + \\
t(f_1 \mid e_2) & \times & t(f_2 \mid e_2) &
\end{array} \right)
$$

$$
(t(f_3 \mid e_1) + t(f_3 \mid e_2)) \left(\begin{array}{lcll}
t(f_1 \mid e_1) & \times & (t(f_2 \mid e_1) + t(f_2 \mid e_2)) & + \\
t(f_1 \mid e_2) & \times & (t(f_2 \mid e_1) + t(f_2 \mid e_2)) &
\end{array} \right)
$$

Word Alignments: IBM Model 1

Assume (f_1, f_2, f_3) and (e_1, e_2): $I = 3$ and $J = 2$

$$\prod_{i=1}^{3} \sum_{a_i=1}^{2} t(f_i \mid e_{a_i})$$

$I \times J = 2 \times 3$ terms to be added:

$$\begin{array}{ccccc} (t(f_1 \mid e_1) & + & t(f_1 \mid e_2)) & \times \\ (t(f_2 \mid e_1) & + & t(f_2 \mid e_2)) & \times \\ (t(f_3 \mid e_1) & + & t(f_3 \mid e_2)) & \end{array}$$

Word Alignments: IBM Model 1

Alignment probability

$$
\begin{aligned}
\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) &= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} \\
&= \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i})}{\sum_{\mathbf{a}} \prod_{i=1}^{I} t(f_i \mid e_{a_i})} \\
&= \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i})}{\prod_{i=1}^{I} \sum_{j=1}^{J} t(f_i \mid e_j)}
\end{aligned}
$$

Learning Parameters: IBM Model 1

1	2	1	2	1	2
the	house	the	book	a	book
1	2	1	2	1	2
das	Haus	das	Buch	ein	Buch

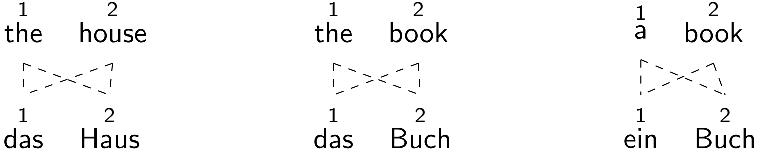
Learning parameters $t(f|e)$ when alignments are known

$$t(das \mid the) = \frac{c(das,the)}{\sum_f c(f,the)} \quad t(house \mid Haus) = \frac{c(Haus,house)}{\sum_f c(f,house)}$$

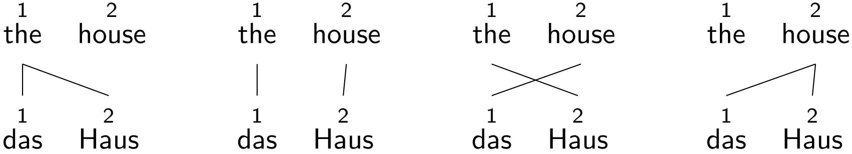
$$t(ein \mid a) = \frac{c(ein,a)}{\sum_f c(f,a)} \quad t(Buch \mid book) = \frac{c(Buch,book)}{\sum_f c(f,book)}$$

$$t(f|e) = \sum_{s=1}^{N} \sum_{f \to e \in \mathbf{f}^{(s)},\mathbf{e}^{(s)}} \frac{c(f,e)}{\sum_f c(f,e)}$$

Learning Parameters: IBM Model 1



Learning parameters $t(f|e)$ when alignments are *unknown*



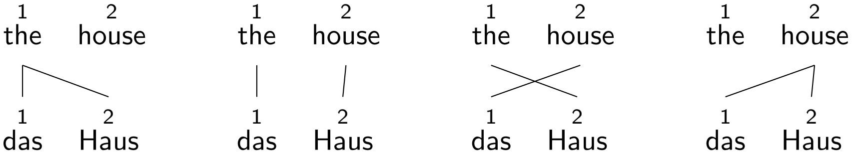
Also list alignments for *(the book, das Buch)* and *(a book, ein Buch)*

Learning Parameters: IBM Model 1

Initialize $t^0(f|e)$

$$
\begin{aligned}
t(\mathit{Haus} \mid \mathit{the}) &= 0.25 \\
t(\mathit{das} \mid \mathit{the}) &= 0.5 \\
t(\mathit{Buch} \mid \mathit{the}) &= 0.25
\end{aligned}
\qquad
\begin{aligned}
t(\mathit{das} \mid \mathit{house}) &= 0.5 \\
t(\mathit{Haus} \mid \mathit{house}) &= 0.5 \\
t(\mathit{Buch} \mid \mathit{house}) &= 0.0
\end{aligned}
$$

Compute posterior for each alignment



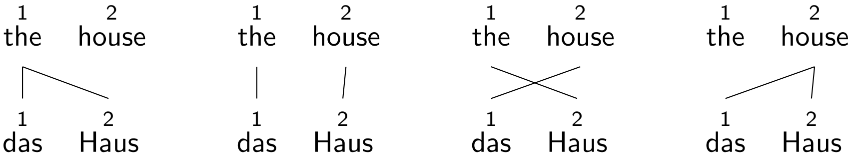
$$\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) = \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} = \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i})}{\prod_{i=1}^{I} \sum_{j=1}^{J} t(f_i \mid e_j)}$$

Learning Parameters: IBM Model 1

Initialize $t^0(f|e)$

$$\begin{aligned} t(Haus \mid the) &= 0.25 \\ t(das \mid the) &= 0.5 \\ t(Buch \mid the) &= 0.25 \end{aligned} \qquad \begin{aligned} t(das \mid house) &= 0.5 \\ t(Haus \mid house) &= 0.5 \\ t(Buch \mid house) &= 0.0 \end{aligned}$$

Compute $\Pr(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ for each alignment

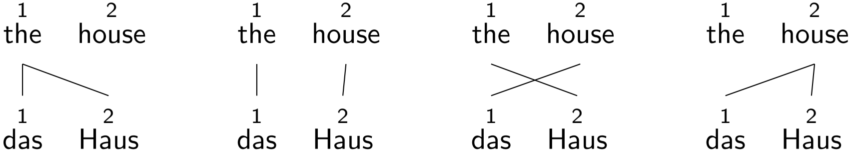


0.5×0.25	0.5×0.5	0.25×0.5	0.5×0.5
0.125	0.25	0.125	0.25

Learning Parameters: IBM Model 1

Compute $\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) = \frac{\Pr(\mathbf{a}, \mathbf{f} | \mathbf{e})}{\Pr(\mathbf{f} | \mathbf{e})}$

$\Pr(\mathbf{f} \mid \mathbf{e}) = 0.125 + 0.25 + 0.125 + 0.25 = 0.75$

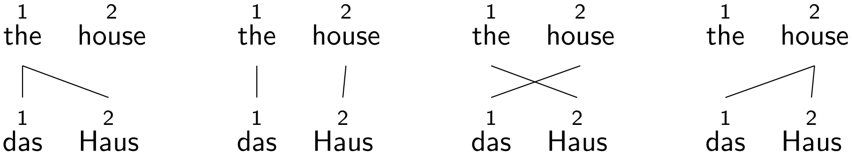


$\frac{0.125}{0.75}$	$\frac{0.25}{0.75}$	$\frac{0.125}{0.75}$	$\frac{0.25}{0.75}$
0.167	0.334	0.167	0.334

Compute fractional counts $c(f, e)$

$$\begin{aligned} c(Haus, the) &= 0.125 + 0.125 & \quad c(das, house) &= 0.125 + 0.25 \\ c(das, the) &= 0.125 + 0.25 & \quad c(Haus, house) &= 0.25 + 0.25 \\ c(Buch, the) &= 0.0 & \quad c(Buch, house) &= 0.0 \end{aligned}$$

Learning Parameters: IBM Model 1



$\Pr(\mathbf{f} \mid \mathbf{e}) = 0.125 + 0.25 + 0.125 + 0.25 = 0.75$

Expectation step: expected counts $g(f, e)$

$$
\begin{aligned}
g(\mathit{das}, \mathit{the}) &= \tfrac{0.125+0.25}{0.75} & \quad g(\mathit{das}, \mathit{house}) &= \tfrac{0.125+0.25}{0.75} \\
g(\mathit{Haus}, \mathit{the}) &= \tfrac{0.125+0.125}{0.75} & \quad g(\mathit{Haus}, \mathit{house}) &= \tfrac{0.25+0.25}{0.75} \\
g(\mathit{Buch}, \mathit{the}) &= 0.0 & \quad g(\mathit{Buch}, \mathit{house}) &= 0.0
\end{aligned}
$$

Maximization step: get new $t^{(1)}(f \mid e) = \frac{g(f,e)}{\sum_f g(f,e)}$

Learning Parameters: IBM Model 1

Expectation step: expected counts $g(f, e)$

$$
\begin{array}{lcl}
g(das, the) & = & 0.5 \\
g(Haus, the) & = & 0.334 \\
g(Buch, the) & = & 0.0 \\
\textbf{total} & = & 0.834
\end{array}
\qquad
\begin{array}{lcl}
g(das, house) & = & 0.5 \\
g(Haus, house) & = & 0.667 \\
g(Buch, house) & = & 0.0 \\
\textbf{total} & = & 1.167
\end{array}
$$

Maximization step: get new $t^{(1)}(f \mid e) = \frac{g(f,e)}{\sum_f g(f,e)}$

$$
\begin{array}{lcl}
t(Haus \mid the) & = & 0.4 \\
t(das, \mid the) & = & 0.6 \\
t(Buch \mid the) & = & 0.0
\end{array}
\qquad
\begin{array}{lcl}
t(das \mid house) & = & 0.43 \\
t(Haus \mid house) & = & 0.57 \\
t(Buch \mid house) & = & 0.0
\end{array}
$$

Keep iterating: Compute $t^{(0)}, t^{(1)}, t^{(2)}, \ldots$ until convergence

Parameter Estimation: IBM Model 1

EM learns the parameters $t(\cdot \mid \cdot)$ that maximizes the log-likelihood of the training data:

$$\arg\max_t L(t) = \arg\max_t \sum_s \log \Pr(\mathbf{f}^{(s)} \mid \mathbf{e}^{(s)}, t)$$

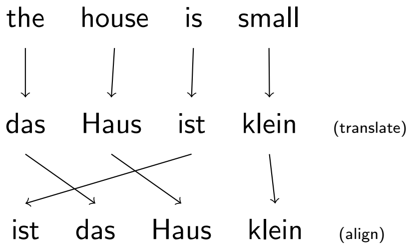
- Start with an initial estimate t_0
- Modify it iteratively to get $t_1, t_2, \ldots$
- Re-estimate t from parameters at previous time step t_{-1}
- The convergence proof of EM guarantees that $L(t) \geq L(t_{-1})$
- EM converges when $L(t) - L(t_{-1})$ is zero (or almost zero).

Statistical Machine Translation

Generative Model of Word Alignment

Word Alignments: IBM Model 2

Generative "story" for Model 2



$$\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \prod_{i=1}^{I} t(f_i \mid e_{a_i}) \times a(a_i \mid i, I, J)$$

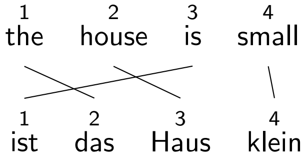
Word Alignments: IBM Model 2

Alignment probability

$$\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) = \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}$$

$$\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \prod_{i=1}^{I} t(f_i \mid e_{a_i}) \times a(a_i \mid i, I, J)$$

Example alignment



$$\begin{aligned}
\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \\
& t(\text{das} \mid \text{the}) \times a(1 \mid 2, 4, 4) \times \\
& t(\text{Haus} \mid \text{house}) \times a(2 \mid 3, 4, 4) \times \\
& t(\text{ist} \mid \text{is}) \times a(3 \mid 1, 4, 4) \times \\
& t(\text{klein} \mid \text{small}) \times a(4 \mid 4, 4, 4)
\end{aligned}$$

Word Alignments: IBM Model 2

Alignment probability

$$
\begin{aligned}
\Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) &= \frac{\Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\Pr(\mathbf{f} \mid \mathbf{e})} \\
&= \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i}) \times a(a_i \mid i, I, J)}{\sum_{\mathbf{a}} \prod_{i=1}^{I} t(f_i \mid e_{a_i}) \times a(a_i \mid i, I, J)} \\
&= \frac{\prod_{i=1}^{I} t(f_i \mid e_{a_i}) \times a(a_i \mid i, I, J)}{\prod_{i=1}^{I} \sum_{j=1}^{J} t(f_i \mid e_j) \times a(j \mid i, I, J)}
\end{aligned}
$$

Word Alignments: IBM Model 2

Learning the parameters

- EM training for IBM Model 2 works the same way as IBM Model 1
- We can do the same factorization trick to efficiently learn the parameters
- The EM algorithm:
  - Initialize parameters t and a (prefer the diagonal for alignments)
  - Expectation step: We collect expected counts for t and a parameter values
  - Maximization step: add up expected counts and normalize to get new parameter values
  - Repeat EM steps until convergence.

Statistical Machine Translation

Generative Model of Word Alignment

- Word Alignments: IBM Model 3
- Word Alignments: IBM Model 1
- Finding the best alignment: IBM Model 1
- Learning Parameters: IBM Model 1
- IBM Model 2
- Back to IBM Model 3

Learning Parameters: IBM Model 3

Parameter Estimation: Sum over all alignments

$$\sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \sum_{\mathbf{a}} \prod_{i=1}^{I} n(\phi_{a_i} \mid e_{a_i}) \times t(f_i \mid e_{a_i}) \times d(i \mid a_i, I, J)$$

Sampling the Alignment Space [from P.Koehn SMT book slides]

- Training IBM Model 3 with the EM algorithm
  - The trick that reduces exponential complexity does not work anymore

  $\rightarrow$ Not possible to exhaustively consider all alignments
- Finding the most probable alignment by hillclimbing
  - start with initial alignment
  - change alignments for individual words
  - keep change if it has higher probability
  - continue until convergence
- Sampling: collecting variations to collect statistics
  - all alignments found during hillclimbing
  - neighboring alignments that differ by a move or a swap

Higher IBM Models[from P.Koehn SMT book slides]

IBM Model 1	lexical translation
IBM Model 2	adds absolute reordering model
IBM Model 3	adds fertility model
IBM Model 4	relative reordering model
IBM Model 5	fixes deficiency

- 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
  - $\rightarrow$ exhaustive count collection becomes computationally too expensive
  - sampling over high probability alignments is used instead

Summary[from P.Koehn SMT book slides]

- IBM Models were the pioneering models in statistical machine translation
- Introduced important concepts
  - generative model
  - EM training
  - reordering models

- Only used for niche applications as translation model
- ... but still in common use for word alignment (e.g., GIZA++, mgiza toolkit)

Natural Language Processing

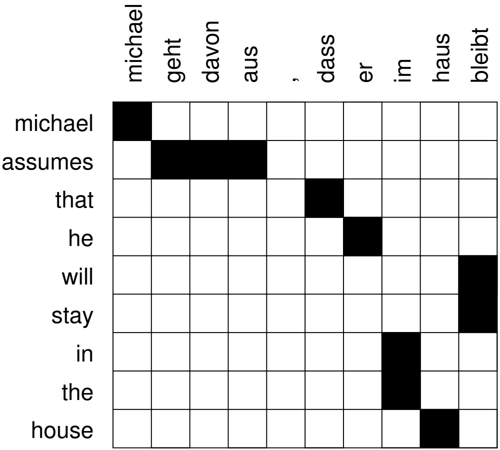
Anoop Sarkar
anoopsarkar.github.io/nlp-class

Simon Fraser University

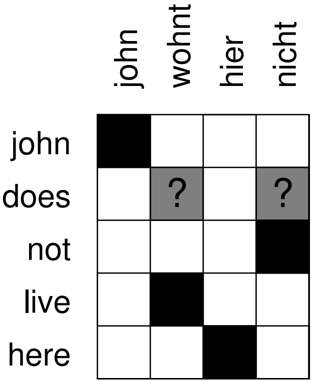
Part 2: Word Alignment

Word Alignment[from P.Koehn SMT book slides]

Given a sentence pair, which words correspond to each other?

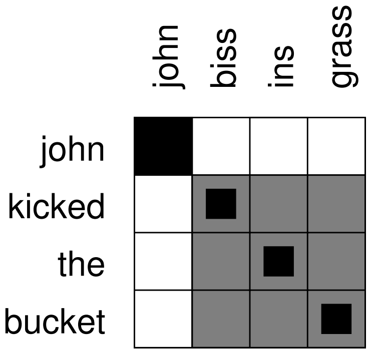


Word Alignment? [from P.Koehn SMT book slides]



Is the English word *does* aligned to
the German *wohnt* (verb) or *nicht* (negation) or neither?

Word Alignment?[from P.Koehn SMT book slides]



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*

Measuring Word Alignment Quality[from P.Koehn SMT book slides]

- Manually align corpus with *sure* (S) and *possible* (P) alignment points ($S \subseteq P$)
- Common metric for evaluation word alignments: Alignment Error Rate (AER)

$$\text{AER}(S, P; A) = \frac{|A \cap S| + |A \cap P|}{|A| + |S|}$$

- AER $= 0$: alignment A matches all sure, any possible alignment points
- However: different applications require different precision/recall trade-offs

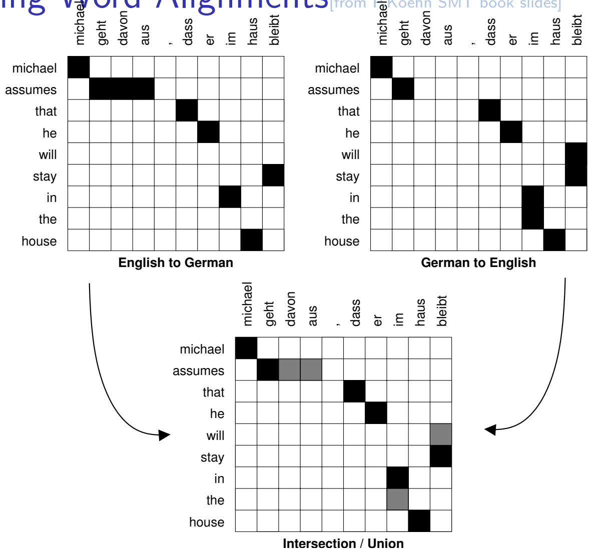
Word Alignment with IBM Models[from P.Koehn SMT book slides]

- 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

Symmetrizing Word Alignments [from P.Koehn SMT book slides]



- Intersection plus grow additional alignment points [Och and Ney, CompLing2003]

Growing heuristic [from P.Koehn SMT book slides]

grow-diag-final(e2f,f2e)

neighboring = {(-1,0),(0,-1),(1,0),(0,1),(-1,-1),(-1,1),(1,-1),(1,1)}
alignment A = intersect(e2f,f2e); grow-diag(); final(e2f); final(f2e);

grow-diag()

while new points added **do**
  for all English word $e \in [1...e_n]$, foreign word $f \in [1...f_n]$, $(e, f) \in A$ **do**
    for all neighboring alignment points $(e_{\text{new}}, f_{\text{new}})$ **do**
      if (e_{new} unaligned OR f_{new} unaligned) AND $(e_{\text{new}}, f_{\text{new}}) \in$ union(e2f,f2e) **then**
        add $(e_{\text{new}}, f_{\text{new}})$ to A
      end if
    end for
  end for
end while

final()

for all English word $e_{\text{new}} \in [1...e_n]$, foreign word $f_{\text{new}} \in [1...f_n]$ **do**
  if (e_{new} unaligned OR f_{new} unaligned) AND $(e_{\text{new}}, f_{\text{new}}) \in$ union(e2f,f2e) **then**
    add $(e_{\text{new}}, f_{\text{new}})$ to A
  end if
end for

More Recent Work on Symmetrization [from P.Koehn SMT book slides]

- Symmetrize after each iteration of IBM Models [Matusov et al., 2004]
  - run one iteration of E-step for each direction
  - symmetrize the two directions
  - count collection (M-step)

- Use of posterior probabilities in symmetrization
  - generate n-best alignments for each direction
  - calculate how often an alignment point occurs in these alignments
  - use this posterior probability during symmetrization

Link Deletion / Addition Models[from P.Koehn SMT book slides]

- Link deletion [Fossum et al., 2008]
  - start with union of IBM Model alignment points
  - delete one alignment point at a time
  - uses a neural network classifiers that also considers aspects such as how useful the alignment is for learning translation rules
- Link addition [Ren et al., 2007] [Ma et al., 2008]
  - possibly start with a skeleton of highly likely alignment points
  - add one alignment point at a time

Discriminative Training Methods[from P.Koehn SMT book slides]

- Given some annotated training data, supervised learning methods are possible
- Structured prediction
  - not just a classification problem
  - solution structure has to be constructed in steps
- Many approaches: maximum entropy, neural networks, support vector machines, conditional random fields, MIRA, ...
- Small labeled corpus may be used for parameter tuning of unsupervised aligner [Fraser and Marcu, 2007]

Better Generative Models[from P.Koehn SMT book slides]

- Aligning phrases
  - joint model [Marcu and Wong, 2002]
  - problem: EM algorithm likes really long phrases

- Fraser and Marcu: LEAF
  - decomposes word alignment into many steps
  - similar in spirit to IBM Models
  - includes step for grouping into phrase

Summary [from P.Koehn SMT book slides]

- Lexical translation
- Alignment
- Expectation Maximization (EM) Algorithm
- Noisy Channel Model
- IBM Models 1–5
  - IBM Model 1: lexical translation
  - IBM Model 2: alignment model
  - IBM Model 3: fertility
  - IBM Model 4: relative alignment model
  - IBM Model 5: deficiency
- Word Alignment

Acknowledgements

Many slides borrowed or inspired from lecture notes by Michael Collins, Chris Dyer, Kevin Knight, Philipp Koehn, Adam Lopez, and Luke Zettlemoyer from their NLP course materials.

All mistakes are my own.