# INF5820/INF9820

## LANGUAGE TECHNOLOGICAL APPLICATIONS

Jan Tore Lønning, Lecture 5, 21 Sep. 2016
jtl@ifi.uio.no

# Today



- Repetition:
  - Statistical machine translation:
    - The noisy channel model
      - IBM model 1
    - Training the intuitive way
- Training – the fast way
- Higher IBM-models

# The noisy channel model

$$\hat{E} = \arg\max_{E} P(E \mid F)$$

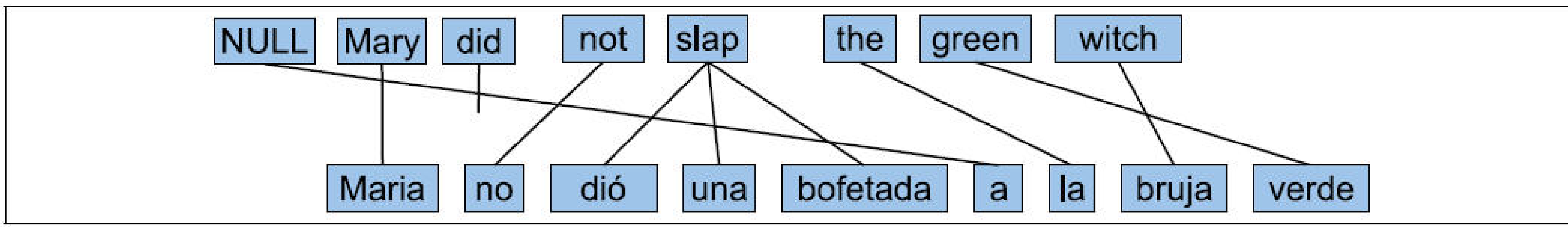
$$= \arg\max_{E} \frac{P(F \mid E)P(E)}{P(F)}$$

$$= \arg\max_{E} P(F \mid E)P(E)$$

- Use n-gram language model for P(E)

# Alignment






- Length of English string: *k* (=7)
- Length of foreign string: *m* (=9)
- An alignment is a vector of length *m*, each entry a number between 0 and k
- The example:
  - $\langle a_1, a_2, \ldots, a_9, \rangle = \langle 1, 3, 4, 4, 4, 0, 5, 7, 6 \rangle$

# IBM Model 1



- Consider all possible alignments **a**:

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

- For each alignment use the simplified generative model:

$$P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \frac{\varepsilon}{(k+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

  - ε is a normalisation factor
  - Formula 4.7 in the SMT book
    - (The book goes f→ e, not e → f)

# Step 3.1: Collect frac. counts ctd



$e_2$: Dog bit dog
$f_2$: Hund bet hund

$f_2, e_2$:

- 64 aligments, with prob 1/64
- $f$ =bet, $e$ = bit
  - 16 alignments connect them: <x,2,z> for x,z in {0,1,2,3}
  - c(bet|bit; $\mathbf{f}_2$, $\mathbf{e}_2$) = 16/64 = 1/4
- $f$ =bet, $e$ = dog
  - all alignments <x,1,z> and <x,3,z> for x,z in {0,1,2,3}
  - c(bet|dog; $\mathbf{f}_2$, $\mathbf{e}_2$) = 2*16/64 = 1/2

| | |
|---|---|
| c(hund\|dog; $\mathbf{f}_2$, $\mathbf{e}_2$)= 1 | c(bet\|dog; $\mathbf{f}_2$, $\mathbf{e}_2$) = 1/2 |
| c(hund\|bit; $\mathbf{f}_2$, $\mathbf{e}_2$) = 1/2 | c(bet\|bit; $\mathbf{f}_2$, $\mathbf{e}_2$) = 1/4 |
| c(hund\|0; $\mathbf{f}_2$, $\mathbf{e}_2$) = 1/2 | c(bet\|0; $\mathbf{f}_2$, $\mathbf{e}_2$) = 1/4 |

# Step 3.1: Collect frac. counts ctd



$f_2$, $e_2$: 64 aligments, with prob 1/64

- $f$ =hund, $e$ = dog
- 2 ways of thinking

$e_2$: Dog bit dog
$f_2$: Hund bet hund

What is the count where
- f1 is a translation of e1?
  - The alignments <1,x,y> for any x and y
  - 16/64
- f1 is a translation of e3?
- e3 a translation of f1?
- e3 a translation of f3?

4*(16/64)=1

For the alignment <1,2,3>:
- How many times is *hund* aligned with *dog*? 2
- Similarly for <x,y,z> where x is 1 or 3 and z is 1 or 3:
  - 16 alignments
  - Frac count: 16*2/64

For the alignments <1,y,z> where z is 0 or 2:
- *hund-dog* aligned 1 time
- 8*1/64=1/8
- Similarly for
  - <3,y,z>, z is 0 or 2
  - <x,y,z>, for $x \in \{0,2\}, y \in \{1,3\}$

# Fractional counts

Counting for one sentence

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}} \left( p(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j}) \right)$$

- The part $\sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j})$

  counts how many times the alignment ***a*** connects a word of the type *f* with one of type e
  - $\delta(a, b) = 1$ if and only if $a = b$, otherwise 0
- We multiply with the probability of this alignment
- And sum over all alignments

# Today



- Repetition:
  - Statistical machine translation:
    - The noisy channel model
      - IBM model 1
    - Training the intuitive way
- Training – the fast way
- Higher IBM-models

# Training – the idea



1. From the translation probabilities, we may estimate alignment probabilities
   - (We do not choose only the best alignment)
2. From alignment probabilities, we may maximize translation probabilities

- By alternating between (1) and (2), the numbers converge towards better results
- For IBM Model 1 it may be proved that they converge towards a global optimum

# Too many alignments

| Words, m=k | 2 | 3 | 4 | | 6 | | 8 | | 10 |
|---|---|---|---|---|---|---|---|---|---|
| Align. | 9 | 64 | 625 | | 117 649 | | 43mill | | 25 billions |

# Training – the intuitive approach



1. Initalize the parameter values $t(f\,|e)$ for pairs of words $f$ and $e$ .
2. For each sentences pair $\boldsymbol{f}$, $\boldsymbol{e}$ calculate the probabilities $P(\boldsymbol{a} \mid \boldsymbol{f}, \boldsymbol{e})$ of all alignments $\boldsymbol{a}$.
3. Collect fractional counts, $tc(f\,|e)$:
   1. First, calculate this, $c(f\,|e\,;\,\boldsymbol{f}, \boldsymbol{e})$ for each sentence $\boldsymbol{f}$, $\boldsymbol{e}$,
   2. Then add over all sentences
4. Calculate the new translation probabilities $t(f\,|e)$
5. Repeat from 2 as long as you like

# Training – the efficient approach



1. Initalize the parameter values $t(f\,|e)$ for pairs of words $f$ and $e$ .
2. ~~For each sentences pair $\boldsymbol{f}$, $\boldsymbol{e}$ calculate the probabilities $P(\boldsymbol{a}\,|\,\boldsymbol{f}, \boldsymbol{e})$ to all alignments $\boldsymbol{a}$.~~
3. Collect fractional counts, $tc(f\,|e)$:
   1. First, calculate this, $c(f\,|e\,;\,\boldsymbol{f}, \boldsymbol{e})$ for each sentence $\boldsymbol{f}$, $\boldsymbol{e}$,
   2. Then add over all sentences
4. Calculate the new translation probabilities $t(f\,|e)$
5. Repeat from 2 as long as you like

# IBM Model 1



- Consider all possible alignments **a**:

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

- For each alignment use the simplified generative model:

$$\boxed{P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \frac{\varepsilon}{(k+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})}$$

  - ε is a normalisation factor
  - Formula 4.7 in the SMT book
    - (The book goes f→ e, not e → f)

# Necessary simplification

$$P(\mathbf{f} \mid \mathbf{e}) = \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \sum_{\mathbf{a}} \frac{\varepsilon}{(k+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

$$P(\mathbf{f} \mid \mathbf{e}) = \sum_{a_1=0}^{k} \cdots \sum_{am=0}^{k} \frac{\varepsilon}{(k+1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a_j})$$

- This equals

$$P(\mathbf{f} \mid \mathbf{e}) = \frac{\varepsilon}{(k+1)^m} \prod_{j=1}^{m} \sum_{i=0}^{k} t(f_j \mid e_i)$$

- Because

$$\prod_{j=1}^{m} \sum_{i=0}^{k} c_{j,i} = (c_{1,0} + c_{1,1} + \ldots + c_{1,k})(c_{2,0} + \ldots + c_{2,k}) \cdots (c_{m,0} + \ldots + c_{m,k}) = \sum_{i=0}^{k} \cdots \sum_{i=0}^{k} \prod_{j=1}^{m} c_{j,i}$$

- Reduces the problem from the order $(k+1)^n$ to roughly $k \times n$

# Putting this together

- So far

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

$$P(\mathbf{f},\mathbf{a}|\mathbf{e}) = \frac{\varepsilon}{(k+1)^m}\prod_{j=1}^{m} t(f_j | e_{a_j})$$

$$P(\mathbf{f}|\mathbf{e}) = \frac{\varepsilon}{(k+1)^m}\prod_{j=1}^{m}\sum_{i=0}^{k} t(f_j | e_i)$$

- Hence

$$P(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\frac{\varepsilon}{(k+1)^m}\prod_{j=1}^{m} t(f_j | e_{a_j})}{\frac{\varepsilon}{(k+1)^m}\prod_{j=1}^{m}\sum_{i=0}^{k} t(f_j | e_i)}$$

- Formula 4.11

$$P(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\prod_{j=1}^{m} t(f_j | e_{a_j})}{\prod_{j=1}^{m}\sum_{i=0}^{k} t(f_j | e_i)}$$

# Fractional counts

Counting for one sentence

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}} \left( p(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j}) \right)$$

- The part $\sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j})$

  counts how many times the alignment ***a*** connects a word of the type *f* with one of type e
  - $\delta(a, b) = 1$ if and only if $a = b$, otherwise $0$
- We multiply with the probability of this alignment
- And sum over all alignments

# Fractional counts

- Counting for one sentence

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}} \left( p(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j}) \right)$$

- Substituting in for p(**a | e,f**)

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}} \left( \frac{\prod_{j=1}^{m} t(f_j \mid e_{\mathbf{a}_j})}{\prod_{j=1}^{m} \sum_{i=0}^{k} t(f_j \mid e_i)} \sum_{j=1}^{m} \delta(f, f_j) \times \delta(e, e_{\mathbf{a}_j}) \right)$$

- and doing some non-trivial calculation:

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \frac{t(f \mid e)}{\sum_{i=0}^{k} t(f \mid e_i)} \sum_{j=1}^{m} \delta(f, f_j) \sum_{i=0}^{k} \delta(e, e_i)$$

Observe:
Directly from *t* to
*c*(*f*|*e*;**e**,**f**) without
mentioning the ***a***-s

# Fractional counts

- Counting over the whole corpus and normalize as before

$$t(f \mid e) = \frac{\sum_{(\mathbf{f},\mathbf{e})} c(f \mid e; \mathbf{f}, \mathbf{e})}{\sum_{f'} \sum_{(\mathbf{f},\mathbf{e})} c(f' \mid e; \mathbf{f}, \mathbf{e})}$$

# Example – the efficient way



- Corpus

$\mathbf{e}_1$: Dog barked
$\mathbf{f}_1$: Hund bjeffet

$\mathbf{e}_2$: Dog bit dog
$\mathbf{f}_2$: Hund bet hund

3 English words: dog bit barked
3 foreign words: hund bjeffet bet

## Uniform initilaization

| | | |
|---|---|---|
| t(hund \| dog) = 1/3 | t(bet \| dog) = 1/3 | t(bjeffet \| dog) = 1/3 |
| t(hund \| bit) = 1/3 | t(bet \| bit) = 1/3 | t(bjeffet \| bit) = 1/3 |
| t(hund \| barked) = 1/3 | t(bet \| barked) = 1/3 | t(bjeffet \| barked) = 1/3 |
| t(hund \| 0) = 1/3 | t(bet \| 0) = 1/3 | t(bjeffet \| 0) = 1/3 |

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \frac{t(f \mid e)}{\sum_{i=0}^{k} t(f \mid e_i)} \sum_{j=1}^{m} \delta(f, f_j) \sum_{i=0}^{k} \delta(e, e_i)$$

**e**$_1$: Dog barked
**f**$_1$: Hund bjeffet

$$c(hund \mid barked; \mathbf{e}_1, \mathbf{f}_1) = \frac{t(hund \mid barked)}{\sum_{i=0}^{2} t(f \mid e_i)} \sum_{j=1}^{2} \delta(hund, f_j) \sum_{i=0}^{2} \delta(barked, e_i) =$$

$$\frac{1/3}{\sum_{i=0}^{2} (1/3)} (\delta(hund, hund) + \delta(hund, bjeffet)) \times$$

$$(\delta(barked, 0) + \delta(barked, dog) + \delta(barked, barked)) = 1/3$$

## Uniform initilaization

| | | |
|---|---|---|
| t(hund \| dog) = 1/3 | t(bet \| dog) = 1/3 | t(bjeffet \| dog) = 1/3 |
| t(hund \| bit) = 1/3 | t(bet \| bit) = 1/3 | t(bjeffet \| bit) = 1/3 |
| t(hund \| barked) = 1/3 | t(bet \| barked) = 1/3 | t(bjeffet \| barked) = 1/3 |
| t(hund \| 0) = 1/3 | t(bet \| 0) = 1/3 | t(bjeffet \| 0) = 1/3 |

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \frac{t(f \mid e)}{\sum_{i=0}^{k} t(f \mid e_i)} \sum_{j=1}^{m} \delta(f, f_j) \sum_{i=0}^{k} \delta(e, e_i)$$

**$\mathbf{e}_2$**: Dog bit dog
**$\mathbf{f}_2$**: Hund bet hund

$$c(bet \mid bit; \mathbf{e}_2, \mathbf{f}_2) = \frac{t(bet \mid bit)}{\sum_{i=0}^{3} t(bet \mid e_i)} \sum_{j=1}^{3} \delta(bet, f_j) \sum_{i=0}^{3} \delta(bit, e_i) = \frac{1/3}{\sum_{i=0}^{3} (1/3)} \times 1 \times 1 = 1/4$$

$$c(hund \mid dog; \mathbf{e}_2, \mathbf{f}_2) = \frac{t(hund \mid dog)}{\sum_{i=0}^{3} t(hund \mid e_i)} \sum_{j=1}^{3} \delta(hund, f_j) \sum_{i=0}^{3} \delta(dog, e_i) = \frac{1/3}{\sum_{i=0}^{3} (1/3)} \times 2 \times 2 = 1$$

# Collect fractional counts



$e_1$: Dog barked
$f_1$: Hund bjeffet

Results are the same as the intuitive way

| | |
|---|---|
| c(hund \| dog; $f_1$, $e_1$)= 1/3 | c(bjeffet \| dog; $f_1$, $e_1$) = 1/3 |
| c(hund \| barked; $f_1$, $e_1$) = 1/3 | c(bjeffet \| barked; $f_1$, $e_1$) = 1/3 |
| c(hund \| 0; $f_1$, $e_1$) = 1/3 | c(bjeffet \| 0; $f_1$, $e_1$) = 1/3 |

$e_2$: Dog bit dog
$f_2$: Hund bet hund

| | |
|---|---|
| c(hund \| dog; $f_2$, $e_2$)= 1 | c(bet \| dog; $f_2$, $e_2$) = 1/2 |
| c(hund \| bit; $f_2$, $e_2$) = 1/2 | c(bet \| bit; $f_2$, $e_2$) = 1/4 |
| c(hund \| 0; $f_2$, $e_2$) = 1/2 | c(bet \| 0; $f_2$, $e_2$) = 1/4 |

# Step 3.2: Total counts (as before)



$$tc(f \mid e) = \sum_{(\mathbf{f},\mathbf{e})} c(f \mid e; \mathbf{f}, \mathbf{e})$$

| | | | |
|---|---|---|---|
| tc(hund\|dog) = 1+1/3 | tc(bet\|dog) = 1/2 | tc(bjeffet\|dog) = 1/3 | tc(*\|dog)=4/3+1/2+1/3 =13/6 |
| tc(hund\|bit) = ½ | tc(bet\|bit) = ¼ | tc(bjeffet\|bit) = 0 | tc(*\|bit)=3/4 |
| tc(hund\|barked) = 1/3 | tc(bet\|barked) = 0 | tc(bjeffet\|barked) = 1/3 | tc(*\|barked) =2/3 |
| tc(hund\|0) = ½+1/3 | tc(bet\|0) = 1/4 | tc(bjeffet\|0) = 1/3 | tc(*\|0)=17/12 |

# Step 4: new trans. probabilities



$$t(f|e) = \frac{tc(f|e)}{\sum_{f'} tc(f'|e)}$$

| e | f | t(f\|e) | exact | decimal |
|---|---|---|---|---|
| 0 | hund | (5/6)/(17/12) | 10/17 | 0.588235 |
| 0 | bet | (1/4)/(17/12) | 3/17 | 0.176471 |
| 0 | bjeffet | (1/3)/(17/12) | 4/17 | 0.235294 |
| dog | hund | (4/3)/(13/6) | 8/13 | 0.615385 |
| dog | bet | (1/2)/(13/6) | 3/13 | 0.230769 |
| dog | bjeffet | (1/3)/(13/6) | 2/13 | 0.153846 |
| bit | hund | (1/2)/(3/4) | 2/3 | 0.666667 |
| bit | bet | (1/4)/(3/4) | 1/3 | 0.333333 |
| barked | hund | (1/3)/(2/3 | 1/2 | 0.5 |
| barked | bjeffet | (1/3)/(2/3) | 1/2 | 0.5 |

# Repeat: calculate fractional counts



- Examples

$$c(hund \mid barked; \mathbf{e}_1, \mathbf{f}_1) = \frac{t(hund \mid barked)}{\sum_{i=0}^{2} t(f \mid e_i)} \sum_{j=1}^{2} \delta(hund, f_j) \sum_{i=0}^{2} \delta(barked, e_i) =$$

$$\frac{0.5}{0.588235 + 0.615385 + 0.5} = \frac{0.5}{1.70362} = 0.2934927$$

$$c(hund \mid dog; \mathbf{e}_2, \mathbf{f}_2) = \frac{t(hund \mid dog)}{\sum_{i=0}^{3} t(hund \mid e_i)} \sum_{j=1}^{3} \delta(hund, f_j) \sum_{i=0}^{3} \delta(dog, e_i) =$$

$$\frac{0.615385}{0.588235 + 0.615385 + 0.666667 + 0.615385} \times 2 \times 2 = ?$$

# After some iterations

| | | 1st iterat. | 2nd iter. | 5th iter. | 25th iter | 100th |
|---|---|---|---|---|---|---|
| 0 | hund | 0.588235 | | | | |
| 0 | bet | 0.176471 | | | | |
| 0 | bjeffet | 0.235294 | | | | |
| dog | hund | 0.615385 | | | | |
| dog | bet | 0.230769 | | | | |
| dog | bjeffet | 0.153846 | | | | |
| bit | hund | 0.666667 | | | | |
| bit | bet | 0.333333 | | | | |
| barked | hund | 0.5 | | | | |
| barked | bjeffet | 0.5 | | | | |

# After some iterations

| | | 1st iterat. | 2nd iter. | 5th iter. | 25th iter | 100th |
|---|---|---|---|---|---|---|
| 0 | hund | 0.588235 | 0.647158 | | | |
| 0 | bet | 0.176471 | 0.14363 | | | |
| 0 | bjeffet | 0.235294 | 0.209212 | | | |
| dog | hund | 0.615385 | 0.675859 | | | |
| dog | bet | 0.230769 | 0.237614 | | | |
| dog | bjeffet | 0.153846 | 0.086527 | | | |
| bit | hund | 0.666667 | 0.609848 | | | |
| bit | bet | 0.333333 | 0.390152 | | | |
| barked | hund | 0.5 | 0.342932 | | | |
| barked | bjeffet | 0.5 | 0.657068 | | | |

# After some iterations

| | | 1st iterat. | 2nd iter. | 5th iter. | 25th iter | 100th |
|---|---|---|---|---|---|---|
| 0 | hund | 0.588235 | 0.647158 | 0.81929 | | |
| 0 | bet | 0.176471 | 0.14363 | 0.067291 | | |
| 0 | bjeffet | 0.235294 | 0.209212 | 0.113419 | | |
| dog | hund | 0.615385 | 0.675859 | 0.773893 | | |
| dog | bet | 0.230769 | 0.237614 | 0.214793 | | |
| dog | bjeffet | 0.153846 | 0.086527 | 0.011313 | | |
| bit | hund | 0.666667 | 0.609848 | 0.417491 | | |
| bit | bet | 0.333333 | 0.390152 | 0.582509 | | |
| barked | hund | 0.5 | 0.342932 | 0.097766 | | |
| barked | bjeffet | 0.5 | 0.657068 | 0.902234 | | |

# After some iterations

| | | 1st iterat. | 2nd iter. | 5th iter. | 25th iter | 100th |
|---|---|---|---|---|---|---|
| 0 | hund | 0.588235 | 0.647158 | 0.81929 | 0.998457 | |
| 0 | bet | 0.176471 | 0.14363 | 0.067291 | 0.000122 | |
| 0 | bjeffet | 0.235294 | 0.209212 | 0.113419 | 0.001421 | |
| dog | hund | 0.615385 | 0.675859 | 0.773893 | 0.947458 | |
| dog | bet | 0.230769 | 0.237614 | 0.214793 | 0.052541 | |
| dog | bjeffet | 0.153846 | 0.086527 | 0.011313 | 0 | |
| bit | hund | 0.666667 | 0.609848 | 0.417491 | 0.005351 | |
| bit | bet | 0.333333 | 0.390152 | 0.582509 | 0.994648 | |
| barked | hund | 0.5 | 0.342932 | 0.097766 | 6.0e-07 | |
| barked | bjeffet | 0.5 | 0.657068 | 0.902234 | 0.999999 | |

# After some iterations

| | | 1st iterat. | 2nd iter. | 5th iter. | 25th iter | 100th |
|---|---|---|---|---|---|---|
| 0 | hund | 0.588235 | 0.647158 | 0.81929 | 0.998457 | 1 |
| 0 | bet | 0.176471 | 0.14363 | 0.067291 | 0.000122 | 0 |
| 0 | bjeffet | 0.235294 | 0.209212 | 0.113419 | 0.001421 | 0 |
| dog | hund | 0.615385 | 0.675859 | 0.773893 | 0.947458 | 0.966031 |
| dog | bet | 0.230769 | 0.237614 | 0.214793 | 0.052541 | 0.033968 |
| dog | bjeffet | 0.153846 | 0.086527 | 0.011313 | 0 | 0 |
| bit | hund | 0.666667 | 0.609848 | 0.417491 | 0.005351 | 0 |
| bit | bet | 0.333333 | 0.390152 | 0.582509 | 0.994648 | 1 |
| barked | hund | 0.5 | 0.342932 | 0.097766 | 6.0e-07 | 0 |
| barked | bjeffet | 0.5 | 0.657068 | 0.902234 | 0.999999 | 1 |

# Results (perplexitiy)



- Claim: «the numbers converge towards better results»
- Means: for each round

$$\prod_{(\mathbf{e},\mathbf{f})} P(\mathbf{f} \mid \mathbf{e})$$

  does not decrease
- For IBM Model 1 it may be proved that they converge towards a global optimum

# Today



- Repetition:
  - Statistical machine translation:
    - The noisy channel model
      - IBM model 1
    - Training the intuitive way
- Training – the fast way
- Higher IBM-models

# IBM model 2

$$P(\mathbf{f},\mathbf{a}\,|\,\mathbf{e}) = P(m\,|\,\mathbf{e})\prod_{j=1}^{m} P(a_j\,|\,a_1^{j-1}, f_1^{j-1}, m, \mathbf{e})P(f_j\,|\,a_1^{j}, f_1^{j-1}, m, \mathbf{e})$$

- *New*
  - $P(a_j\,|\,a_1^{j-1}, f_1^{j-1}, m, \mathbf{e}) = a(a_j\,|\,j, m, k)$
    - For a probaility distribution a
    - i.e. it depends on the length of the string and the position
    - (less likely to move far than to stay close)
- As for Model1
  - $P(m\,|\,\mathbf{e})$ is a constant, independent of *m* and *E*
  - the word translation probability only depends on source word $P(f_j\,|\,a_1^{j}, f_1^{j-1}, m, \mathbf{e}) = t(f_j\,|\,e_{a_j})$

# Model2

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

- We can do similar steps as for Model1 for expressing P(**f**|**e**) and P(**a**).
- We can do similar simplifications to bypass the exponential number of alignments, and
- Learn the alignment probabilities $a(a_j|j,m,k)$ at the same time as the translation probabilities
- You don't have to learn the details

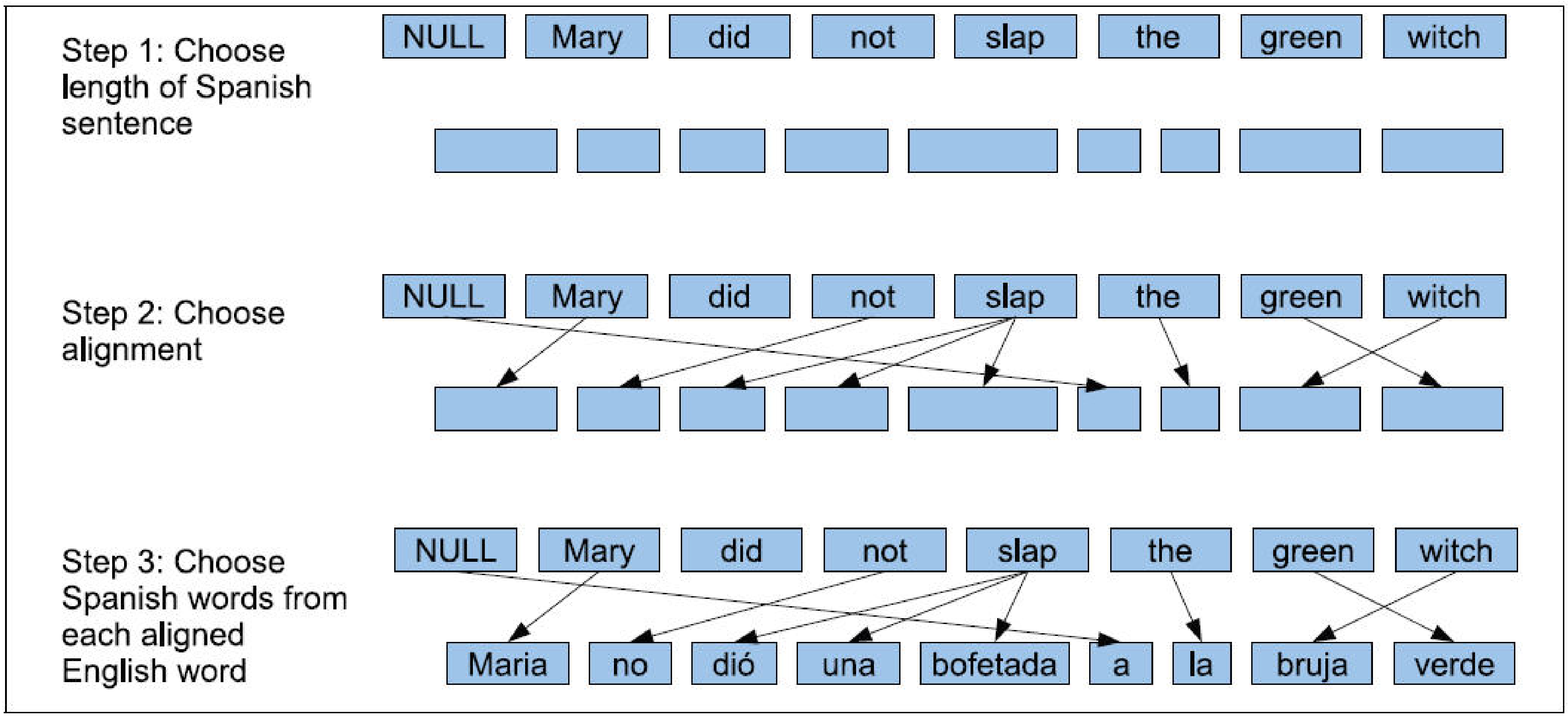
# HMM Alignment

$$P(\mathbf{f},\mathbf{a}\,|\,\mathbf{e}) = P(m\,|\,\mathbf{e})\prod_{j=1}^{m} P(a_j\,|\,a_1^{j-1}, f_1^{\,j-1}, m, \mathbf{e}) P(f_j\,|\,a_1^{j}, f_1^{\,j-1}, m, \mathbf{e})$$

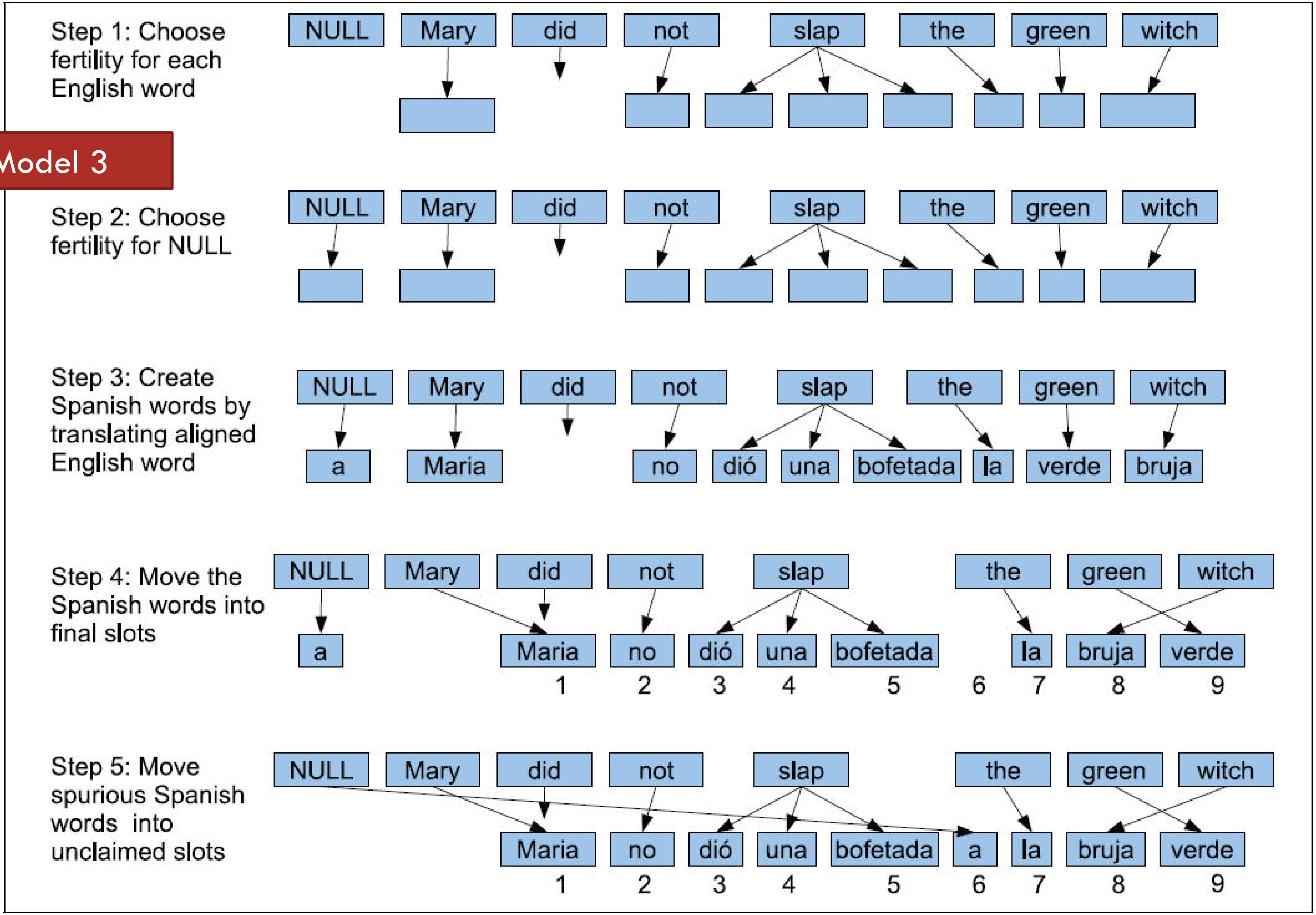
$$P(\mathbf{f},\mathbf{a}\,|\,\mathbf{e}) = P(m\,|\,k)\prod_{j=1}^{m} P(a_j\,|\,a_1^{j-1}, k)\, t(f_j\,|\,e_{a_j})$$

- $P(m\,|\,k)$ depends on the length *k* of **e**.
- $P(a_j\,|\,a_1^{j-1}, f_1^{\,j-1}, m, \mathbf{e}) = P(a_j\,|\,a_{j-1}, k) = \lambda c(a_j - a_{j-1})$
  - Where word j should come from, depends on where word j-1 came from
  - This is again reduced to probabilities, c, of the distance between $a_j$ and $a_{j-1}$ independently of the actual j.

# Model 1 & 2 and HMM alignment



## Model 3



Step 1: Choose fertility for each English word
NULL
Mary
did
not
slap
the
green
witch
Step 2: Choose fertility for NULL
Step 3: Create Spanish words by translating aligned English word
a
Maria
no
dió
una
bofetada
la
verde
bruja
Step 4: Move the Spanish words into final slots
1
2
3
4
5
6
7
8
9
Step 5: Move spurious Spanish words into unclaimed slots

# IBM Model 3: Fertility

- Fertility: number of F words produced by an E word
- Modelled by a distribution $n(x|e)$

Example:
F = Norw.
n(2 | yesterday) ≈ 1
n(1 | to) ≈ 0.8
n(2 | to) ≈ 0.2
n(1 | car) ≈ 1
n(0 | the) ≈ 0.6
n(1 | the) ≈ 0.4

Example:
Norw.→Eng.
n(2 | bilen) ≈ 0.7
n(1 | bilen) ≈ 0.3
n(1 | å) ≈ 0.8
n(0 | å) ≈ 0.2

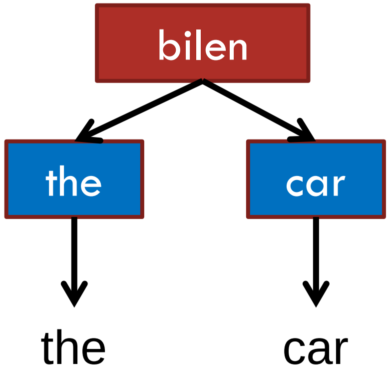
# IBM Model 3: Null insertion

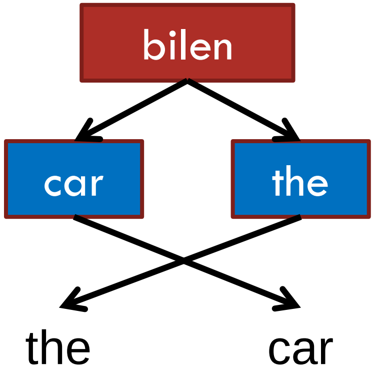
- Modelled by:
- There is a probability p0:
  - After each inserted word there is the probability p0 of not inserting a null-word
  - And a probability p1 = (1-p0) of inserting a null-word
- A rather complex expression for what this contributes into P(a, **f**|**e**) which considers
  - Permutations
  - Length of **f**

# IBM Model 3: Distortion

$$d(j \mid a_j, m, k)$$

- A probability distribution which gives the probability of word $a_i$ ending up in position $j$.
- Similar to alignment in model 2 but:
  - Oposite direction
  - Different choices of words + distortion may correpsond to the same alignment



# IBM model 3

$$P(\mathbf{f},\mathbf{a}\,|\,\mathbf{e}) = \prod_{j=1}^{m} t(f_j\,|\,e_{a_j}) \prod_{j=1}^{m} d(j\,|\,a_j,k,m) \times \text{ more}$$

- Where *more* is an expression which counts
  - $n(x\,|\,e_i)$ the right number of times
  - And uses p0 to give the right probability to null-insertion.

# Training Model 3

- In principle like Model 1, but
  - The trick to get rid of the alignments does not work
  - Too costly to calculate all alignments
- Strategy
  - Sample and use the most probable alignments
  - Start with alignments for Model 1 and Model 2
  - Use hill-climbing algorithm

# Hill-climbing algorithm

- Assign some initial parameter values
- Consider several alternative sets of parameter values in the vicinity of where you are
- Compare the resulting values and choose the parameters which yield the best results
- Repeat

# Training model 3

- Model 1: The optimum we find is global
- Model 3 (and model 2):
  - A local optimum does not have to be global
- First run some iterations of Model1 and maybe some iterations of Model 2
- Use the results, in particular the alignment, as input to Model 3
- Hill-climb the space of alignments from here, doing minimal changes.

# IBM Model 4

- Better reordering model
- Consider group of words (phrases)
- Distinguish between
  - the placement of the whole group
  - The placement within the group

# The IBM-models

- IBM models 1-4 are not true probability models.
- Model 5 fixes this
  - Based on model 4
- We will not consider models 4 and 5
- Phrase Based translation makes use of Model 3