Advanced Alignment Models

Philipp Koehn

11 February 2016

IBM Model 1

- Generative model: break up translation process into smaller steps
  - IBM Model 1 only uses lexical translation

- Translation probability
  - for a foreign sentence $\mathbf{f} = (f_1, ..., f_{l_f})$ of length l_f
  - to an English sentence $\mathbf{e} = (e_1, ..., e_{l_e})$ of length l_e
  - with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \to i$

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{\epsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

  - parameter ϵ is a normalization constant

IBM Model 1 and EM

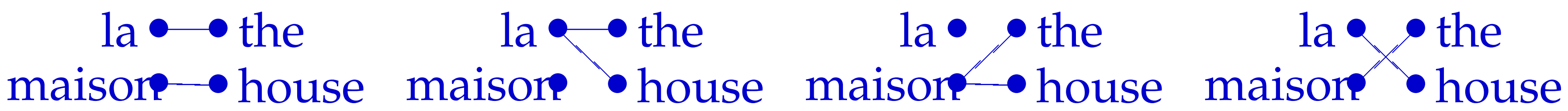
- EM Algorithm consists of two steps
- Expectation-Step: Apply model to the data
  - parts of the model are hidden (here: alignments)
  - using the model, assign probabilities to possible values
- Maximization-Step: Estimate model from data
  - take assign values as fact
  - collect counts (weighted by probabilities)
  - estimate model from counts
- Iterate these steps until convergence

IBM Model 1 and EM

- **Probabilities**

$$p(\text{the}|\text{la}) = 0.7 \qquad p(\text{house}|\text{la}) = 0.05$$
$$p(\text{the}|\text{maison}) = 0.1 \qquad p(\text{house}|\text{maison}) = 0.8$$

- **Alignments**



$$p(\mathbf{e}, a|\mathbf{f}) = 0.56 \qquad p(\mathbf{e}, a|\mathbf{f}) = 0.035 \qquad p(\mathbf{e}, a|\mathbf{f}) = 0.08 \qquad p(\mathbf{e}, a|\mathbf{f}) = 0.005$$

$$p(a|\mathbf{e}, \mathbf{f}) = 0.824 \qquad p(a|\mathbf{e}, \mathbf{f}) = 0.052 \qquad p(a|\mathbf{e}, \mathbf{f}) = 0.118 \qquad p(a|\mathbf{e}, \mathbf{f}) = 0.007$$

- **Counts**

$$c(\text{the}|\text{la}) = 0.824 + 0.052 \qquad c(\text{house}|\text{la}) = 0.052 + 0.007$$
$$c(\text{the}|\text{maison}) = 0.118 + 0.007 \qquad c(\text{house}|\text{maison}) = 0.824 + 0.118$$

IBM Model 1 and EM

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

IBM Model 1 and EM: Expectation Step

- We need to compute $p(a|\mathbf{e}, \mathbf{f})$
- Applying the chain rule:

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

- We already have the formula for $p(\mathbf{e}, \mathbf{a}|\mathbf{f})$ (definition of Model 1)

IBM Model 1 and EM: Maximization Step

- Now we have to collect counts

- Evidence from a sentence pair **e,f** that word e is a translation of word f:

$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_a p(a|\mathbf{e},\mathbf{f}) \sum_{j=1}^{l_e} \delta(e, e_j)\delta(f, f_{a(j)})$$

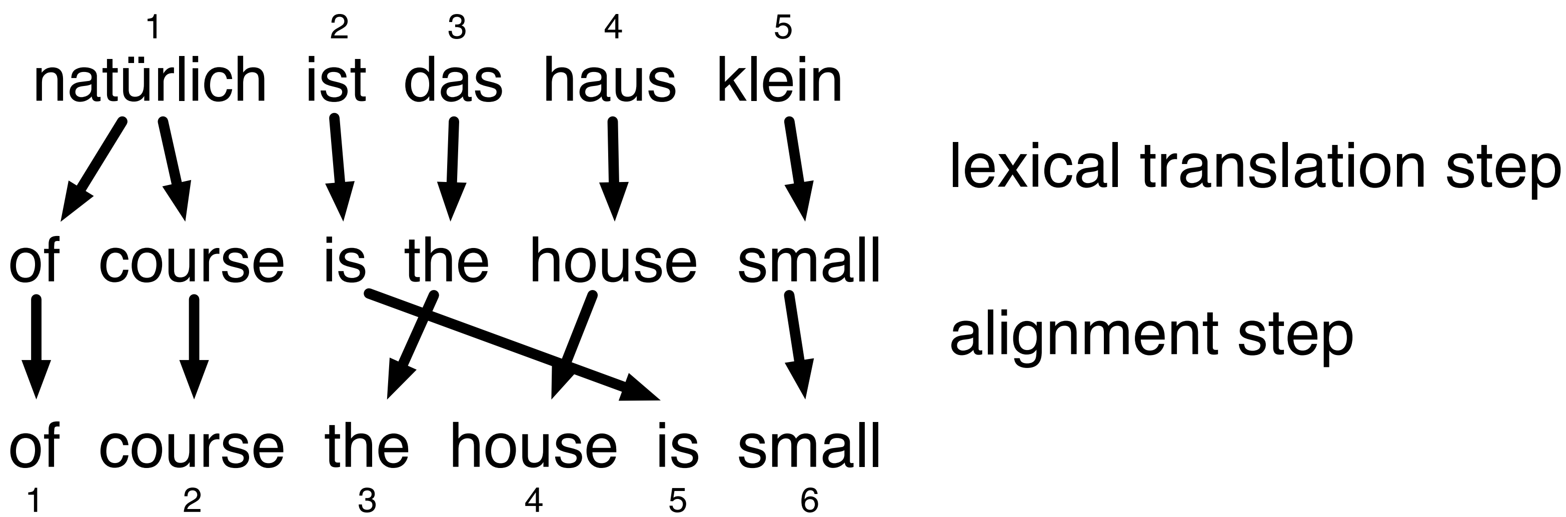
- With the same simplication as before:

$$c(e|f;\mathbf{e},\mathbf{f}) = \frac{t(e|f)}{\sum_{i=0}^{l_f} t(e|f_i)} \sum_{j=1}^{l_e} \delta(e, e_j) \sum_{i=0}^{l_f} \delta(f, f_i)$$

ibm model 2

IBM Model 2

Adding a model of alignment



IBM Model 2

- Modeling alignment with an alignment probability distribution

- Translating English word at position j from foreign word at position $i = a(j)$:

$$a(i|j, l_e, l_f)$$

- Added to IBM Model 1

$$p(\mathbf{e}, a|\mathbf{f}) = \epsilon \prod_{j=1}^{l_e} t(e_j|f_{a(j)})\ a(a(j)|j, l_e, l_f)$$

EM Training of IBM Model 2

- Very similar to IBM Model 1 training
  - number of possible word alignments exponential with number of words
  - but: able to reduce complexity of computing $p(\mathbf{e}|\mathbf{f})$ to polynomial
  - same trick applies to IBM Model 2

$$
\begin{aligned}
p(\mathbf{e}|\mathbf{f}) &= \sum_a p(\mathbf{e},a|\mathbf{f}) \\
&= \epsilon \sum_{a(1)=0}^{l_f} \dots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})\; a(a(j)|j,l_e,l_f) \\
&= \epsilon \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_{a(j)})\; a(a(j)|j,l_e,l_f)
\end{aligned}
$$

Count Collection

- Count collection for lexical translation

$$c(e|f;\mathbf{e},\mathbf{f}) = \sum_{j=1}^{l_e} \sum_{i=0}^{l_f} \frac{t(e|f)\ a(a(j)|j,l_e,l_f)\ \delta(e,e_j)\ \delta(f,f_i)}{\sum_{i'=0}^{l_f} t(e|f_{i'})\ a(i'|j,l_e,l_f))}$$

- Count collection for alignment

$$c(i|j,l_e,l_f;\mathbf{e},\mathbf{f}) = \frac{t(e_j|f_i)\ a(a(j)|j,l_e,l_f)}{\sum_{i'=0}^{l_f} t(e_j|f_{i'})\ a(i'|j,l_e,l_f))}$$

Remarks

- Algorithm for training Model 2 is very similar to the one for IBM Model 1 (pseudo code in book)

- First run a few iterations of IBM Model 1 training

- Initialize probability distributions $t(e|f)$ and $a(i|j, l_e, l_f)$ from IBM Model 1
  - lexical translation probability distribution $t(e|f)$ can be taken verbatim
  - $a(i|j, l_e, l_f)$ initialized to $\frac{1}{l_f+1}$

fast align:
reparameterization of ibm model 2

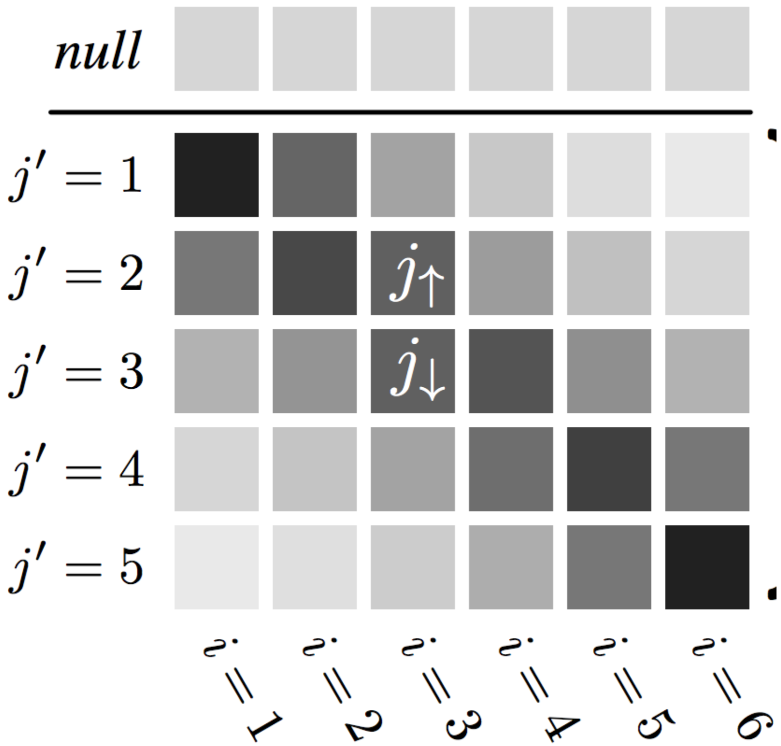
IBM Model 2: A Critique

- Alignment probability distribution has too many parameters ($l_e^2 l_f^2$)

$$a(i|j, l_e, l_f)$$

$\rightarrow$ too sparse data to estimate correctly

- Better: bias towards to diagonal



Diagonal

- Distance from diagonal

$$h(i,j,l_e,l_f) = \left|\frac{i}{l_f} - \frac{j}{l_e}\right|$$

- Function that gives higher values to positions close to diagonal (λ is a scaling factor)

$$e^{-\lambda h(i,j,l_e,l_f)}$$

- Special case: alignment to NULL token: p_0

- Alignment probability distribution

$$\delta(a(j) = i | j, l_e, l_f) = \begin{cases} p_0 & \text{if } i = 0 \\ (1 - p_0)\frac{e^{-\lambda h(i,j,l_e,l_f)}}{Z_\lambda(j,m,n)} & \text{if } 0 < i \leq l_e \end{cases}$$

Remarks

- This model was proposed by Dyer et al. (2013)

- It also changes the word translation probability distribution to include a prior
  - this was originally proposed by Mermer and Saraclar (2011)
  - an efficient estimation method (variational Bayes) was proposed by Riley and Gildea (2012)

- EM training is still simple
  - the probability to align an English word e to a foreign word f does not depend on the choices of other English words
  - the normalization function $Z_\lambda(j, m, n)$ can be computed in $O(1)$

hmm model

HMM Model

- Words do not move independently of each other
  - they often move in groups

  $\rightarrow$ condition word position on previous word's position

- HMM alignment model:

$$a(a(j)|a(j-1), l_e)$$

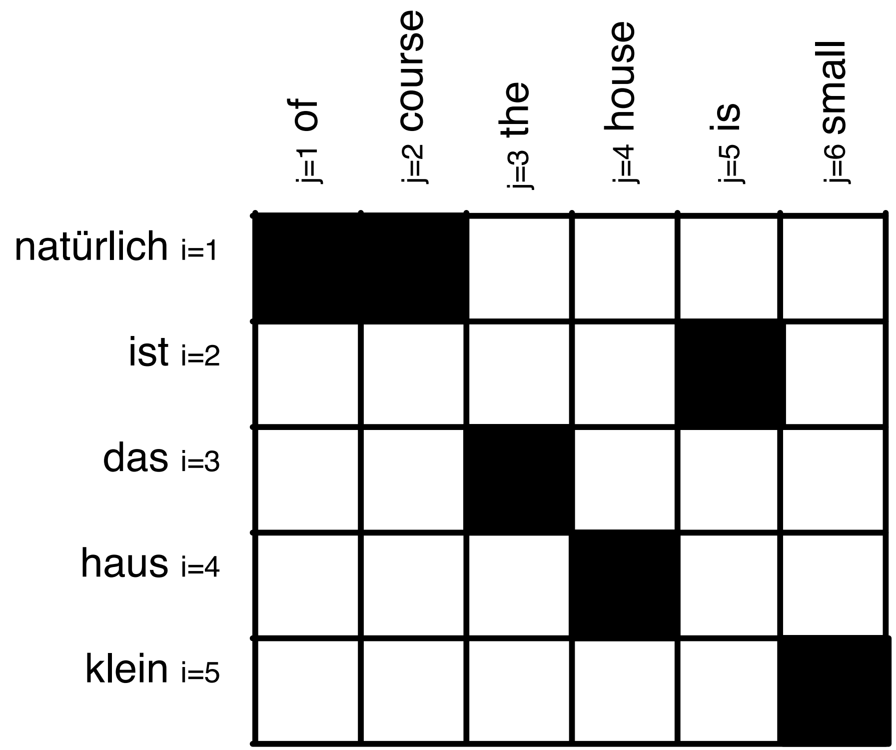
- EM algorithm application slightly harder, requires dynamic programming

- IBM Model 4 is similar, also conditions on word classes

EM for the HMM Model

- Main objective: collect fractional counts to estimate
  - word translation probability distribution $t(e_j|f_{a(j)})$
  - alignment probability distribution $a(a(j)|a(j-1), l_e)$
- Consider all possible word alignments
- Collect evidence from each
- Exponentially many $\rightarrow$ need to do this efficiently

Probability of a Word Alignment



$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$
$t(e_1 \mid f_1) =$ $t(\text{of} \mid \text{natürlich})$	$t(e_2 \mid f_1) =$ $t(\text{course} \mid \text{natürlich})$	$t(e_3 \mid f_3) =$ $t(\text{the} \mid \text{das})$	$t(e_4 \mid f_4) =$ $t(\text{house} \mid \text{hays})$	$t(e_5 \mid f_2) =$ $t(\text{is} \mid \text{its})$	$t(e_6 \mid f_5) =$ $t(\text{small} \mid \text{klein})$
$a(a(1) \mid a(0)) =$ $a(1 \mid 0)$	$a(a(2) \mid a(1)) =$ $a(1 \mid 1)$	$a(a(3) \mid a(2)) =$ $a(3 \mid 1)$	$a(a(4) \mid a(3)) =$ $a(4 \mid 3)$	$a(a(5) \mid a(4)) =$ $a(2 \mid 4)$	$a(a(6) \mid a(5)) =$ $a(5 \mid 2)$

First English Word

	of $j=1$	**course** $j=2$	**the** $j=3$	...
natürlich $a(j)=1$	$q_1(1) = t(\text{of}\mid\text{natürlich}) \times a(1\mid 0)$			
its $a(j)=2$	$q_1(2) = t(\text{of}\mid\text{ist}) \times a(2\mid 0)$			
das $a(j)=3$	$q_1(3) = t(\text{of}\mid\text{das}) \times a(3\mid 0)$			
...				

- Compute probabilities for each choice of $i = a(1)$ by normalizing $q_1(i)$

$$p_1(i) = \frac{q_1(i)}{\sum_{i'} q_1(i')}$$

- Use these probabilities for count collection for $t(\text{of}|\bullet)$ and $a(\bullet|0)$

Next English Word

- One way to get there

	of $j=1$		**course** $j=2$	**the** $j=3$	...
natürlich $a(j)=1$	$p_1(1)$	$\Rightarrow$	$q_2(1 \leftarrow 1) =$ $t(\text{course}\vert\text{natürlich})$ $\times a(1\vert 1) \times p_1(1)$		
ist $a(j)=2$	$p_1(2)$		$q_2(2) = ...$		
das $a(j)=3$	$p_1(3)$		$q_3(2) = ...$		
...					

Next English Word

- Another way to get there

	of $j=1$		course $j=2$	the $j=3$	...
			$q_2(1 \leftarrow 2) =$		
natürlich $a(j)=1$	$p_1(1)$		$t(\text{course}\mid\text{natürlich})$ $\times a(1\mid 2) \times p_1(2)$		
ist $a(j)=2$	$p_1(2)$	$\Rightarrow$	$q_2(2) = ...$		
das $a(j)=3$	$p_1(3)$		$q_3(2) = ...$		
...					

- To compute the score of a state, we have to consider all of the paths

$$q_2(1) = t(e_2|f_1) \times \sum_i p_1(i)a(1|i)$$

Summary of the Math

- Unnormalized score for a transition between two states

$$q_j(i \leftarrow i_{\text{previous}}) = t(e_j|f_i) \times a(i|i_{\text{previous}}) \times p_{j-1}(i_{\text{previous}})$$

- Normalization $$p_j(i \leftarrow i_{\text{previous}}) = \frac{q_j(i \leftarrow i_{\text{previous}})}{\sum_{i,i_{\text{previous}}} q_j(i \leftarrow i_{\text{previous}})}$$

- Probability of a state $$p_j(i) = \sum_{i_{\text{previous}}} p_j(i \leftarrow i_{\text{previous}})$$

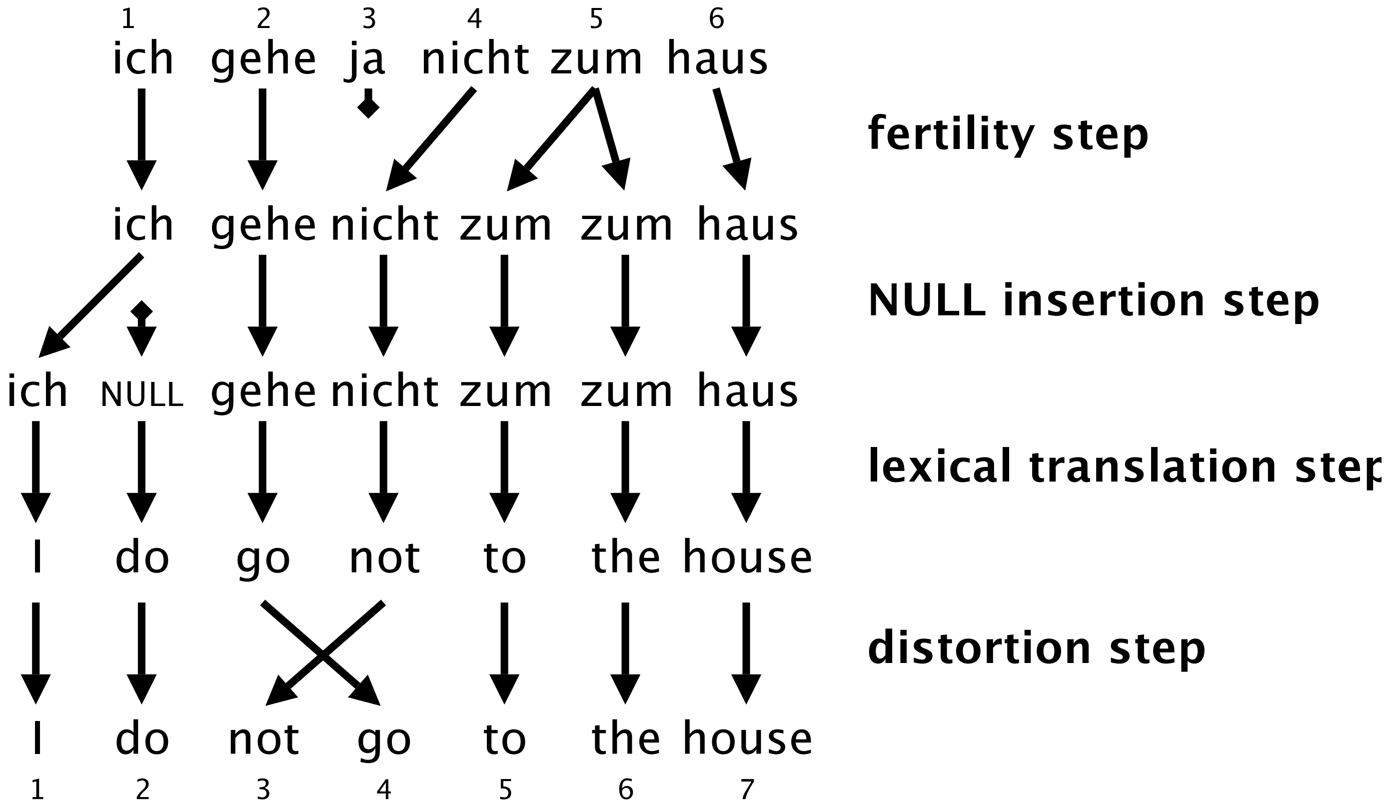
- Count collection $$c(e_j|f_i) = \sum_{i,j} p_j(i)$$

$$c(i|i_{\text{previous}}) = \sum_{i,j,i_{\text{previous}}} p_j(i \leftarrow i_{\text{previous}})$$

ibm model 3

IBM Model 3

Adding a model of fertilty



IBM Model 3: Fertility

- Fertility: number of English words generated by a foreign word
- Modelled by distribution $n(\phi|f)$
- Example:

$$n(1|\text{haus}) \simeq 1$$
$$n(2|\text{zum}) \simeq 1$$
$$n(0|\text{ja}) \simeq 1$$

Sampling the Alignment Space

- 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

IBM Model 4

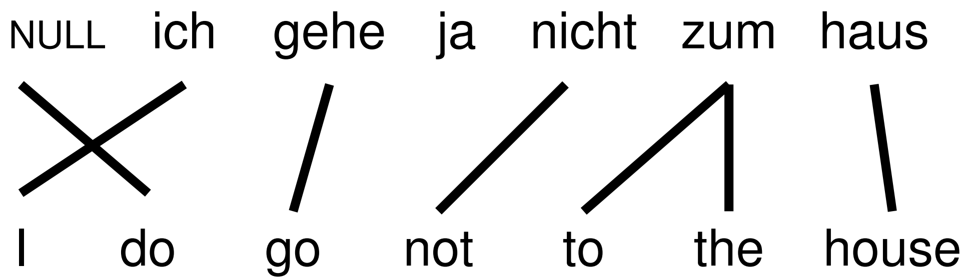
- Better reordering model

- Reordering in IBM Model 2 and 3
  - recall: $d(j|i, l_e, l_f)$
  - for large sentences (large l_f and l_e), sparse and unreliable statistics
  - phrases tend to move together

- Relative reordering model: relative to previously translated words (cepts)

IBM Model 4: Cepts

Foreign words with non-zero fertility forms cepts
(here 5 cepts)



cept π_i	π_1	π_2	π_3	π_4	π_5
foreign position $[i]$	1	2	4	5	6
foreign word $f_{[i]}$	ich	gehe	nicht	zum	haus
English words $\{e_j\}$	I	go	not	to,the	house
English positions $\{j\}$	1	4	3	5,6	7
center of cept $\odot_i$	1	4	3	6	7

IBM Model 4: Relative Distortion

j	1	2	3	4	5	6	7
e_j	I	do	not	go	to	the	house
in cept $\pi_{i,k}$	$\pi_{1,0}$	$\pi_{0,0}$	$\pi_{3,0}$	$\pi_{2,0}$	$\pi_{4,0}$	$\pi_{4,1}$	$\pi_{5,0}$
$\odot_{i-1}$	0	-	4	1	3	-	6
$j - \odot_{i-1}$	$+1$	-	-1	$+3$	$+2$	-	$+1$
distortion	$d_1(+1)$	1	$d_1(-1)$	$d_1(+3)$	$d_1(+2)$	$d_{>1}(+1)$	$d_1(+1)$

- Center $\odot_i$ of a cept π_i is ceiling(avg(j))

- Three cases:
  - uniform for NULL generated words
  - first word of a cept: d_1
  - next words of a cept: $d_{>1}$

Word Classes

- Some words may trigger reordering $\rightarrow$ condition reordering on words

$$\text{for initial word in cept:} \quad d_1(j - \odot_{[i-1]} | f_{[i-1]}, e_j)$$
$$\text{for additional words:} \quad d_{>1}(j - \Pi_{i,k-1} | e_j)$$

- Sparse data concerns $\rightarrow$ cluster words into classes

$$\text{for initial word in cept:} \quad d_1(j - \odot_{[i-1]} | \mathcal{A}(f_{[i-1]}), \mathcal{B}(e_j))$$
$$\text{for additional words:} \quad d_{>1}(j - \Pi_{i,k-1} | \mathcal{B}(e_j))$$

IBM Model 5

- IBM Models 1–4 are *deficient*
  - some impossible translations have positive probability
  - multiple output words may be placed in the same position
  - $\rightarrow$ probability mass is wasted

- IBM Model 5 fixes deficiency by keeping track of vacancies (available positions)

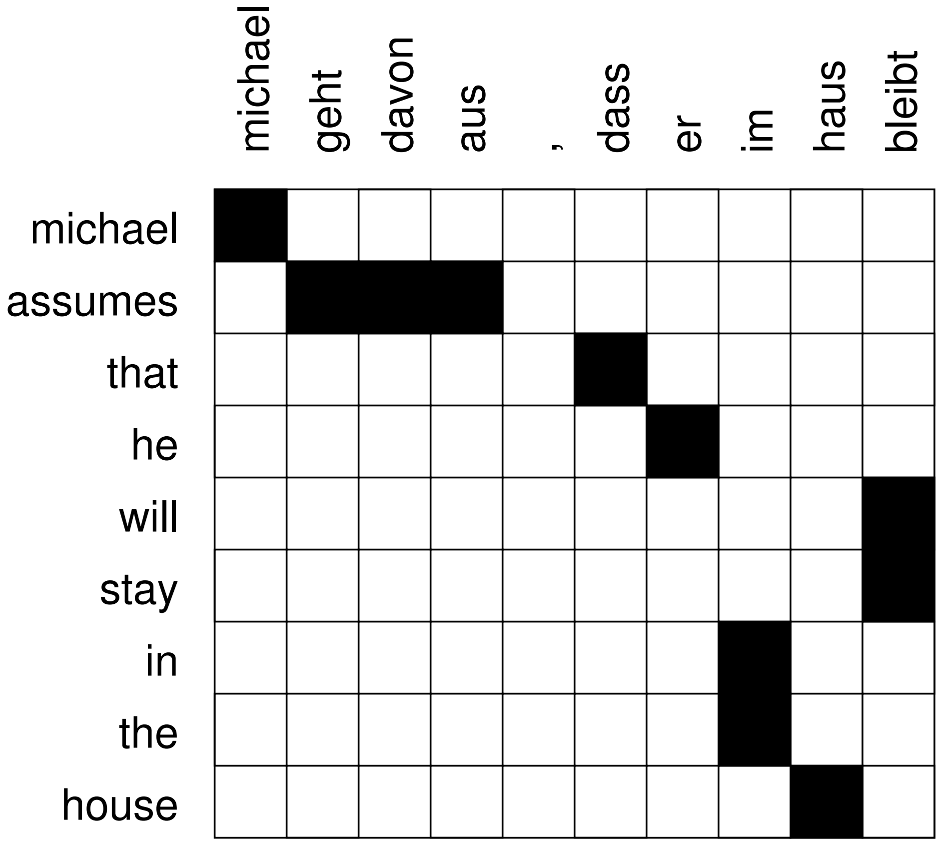
Conclusion

- 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++ toolkit)

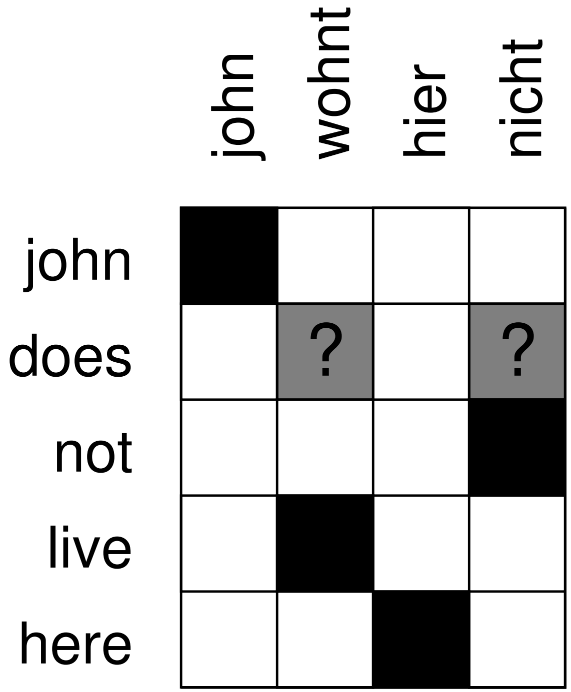
word alignment

Word Alignment

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

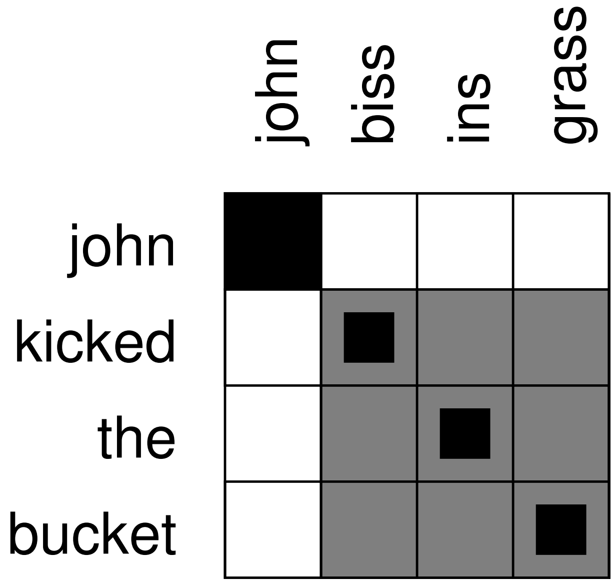


Word Alignment?



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

Word Alignment?



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

- 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

symmetrization

Word Alignment with IBM Models

- 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

Symmetrization

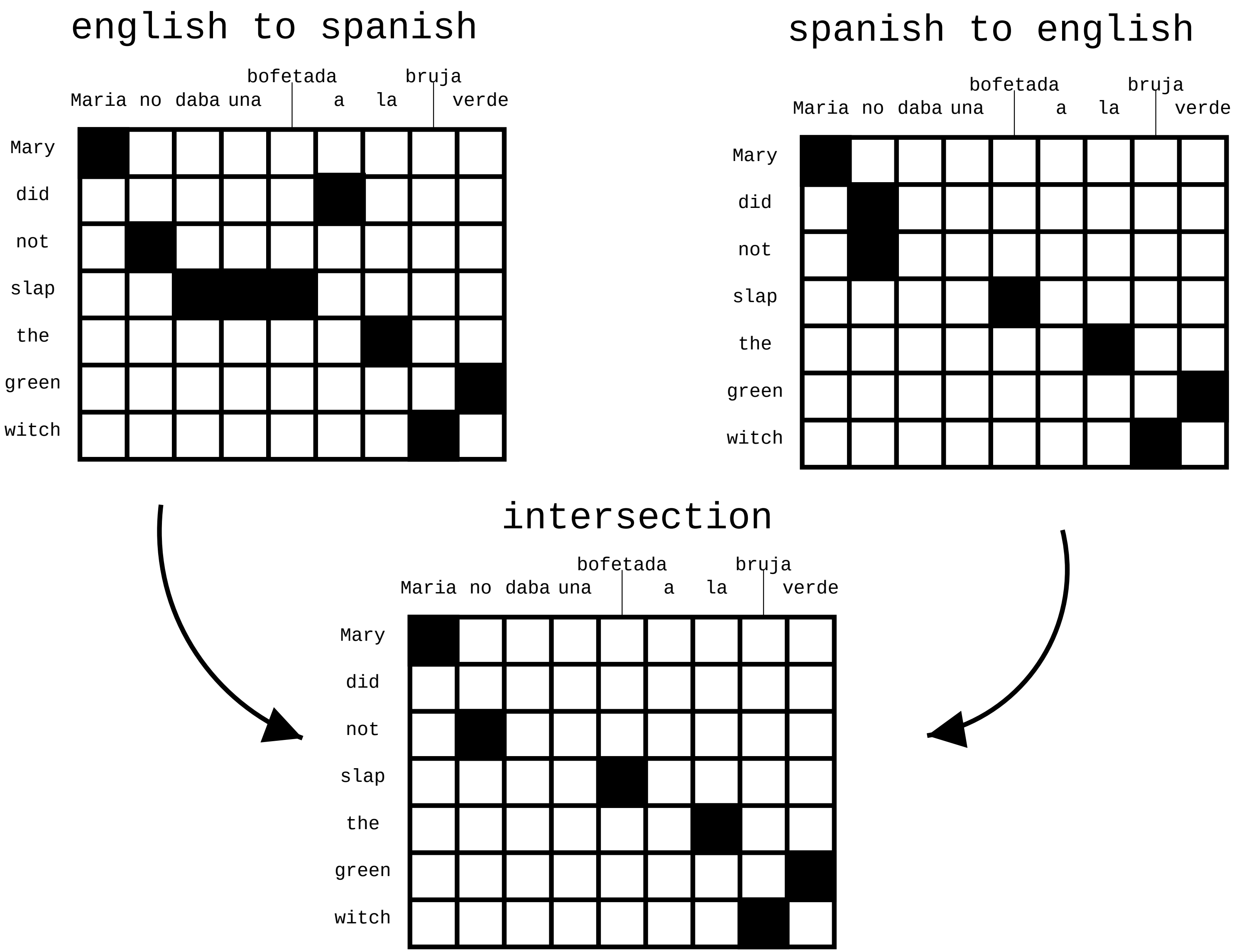
- Run IBM Model training in both directions

→ two sets of word alignment points

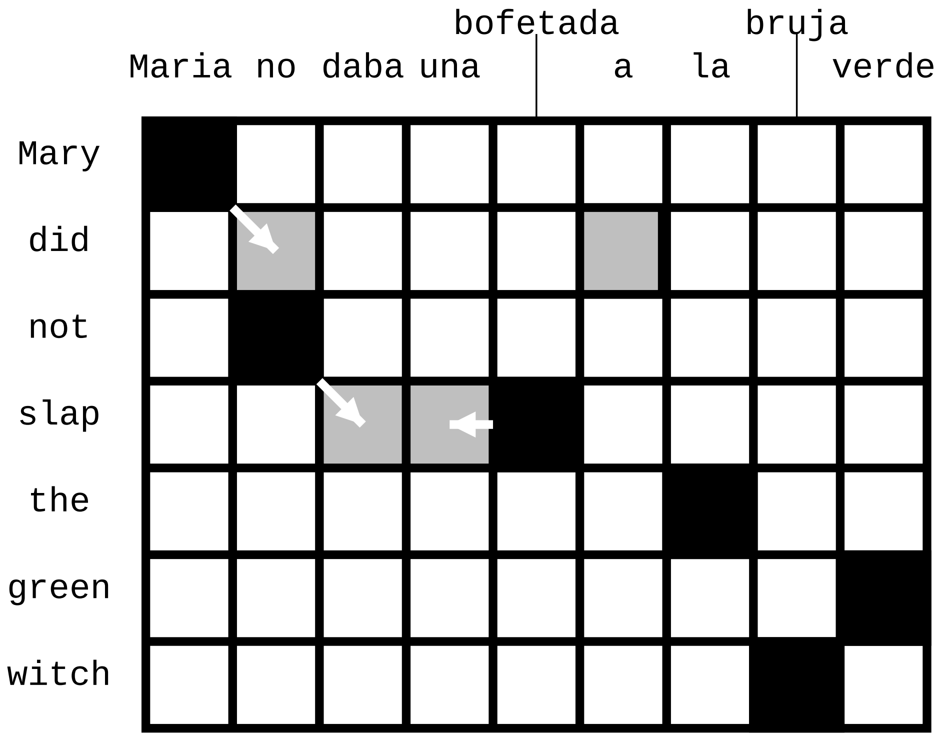
- Intersection: high precision alignment points
- Union: high recall alignment points

- Refinement methods explore the sets between intersection and union

Example


english to spanish
spanish to english
intersection
Maria no daba una bofetada a la bruja verde
Mary
did
not
slap
the
green
witch

Growing Heuristics



black: intersection **grey**: additional points in union

- Add alignment points from union based on heuristics:
  - directly/diagonally neighboring points
  - finally, add alignments that connect unaligned words in source and/or target
- Popular method: grow-diag-final-and

Growing heuristic

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 Work on Symmetrization

- 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

- 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

- 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

- Aligning phrases
  - joint model [Marcu and Wong, 2002]
  - problem: EM algorithm likes really long phrases
- Fraser's LEAF
  - decomposes word alignment into many steps
  - similar in spirit to IBM Models
  - includes step for grouping into phrase
- Riesa's NILE
  - use syntactic parse trees to guide word alignment
  - build up words bottom up following the parse tree

Final Remarks

- Research on word alignment has recently picked up again
  - speed matters
  - incremental (”online”) training

- Unclear link betwwn
  - word alignment quality measured against manual gold standard
  - impact on machine translation quality

- Advice: if you develop method, make easy-to-use toolkit available