PHRASE-BASED MACHINE TRANSLATION

David Talbot

22nd April, 2017

Computer Science Club, St. Petersburg, Russia

bayes' decision rule

Given foreign sentence $\mathbf{f}$ and a set of possible translations E, choose translation $\mathbf{e}^*$ s.t.

$$\begin{aligned} \mathbf{e}^* &= \operatorname*{argmax}_{\mathbf{e} \in E} \Pr(\mathbf{e}|\mathbf{f}) \\ &= \operatorname*{argmax}_{\mathbf{e} \in E} \Pr(\mathbf{e})\Pr(\mathbf{f}|\mathbf{e}) \end{aligned}$$

Why might the second line be easier to deal with?

source-channel model

$$\mathbf{e}^* = \underset{\mathbf{e} \in E}{\operatorname{argmax}} \Pr(\mathbf{e}) \Pr(\mathbf{f}|\mathbf{e})$$

- $\Pr(\mathbf{e})$ models the *fluency* of the translation
- $\Pr(\mathbf{f}|\mathbf{e})$ models the *adequacy* of the translation
- argmax is the search problem implemented by a *decoder*

Modelling $\Pr(\mathbf{e}|\mathbf{f})$ directly, we would need to handle fluency and adequacy simultaneously which is hard.

modelling fluency: language models

- Language models $\Pr(e)$ help us choose translations that sound good in the target language.
- Goal 1: Assign high probability to well formed candidates:
  - ”The cat in the hat.”
  - ”Green eggs and ham.”
- Goal 2: Assign low probability to malformed candidates:
  - ”Cat the hat in the.”
  - ”Eggs ham green and.”

n-gram language models: markov assumption

"I don't need to remember everything to predict the next ..."

- N-gram models assume each word is conditionally independent given previous $n-1$ words, e.g.

$$\Pr(\mathbf{e}) \approx \prod_i \Pr(e_i | e_{i-1}, \mathbf{e}_{i-2})$$

- What parameters does this model have?
- How could we estimate them?
- What problems will we have with this model?

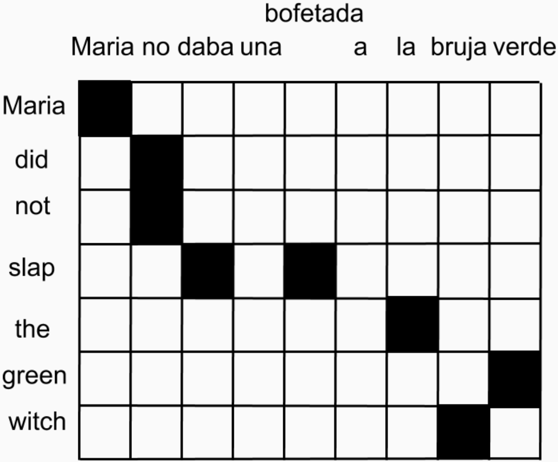
modelling adequacy: translation models

- Not so obvious how to factorize $\Pr(f|e)$
- Would be easier if we could see how the translator worked...
- IBM researchers introduced *word alignments* (1990)

modelling adequacy: translation models

- Alignments provide a *generative* story for the data
- Source words *generate* target words aligned to them
- Alignments can be one-to-one, one-to-many, many-to-one

word alignment matrix



How well can this model represent the data ?

- Choose $a_1 = 1$, generate "*Maria*" given "*Maria*"
- Choose $a_2 = 3$, generate "*no*" given "*not*"
- Choose $a_3 = 2$, generate "*daba*" given "*did*" ...

Maria (did) not slap the green witch.
Maria no daba una bofetada (a) la bruja verde.

word alignments: models 1, 2 and hmm

These models differ only in the prior over alignments

- IBM Model 1 (uniform)

$$\Pr(a_j = i|\mathbf{e}) \approx \epsilon$$

- IBM Model 2 (independent with positional bias)

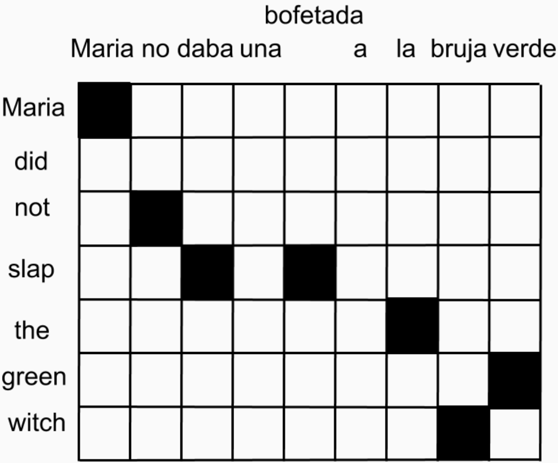
$$\Pr(a_j = i|\mathbf{e}) \approx p(a_j = i|j, I, J)$$

- HMM (Markov dependency with relative bias)

$$\Pr(a_j = i|\mathbf{e}) \approx h(a_j = i|a_{j-1} = i', I, J)$$

word alignments: models 1, 2 and hmm

Only one source word aligned to each target word



New generative story

1. Choose how many target words ϕ_i to generate from each source word e_i
2. Choose whether to insert NULL token
3. Choose how to order each group of words
4. Choose list of target words τ to generate

Why is not possible to train this model with full EM?
What other algorithms could we use?

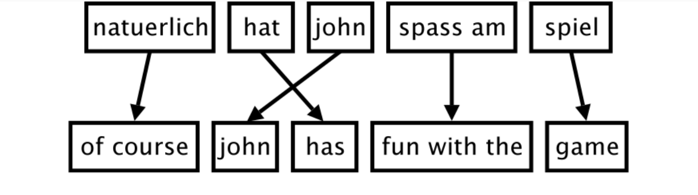
ibm model 3


Maria did not slap the green witch
1. Choose ‘fertility’
Maria not slap slap slap the green witch
2. Insert NULL
Maria not slap slap slap NULL the green witch
3. Reorder
Maria not slap slap slap NULL the witch green
4. Generate
Maria no daba una bofetada a la bruja verde

problems with word based models

- Word based models are still used for alignment
- Rarely used for translation
- They make unrealistic independence assumptions
- Translations don't consider context
- Reordering model is very weak
- Generating the target sentence requires many steps

phrase based machine translation (koehn 2003)



- Sentence split up into contiguous phrases
- Phrases can be reordered (as units)

phrase based machine translation (koehn 2003)

1. Estimate translation probabilities for *phrases* extracted from word aligned data
2. Add *feature functions* for *length* and *reordering*
3. Decode using a simple stack based algorithm

Basis for popularization of MT (Google, Yandex, Bing)

Translation involves the search for $\mathbf{e}^*$

$$\begin{aligned} \mathbf{e}^* &= \operatorname*{argmax}_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) & (1) \\ &= \operatorname*{argmax}_{\mathbf{e}} \Pr(\mathbf{f}|\mathbf{e}) \Pr_{LM}(\mathbf{e}) \omega^{length_{\mathbf{e}}} & (2) \end{aligned}$$

where ω is a new free parameter.

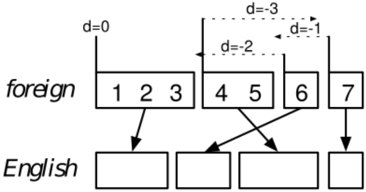
phrase based machine translation

Translation model is over phrases $(\hat{e}, \hat{f})$ rather than words

$$\Pr(\mathbf{f}|\mathbf{e}) = \prod_i \phi(\hat{f}_i|\hat{e}_i) d(a_i - b_{i-1})$$

Here a_i is the start index of the source phrase chosen as the i-th phrase to translate and b_{i-1} is end index of the previously translated source phrase.

distance-based reordering model



phrase	translates	movement	distance
1	1–3	start at beginning	0
2	6	skip over 4–5	+2
3	4–5	move back over 4–6	-3
4	7	skip over 6	+1

Reordering penalty is exponential in distance (usually)

$$d(a_i - b_{i-1}) = \alpha^{d(a_i - b_{i-1})}$$

phrase based machine translation

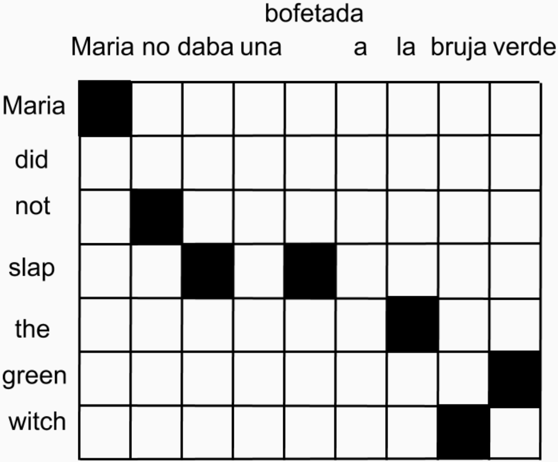
Translation model $\phi(\cdot)$ is estimated from extracted phrases

$$\phi(\hat{f}|\hat{e}) = \frac{count(\hat{f}, \hat{e})}{\sum_{\hat{f}'} count(\hat{f}', \hat{e})}$$

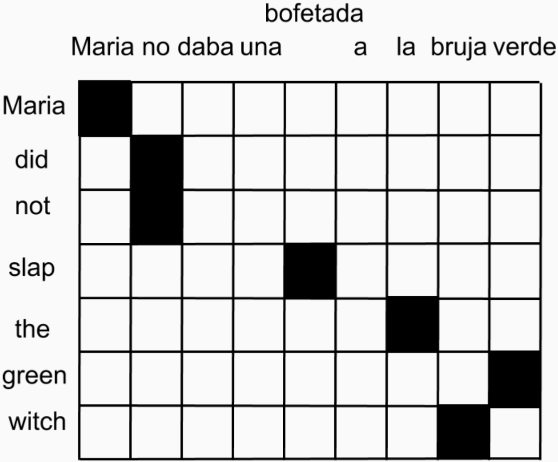
word alignment matrix

	Maria	no	daba	una	bofetada	a	la	bruja	verde
Maria									
did									
not									
slap									
the									
green									
witch									

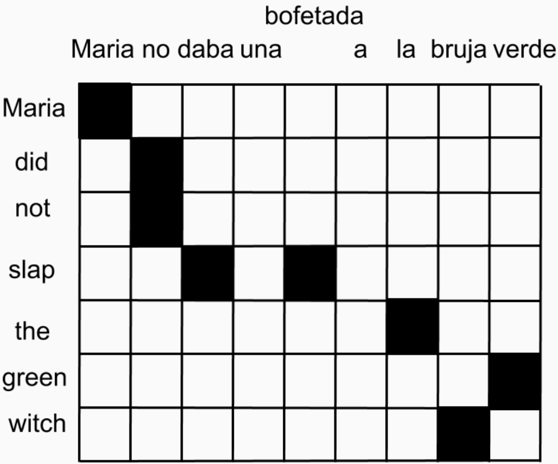
word alignment matrix: pr(f|e)



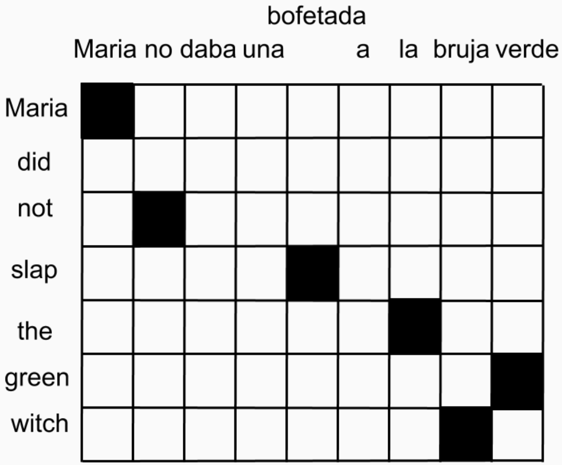
word alignment matrix: pr(e|f)

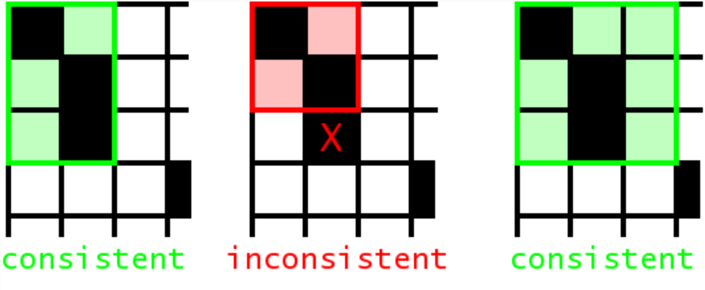


word alignment matrix: union



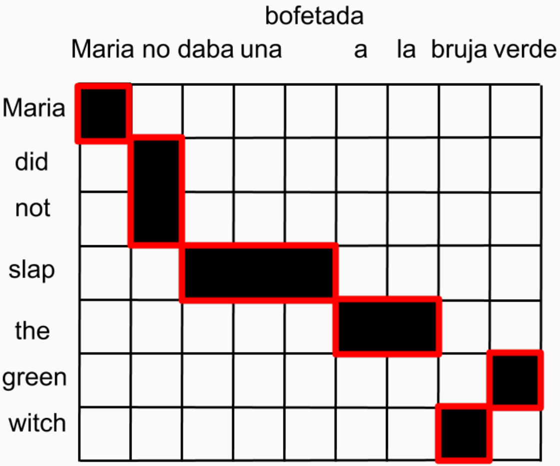
word alignment matrix: intersection



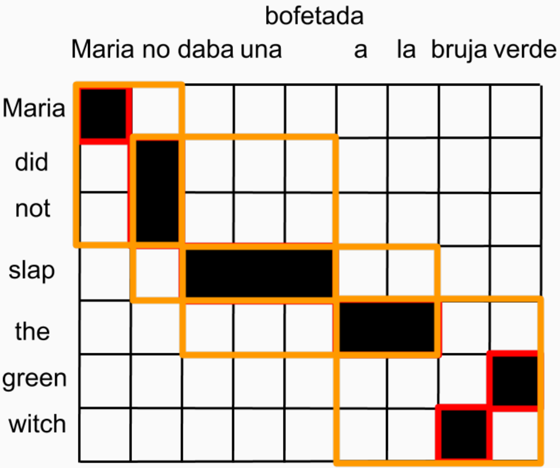


Word alignments constrain the set of possible phrase pairs.

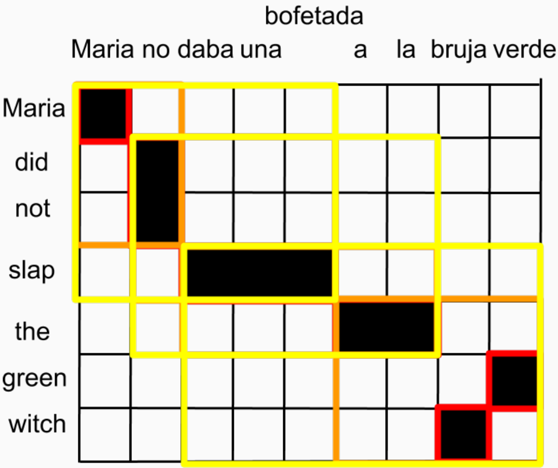
word alignment matrix: initial phrases



word alignment matrix: extensions



word alignment matrix: extensions



- Intersection: high confidence but sparse
- Union: more direct phrases, but also more constraints
- Null aligned words aren't a huge problem
- Many different ways of segmenting the translation
- Not a generative model

Which will produce the most phrase pairs?

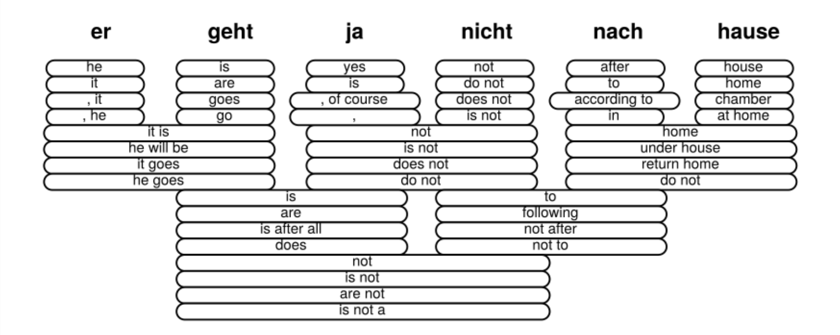
decoding

Find the best target sentence $\mathbf{e}^*$ s.t.

$$\mathbf{e}^* = \operatorname{argmax}_{\mathbf{e}} \prod_i \phi(\hat{\mathbf{f}}_i|\hat{\mathbf{e}}_i) d(a_i, b_{i-1}) \mathrm{Pr}_{LM}(\mathbf{e}) \omega^{length_{\mathbf{e}}}$$

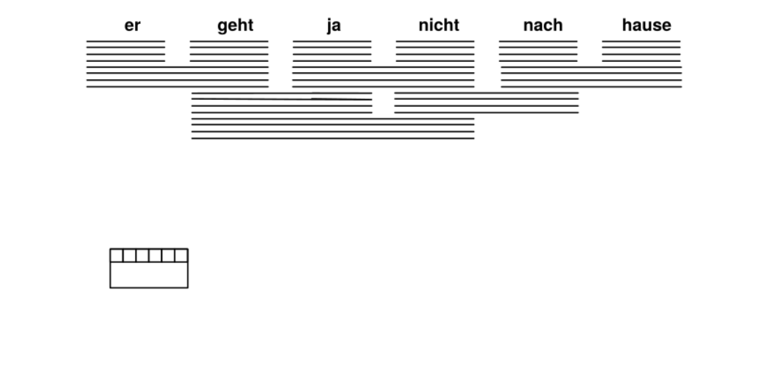
- Apply translation cost (cached for sentence pair)
- Apply reordering cost based on distance
- Apply language model and length costs

phrase based decoding



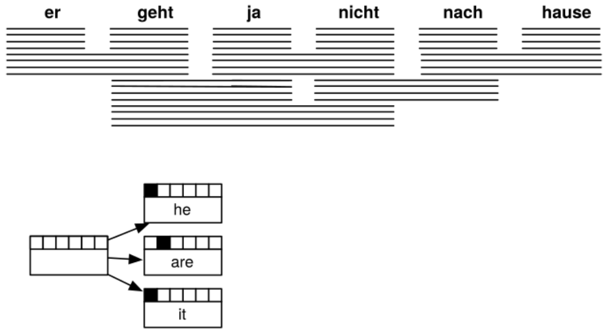
Retrieve translation options for sentence

phrase based decoding



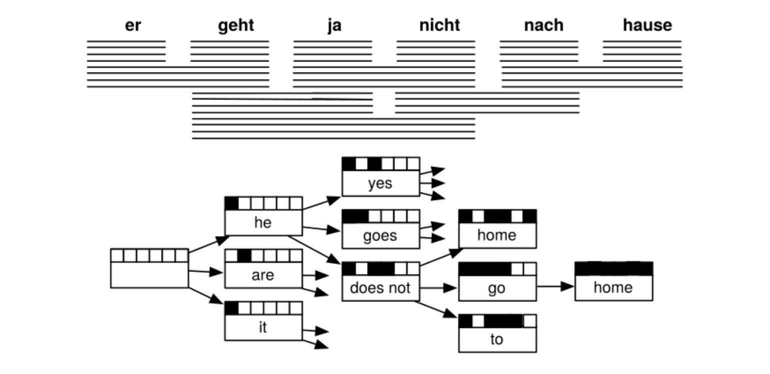
Initialize empty hypothesis

phrase based decoding



Candidates that translate one word

phrase based decoding



Expand until all source words are translated

decoding complexity

- Exponential in sentence length
- How can we compare partial hypotheses?
- How should we compare hypotheses that cover different source words?
- What methods could we use to make this work?

Makes neural machine translation look easy :)