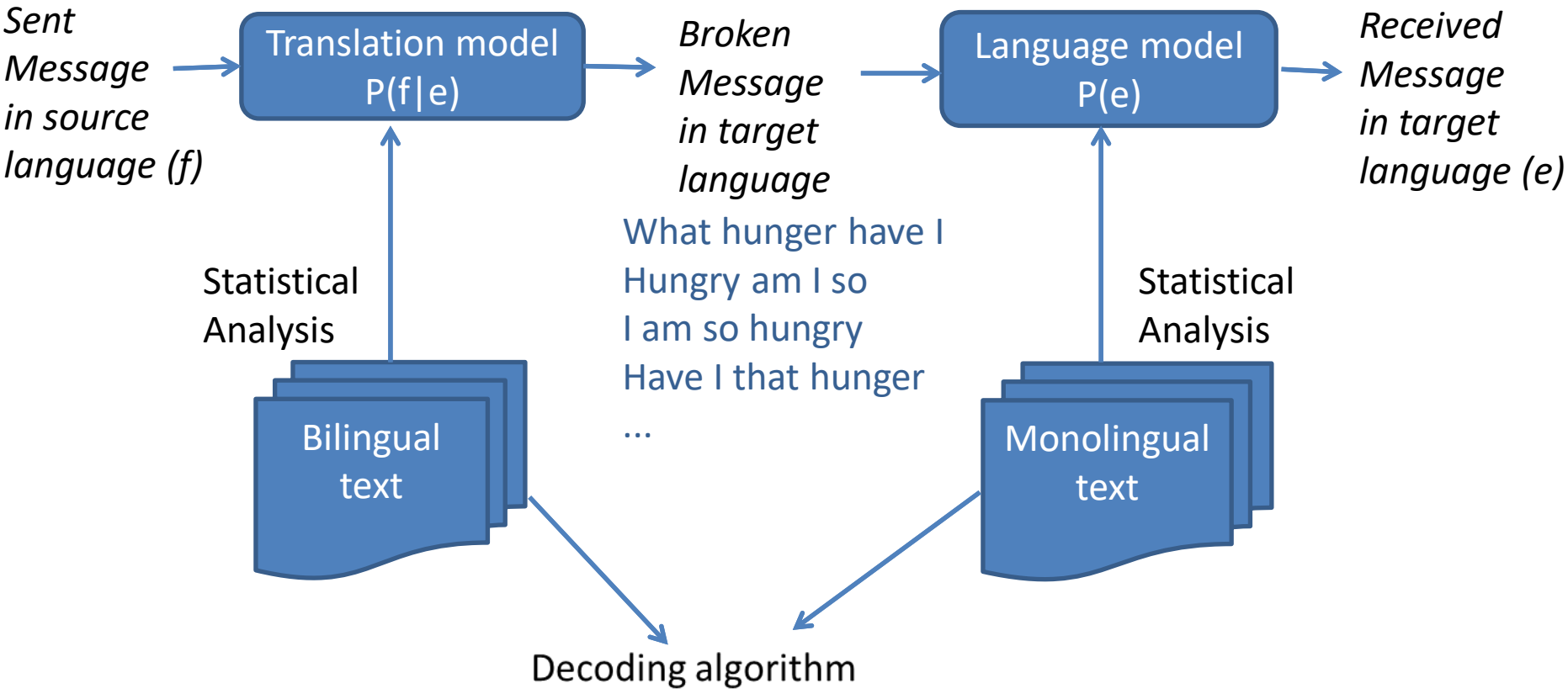
Machine Translation
Course 3

Diana Trandabăț

Academic year: 2022-2023

Noisy Channel Model



$$\hat{e} = argmax_e\, P(e|f) = argmax_e \frac{P(f|e) * P(e)}{P(f)} = argmax_e P(f|e) * P(e)$$

Ce foame am

I am so hungry

Three problems in Statistical MT

- Language Model
  - given an target language string *e*, assign *P(e)*
  - correct target language string => high *P(e)*
  - wrong target language string => low *P(e)*
- Translation Model
  - given a pair of strings <*f, e*>, where *f* is a string in the source language and *e* is the string in target language, assign *P(f|e)*
  - <f, e> look like translations => high *P(f|e)*
  - <f, e> don't look like translations => low *P(f|e)*
- Decoding Algorithm
  - given a language model, a translation model and a new sentence *f*, find translation *e* maximizing *P(e)*P(f|e)*

Language Models - Introduction

- Basically counting words in corpora
- What is a corpus?
- What is a word?

What is a corpus?

- A corpus is a body of naturally occurring language (either written or spoken);
- “A corpus is a collection of (1) *machine-readable* (2) *authentic* texts (including transcripts of spoken data) which is (3) *sampled* to be (4) *representative* of a particular language or language variety.” (McEnery et al., 2006)

What is a word?

- Are cat and cats the same word?
- September and Sept?
- zero and oh?
- Is _ a word? * ?) . ,
- How many words are there in don't ? Gonna ?
- In Japanese and Chinese texts, how do we identify a word?

Word-based Language Models

- A model that enables one to compute the probability, or likelihood, of a sentence *S* as *P(S)*.
- *Simple case*: Every word follows every other word with equal probability (0-gram)
  - Assume |V| is the size of the vocabulary V
  - Likelihood of sentence *S* of length *n* is:

$$P(S) = \prod_{1}^{n} \frac{1}{V}$$

  - If English had 100,000 words, probability of each next word would be 1/100000 = .00001

Word Prediction: Simple vs. Smart

- *Smarter*: probability of each word is related to word frequency (unigram model)
  - Likelihood of sentence S = $P(w_1) \times P(w_2) \times \ldots \times P(w_n)$
  - Assumes probability of each word is independent of probabilities of other words.
- *Even smarter*: Look at probability *given* previous words (N-gram)
  - Likelihood of sentence S = $P(w_1) \times P(w_2|w_1) \times \ldots \times P(w_n|w_{n-1})$
  - Assumes probability of each word is dependent on probabilities of other words.

Estimating Probabilities

- N-gram conditional probabilities can be estimated from raw text based on the ***relative frequency*** of word sequences.

Bigram: $$P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

N-gram: $$P(w_n \mid w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})}$$

- To have a consistent probabilistic model, append a unique start (<s>) and end (</s>) symbol to every sentence and treat these as additional words.
- *P(new_sentence)* can be approximated by multiplying all the bigram probabilities in the sequence.

Exercise: Computing a Language Model

- Consider the following training corpus:

  there is a big house

  i buy a house

  they buy the new house

- Create the bigram language model for this corpus

- Compute the probability of the new sentence

  they buy a big house

Exercise: Computing a Language Model

- Consider the following training corpus:

  <s> there is a big house </s>

  <s> i buy a house </s>

  <s> they buy the new house </s>

- Create the bigram language model for this corpus

$P(\text{there}|\langle s\rangle) = 0.33$
$P(\text{is}|\text{there}) = 1$
$P(a|\text{is}) = 1$
$P(\text{big}|a) = 0.5$
$P(\text{house}|\text{big}) = 1$

$P(\langle/s\rangle|\text{house}) = 1$
$P(I|\langle s\rangle) = 0.33$
$P(\text{buy}|I) = 1$
$P(a|\text{buy}) = 0.5$
$P(\text{house}|a) = 0.5$

$P(\text{they}|\langle s\rangle) = 0.3$
$P(\text{buy}|\text{they}) = 1$
$P(\text{the}|\text{buy}) = 0.5$
$P(\text{new}|\text{the}) = 1$
$P(\text{house}|\text{new}) = 1$

- Compute the probability of the new sentence

  they buy a big house

$$P(S) = P(\text{they}|\langle s\rangle)*P(\text{buy}|\text{they})*P(a|\text{buy})*P(\text{big}|a)*P(\text{house}|\text{big})*P(\langle/s\rangle|\text{house}) =$$
$$= 0.33 * 1 * 0.5 * 0.5 * 1 * 1 = 0.082$$

Unknown Words

- How to handle words in the test corpus that did not occur in the training data, i.e. ***out of vocabulary*** words, or ***sparse data*** problem.
- If a new combination occurs during testing, it is given a probability of zero and the entire sequence gets a probability of zero.
- In practice, parameters are ***smoothed*** to reassign some probability values to unseen events.

Smoothing

- *Using UNK*: Train a model that includes an explicit symbol for an unknown word (<UNK>).
  - Choose a vocabulary in advance and replace other words in the training corpus with <UNK>.
  - Replace the first occurrence of each word in the training data with <UNK>.
- *Laplace (Add-one)*: "Hallucinate" additional training data in which each possible N-gram occurs exactly once and adjust estimates accordingly.

$$\textbf{Bigram:} \quad P(w_n \mid w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

where *V* is the total number of possible (N–1)-grams (i.e. the vocabulary size for a bigram model).

Smoothing

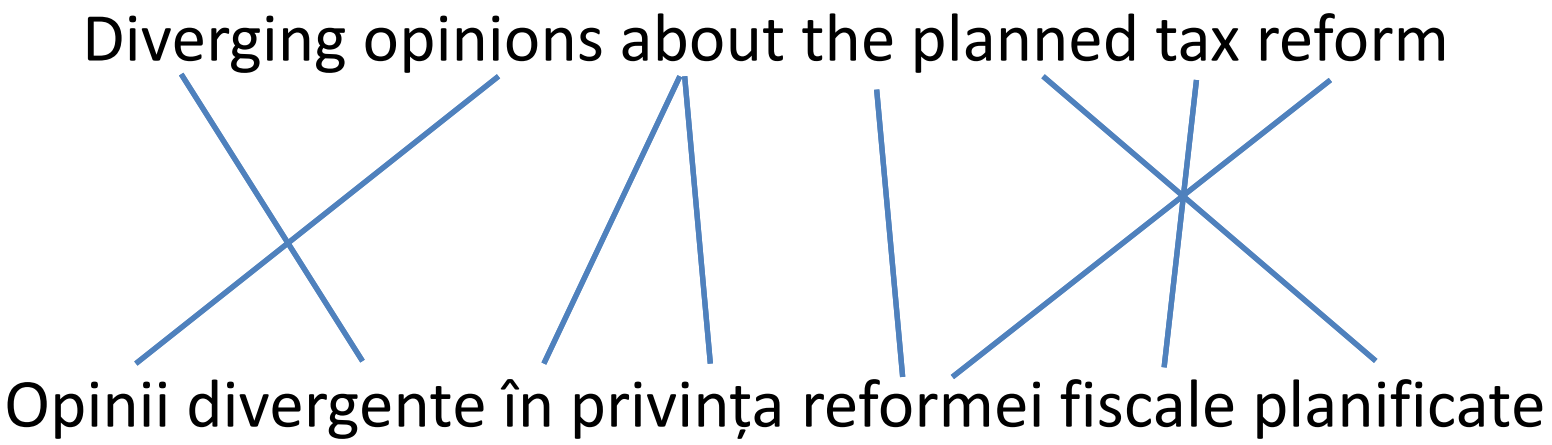
- Many advanced techniques have been developed to improve smoothing for language models.
  - Good-Turing
  - Interpolation
  - Backoff
  - Kneser-Ney
  - Class-based (cluster) N-grams
- **Entropy** (the expected negative log probability): Measuring how well a grammar or language model (LM) models a natural language or a corpus.
- **Perplexity:** Measure of how well a model “fits” the test data.

Translation Models - Introduction

- Determines the probability that the foreign word *f* is a translation of the English word *e*
- How to compute *P(f|e)* from a **parallel corpus**?
- Statistical approaches rely on the co-occurrence of *e* and *f* in the parallel data: If *e* and *f* tend to co-occur in parallel sentence pairs, they are likely to be translations of one another.

Word-Level Alignments

- Given a parallel sentence pair we can link (align) words or phrases that are translations of each other:



Different alignments: 1:1, 1:n, n:1, 1:0, 0:1, n:n

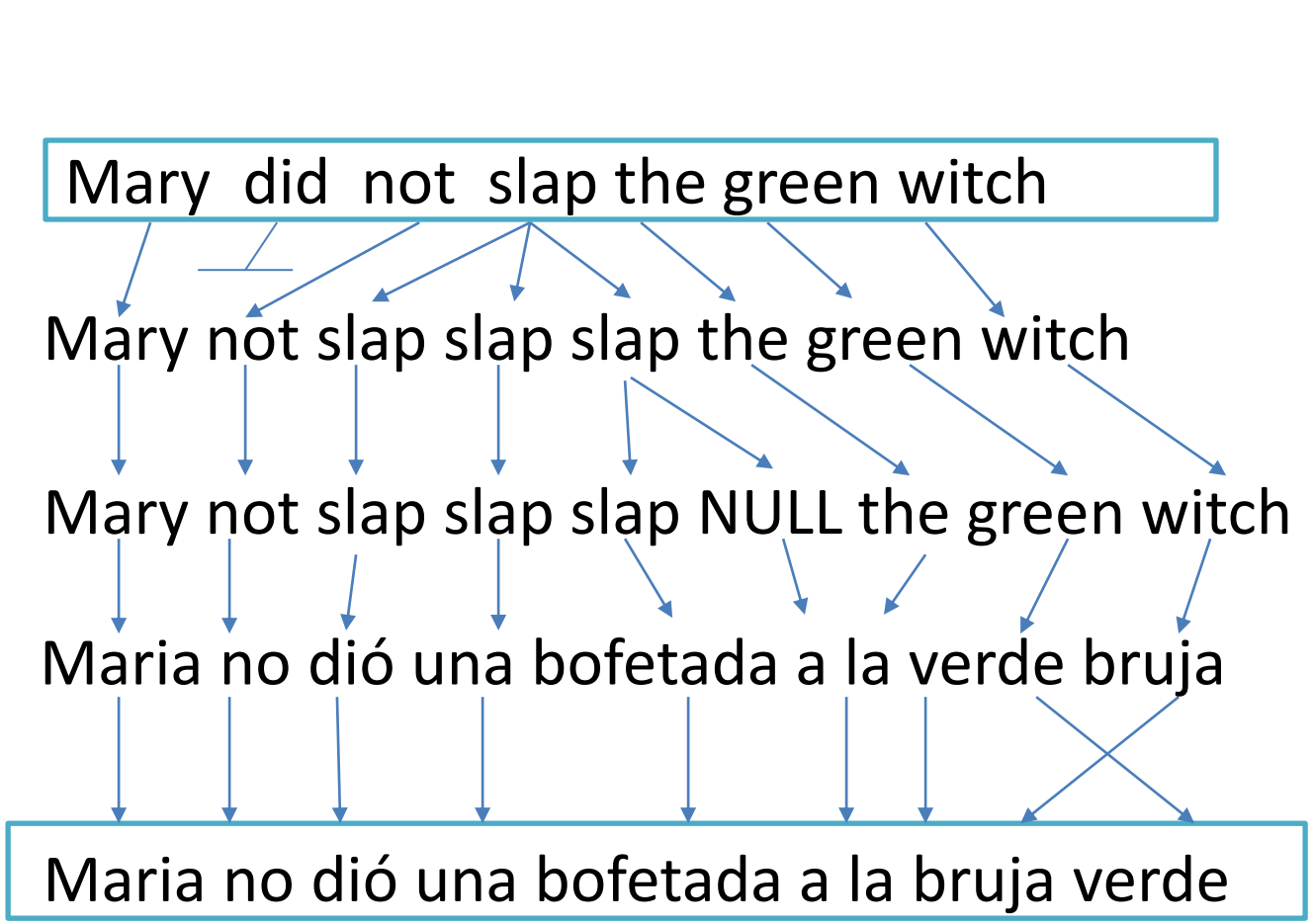
Finding Translations in a Parallel Corpus

- Into which foreign words *f, . . . , f'* does *e* translate?
- Commonly, four factors are used:
  - How often do *e* and *f* co-occur? (translation)
  - How likely is a word occurring at position *i* to translate into a word occurring at position *j*? (distortion)

    For example: English is a verb-second language, whereas German is a verb-final language
  - How likely is *e* to translate into more than one word? (fertility)

    For example: *remember* can translate into *a-și aduce aminte*
  - How likely is a foreign word to be spuriously generated? (null translation)

Translation Model



IBM Models 1–5

- Model 1: Lexical translation
- Model 2: Adds absolute reordering
- Model 3: Adds fertility
- Model 4: Adds relative distortion. The distortion factor determines how likely it is that an English word in position i aligns to a foreign word in position j, given the lengths of both sentences
- Model 5: Extra variables to avoid deficiency

- Training of a higher IBM model builds on previous models

IBM Model 1

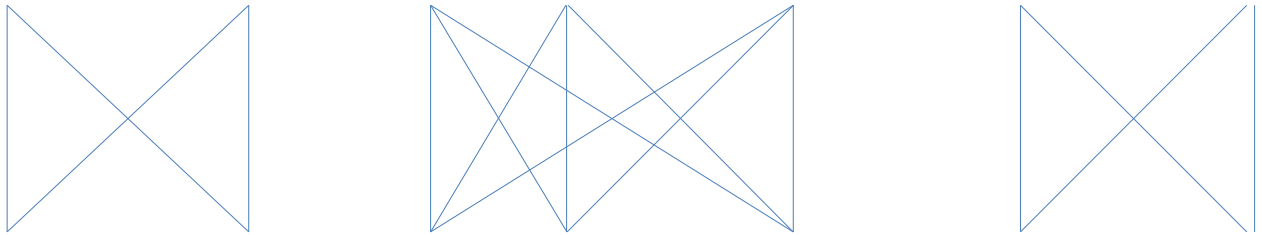
- Simplest of the IBM models
- Efficient Expectation Maximization algorithm
- Does not consider word order (bag-of-words approach)
- Computationally inexpensive
- Useful for parameter estimations that are passed on to more elaborate models

IBM Model 1 and EM

- EM Algorithm consists of two steps
- **Expectation-Step**: Apply model to the data
  - 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

- … la maison … la maison bleu … la fleur …

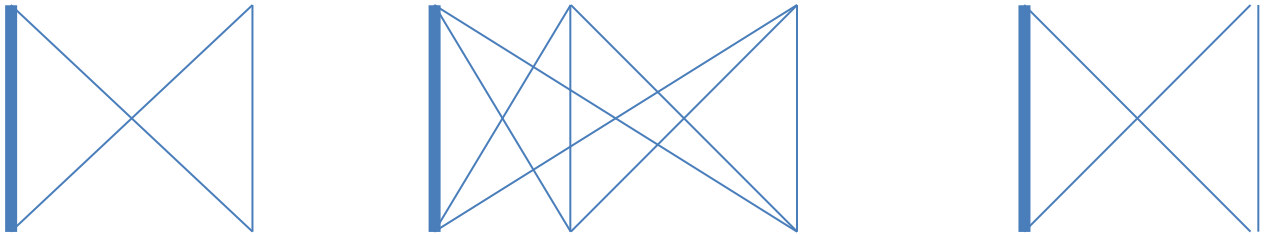


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

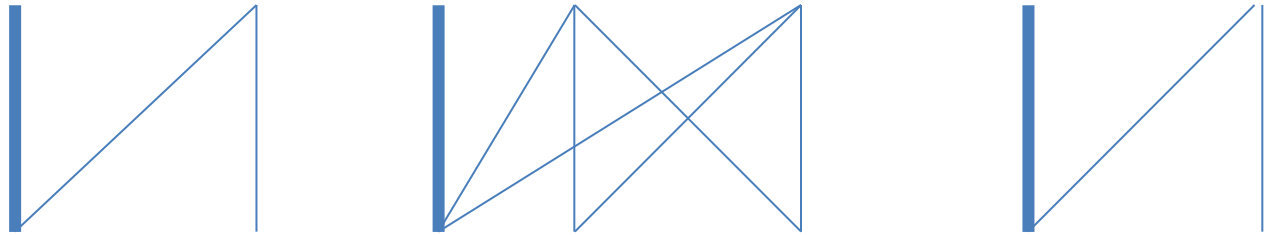


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

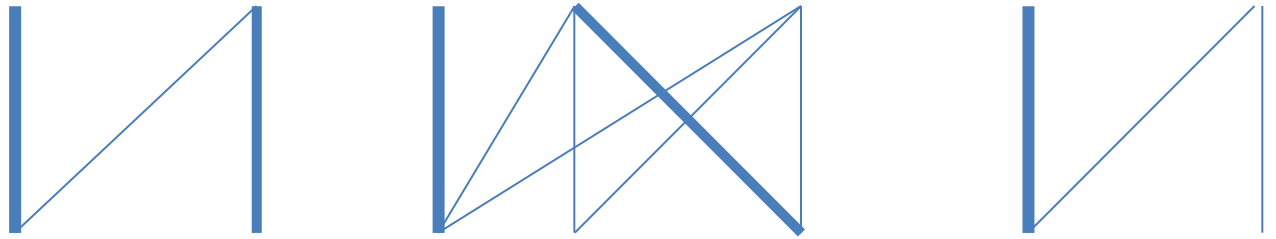


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

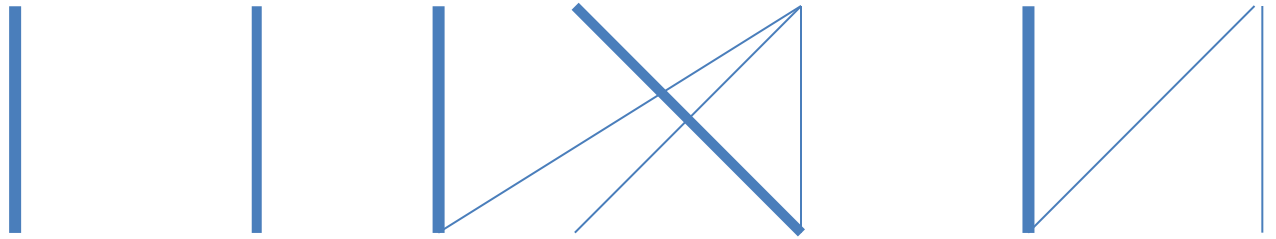


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

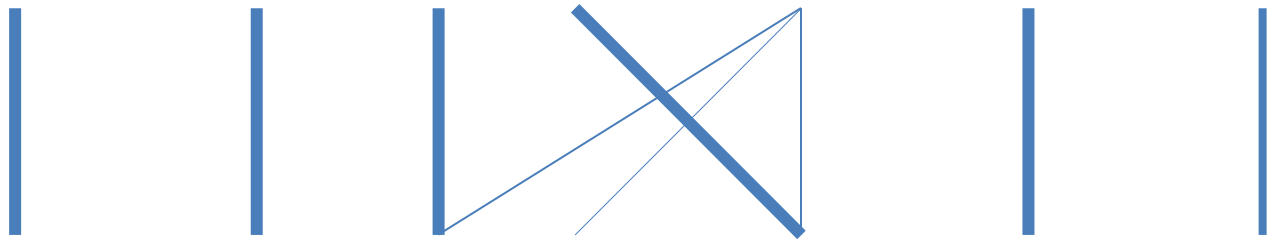


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

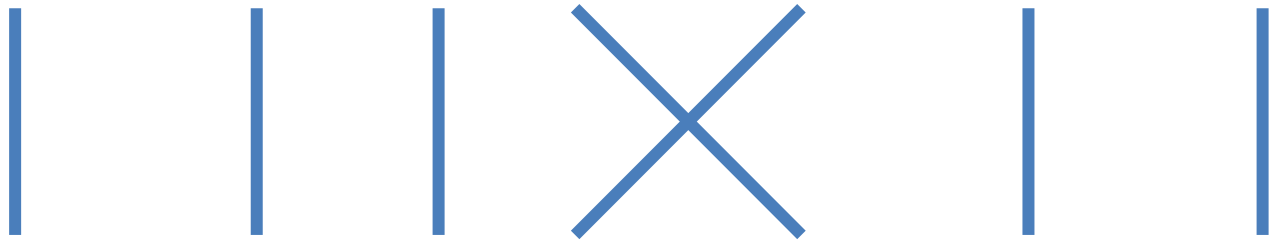


- … the house … the blue house … the flower

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

IBM Model 1

- … la maison … la maison bleu … la fleur …

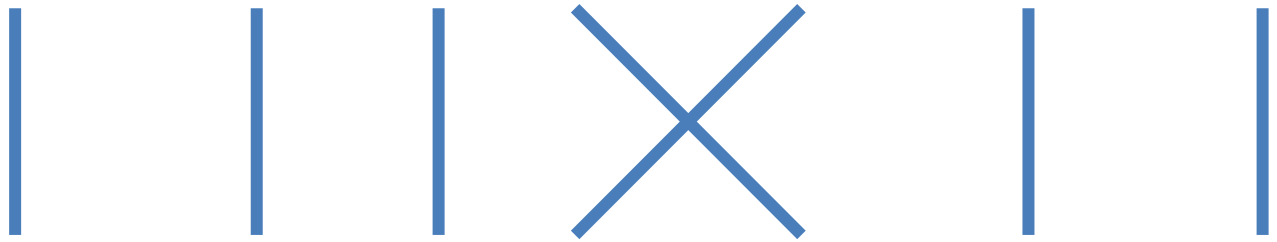


- … the house … the blue house … the flower

- Convergence
- Inherent hidden structure revealed by EM

IBM Model 1

- … la maison … la maison bleu … la fleur …



- … the house … the blue house … the flower

P(la|the)=0.453

P(le|the)=0.334

P(maison|house)=0.876

P(bleu|blue)=0.563

- Parameter estimation from aligned corpus

Limitations of IBM Models

- Only 1-to-N word mapping
- Handling fertility-zero words
- Almost no syntactic information
- Long-distance word movement
- Fluency of the output depends entirely on the language model for the target language

Decoding

- Probability models enable us to *make predictions*:
  - Given a particular source sentence, what is the most probable sentence corresponding to it in a target language?
- The decoder combines the evidence from *P(e)* and *P(f|e)* to find the sequence *e* that is the best translation:

$$\hat{e} = argmax_e\, P(e|f) = argmax_e P(f|e) * P(e)$$

- **Problem**: there are a lot of possible sentences to choose from!!!

Little story...

- A French speaking traveler equipped with a bilingual dictionary enters in a New-York store. While reviewing the price chart, he encounters a line that he does not understand:

```
A sheet of paper .. .. .. .. 0.25$
```

- We will look at a process this traveler could use to decode this strange sentence.

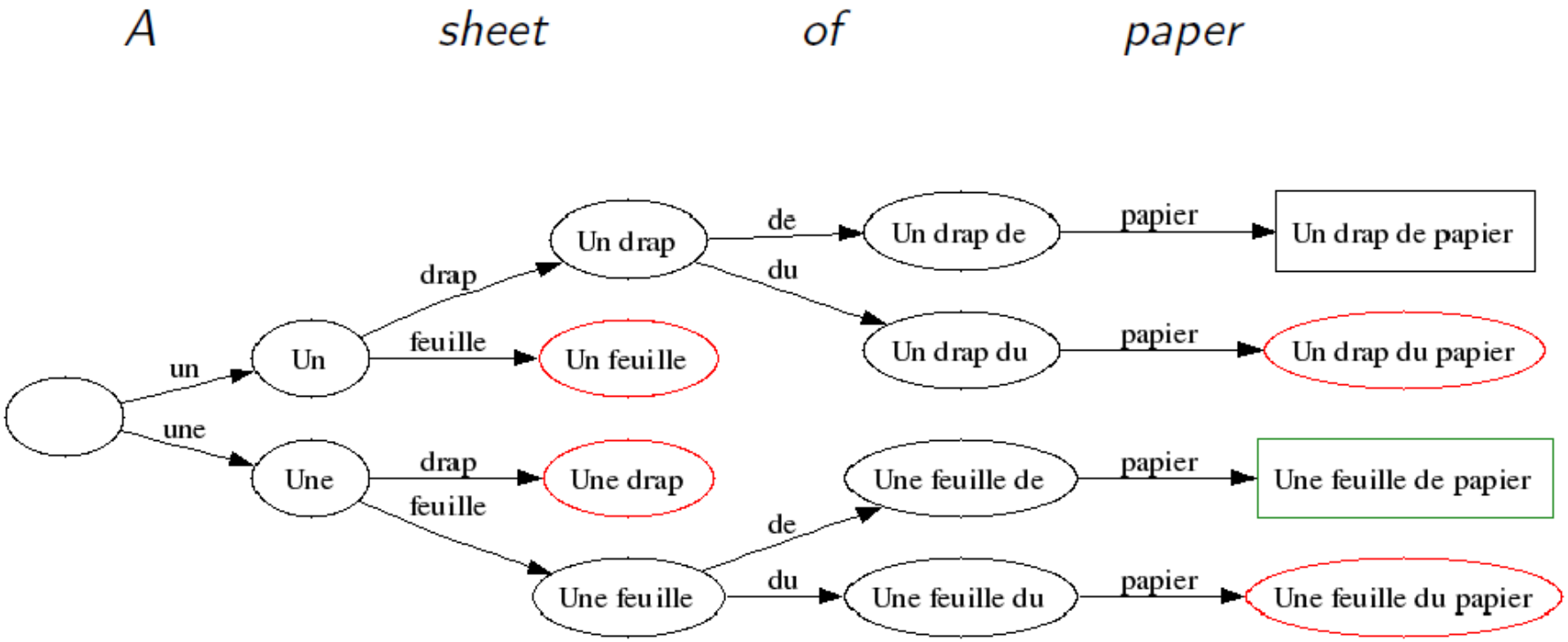
Resources

- Sentence to translate: `A sheet of paper`
- Bilingual dictionary

source	target	source	target
A	Un	of	de
A	Une	of	du
sheet	feuille	paper	papier
sheet	drap		

- *The traveler's common sense*: The traveller can intuitively evaluate the likelihood of a French sentence (since he is French).

Search graph



The traveler concludes that the most likely translation is *Une feuille de papier*.

Search space complexity

- A sentence containing 10 words having each 5 translations can be translated by more than 9 millions target sentences and the corresponding search graph has more than 12 millions vertices

$$\text{Translation} = 5^{10}$$

$$\text{Vertices} = \sum_{i=0}^{10} 5^{i}$$

- If we allow word reordering, the same sentence will have more than 35,000 billions translations and its search graph will contain more than 43,000 billions vertices.

Decoder

- A decoder searches the target document *e* having the highest probability to translate a given source document *f*.
- Most decoders assume that the sentences of a document are independent from each others. The decoder can thus translate each sentence individually.
- Shortcomings:
  - A sentence cannot be omitted, merged with another one, repositioned or sliced by the decoder.
  - The context of a sentence is not considered when it is translated.

The decoder's task

- This problem can be reformulated as a classic AI problem: searching for the shortest path in an implicit graph.
- Two independent problems must be resolved in order to build a decoder:
  - **Model representation** The model defines what a transformation is and how to evaluate the quality of a translation.
  - **Search space exploration** Enumerating all possible sequences of transformations is often impracticable, we must smartly select the ones that will be evaluated.

Conclusions

- The noisy channel model decomposes machine translation into two independent subproblems: Word alignment and Language modeling;
- IBM Models were the pioneering models in statistical machine translation;
- Important concepts: language model, translation model, decoder, n-gram, word and sentence alignment, reordering.

“One naturally wonders if the problem of translation could conceivably be treated as a problem in cryptography.

When I look at an article in Russian, I say: ‘This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode.’ ”

Warren Weaver (1947)

Further reading

- Hutchins, J. *Machine translation: general overview*. Chapter 27 of R Mitkov (ed.) The Oxford Handbook of Computational Linguistics, Oxford (2004)
- Somers, H. *Machine Translation*. Chapter 13 of R Dale, H Moisl & H Somers (eds) Handbook of Natural Language Processing, New York (2000): Marcel Dekker
- Peter Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Robert L. Mercer (1993) *The Mathematics of Statistical Machine Translation: Parameter Estimation*. Computational Linguistics 19:2, 263-311