Introduction to Machine Translation and Phrase-Based Machine Translation

Aleš Tamchyna

Charles University in Prague

September 8, 2015

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

- Neural networks.

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

- Neural networks.
  Wednesday 12:00: Neural Network Models in Google Translate (Keith Stevens)

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

- Neural networks.
  Wednesday 12:00: Neural Network Models in Google Translate (Keith Stevens)
- Deep (dependency) syntax.

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

- Neural networks.
  Wednesday 12:00: Neural Network Models in Google Translate (Keith Stevens)
- Deep (dependency) syntax.
  Thursday 9:00: Deep Syntactic MT and TectoMT (Martin Popel)

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

- Neural networks.
  Wednesday 12:00: Neural Network Models in Google Translate (Keith Stevens)
- Deep (dependency) syntax.
  Thursday 9:00: Deep Syntactic MT and TectoMT (Martin Popel)
- Constituency syntax.

Other Approaches to Machine Translation

Our topic: **phrase-based MT**.
(Some) other approaches:

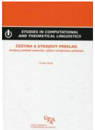
- Neural networks.
  Wednesday 12:00: Neural Network Models in Google Translate (Keith Stevens)
- Deep (dependency) syntax.
  Thursday 9:00: Deep Syntactic MT and TectoMT (Martin Popel)
- Constituency syntax.
  Friday 13:30: Syntax Extraction and Decoding (Hieu Hoang)

Where to Find More?

Books:

Where to Find More?

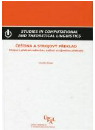
Books:



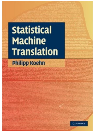
Ondřej Bojar: Čeština a strojový překlad

Where to Find More?

Books:



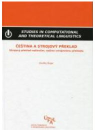
Ondřej Bojar: Čeština a strojový překlad



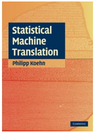
Philipp Koehn: Statistical Machine Translation

Where to Find More?

Books:



Ondřej Bojar: Čeština a strojový překlad



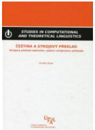
Philipp Koehn: Statistical Machine Translation

Online:

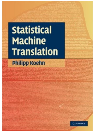
- `http://www.statmt.org/`

Where to Find More?

Books:



Ondřej Bojar: Čeština a strojový překlad



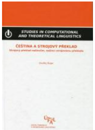
Philipp Koehn: Statistical Machine Translation

Online:

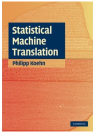
- http://www.statmt.org/
- http://www.statmt.org/moses/

Where to Find More?

Books:



Ondřej Bojar: Čeština a strojový překlad



Philipp Koehn: Statistical Machine Translation

Online:

- http://www.statmt.org/
- http://www.statmt.org/moses/
- http://mttalks.ufal.cz/

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?
- $P(A \cap B)$ or $P(A, B)$... Joint probability (both A and B occur).

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?
- $P(A \cap B)$ or $P(A, B)$... Joint probability (both A and B occur).
- $P(A|B)$ ... Probability of event A given that B occurred.

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?
- $P(A \cap B)$ or $P(A, B)$... Joint probability (both A and B occur).
- $P(A|B)$ ... Probability of event A given that B occurred.
  Given that I see clouds (B), what is the chance it will rain today (A)?

Probability – Quick Refresher

- $P(A) \in [0, 1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?
- $P(A \cap B)$ or $P(A, B)$... Joint probability (both A and B occur).
- $P(A|B)$ ... Probability of event A given that B occurred.
  Given that I see clouds (B), what is the chance it will rain today (A)?

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Probability – Quick Refresher

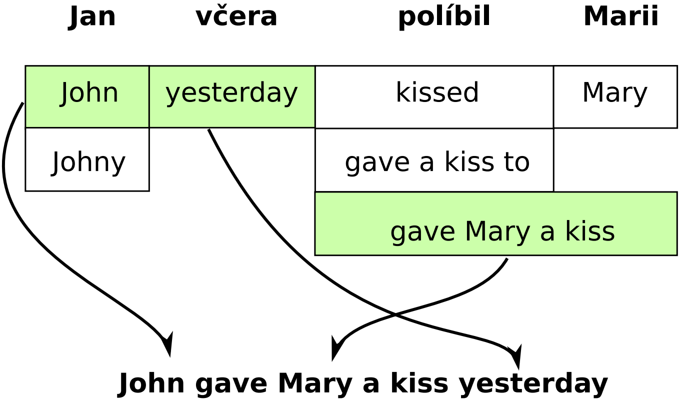
- $P(A) \in [0,1]$ ... Probability of event A.
  E.g. what is the chance it will rain today?
- $P(A \cap B)$ or $P(A, B)$... Joint probability (both A and B occur).
- $P(A|B)$ ... Probability of event A given that B occurred.
  Given that I see clouds (B), what is the chance it will rain today (A)?

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- Bayes' Theorem (inverse probability):

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Our Goal: Phrase-Based Machine Translation



The Essential Ingredient

Our Own Parallel Corpus

žlutý	the		
byl	parrot	žlutý	yellow
ten	was	pes	dog
papoušek	yellow		

Our Own Parallel Corpus

žlutý	the		
byl	parrot	žlutý	yellow
ten	was	pes	dog
papoušek	yellow		

What does "žlutý" mean in English?

Our Own Parallel Corpus

žlutý	the		
byl	parrot	**žlutý**	yellow
ten	was	pes	dog
papoušek	yellow		

What does "žlutý" mean in English?

Our Own Parallel Corpus

žlutý	the
byl	parrot
ten	was
papoušek	**yellow**

žlutý	**yellow**
pes	dog

What does "žlutý" mean in English?

Our Own Parallel Corpus

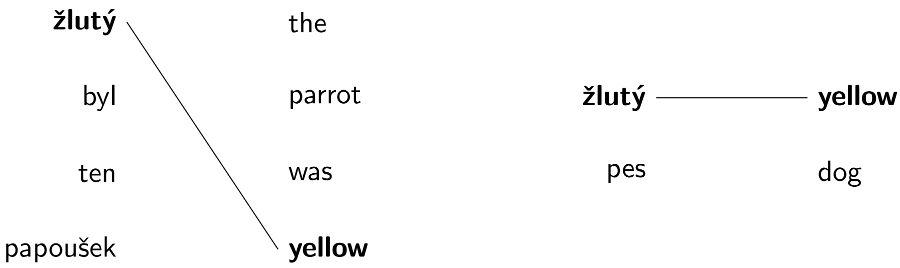
žlutý	the
byl	parrot
ten	was
papoušek	**yellow**

žlutý	**yellow**
pes	dog

What does "žlutý" mean in English?

We used the data to **infer an alignment** between the words.

Our Own Parallel Corpus

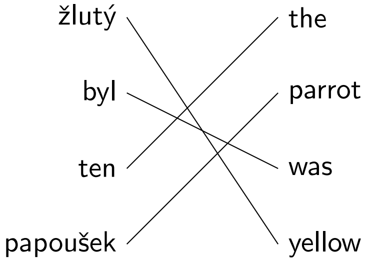


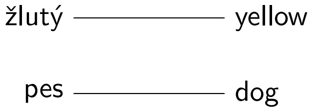
What does "žlutý" mean in English?

We used the data to **infer an alignment** between the words.
Given the alignment, we could find the most probable translation.

Estimating Translation Probability

If we had the alignment:

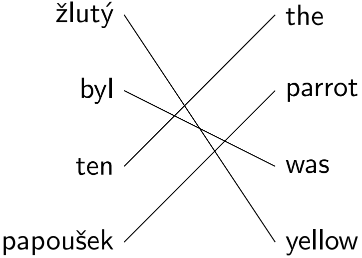


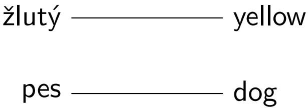


$$P(\text{"yellow"} \mid \text{"žlutý"}) = \frac{c(\text{yellow} \rightarrow \text{žlutý})}{c(\text{žlutý})} = \frac{2}{2} = 1$$

Estimating Translation Probability

If we had the alignment:



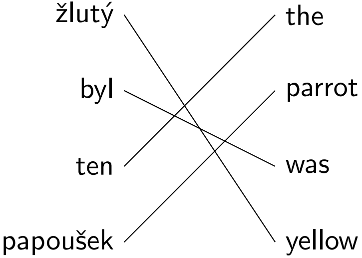


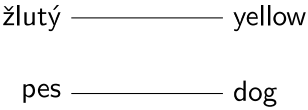
$$P(\text{"yellow"}|\text{"žlutý"}) = \frac{c(\text{yellow} \rightarrow \text{žlutý})}{c(\text{žlutý})} = \frac{2}{2} = 1$$

$$P(*|\text{"žlutý"}) = 0$$

Estimating Translation Probability

If we had the alignment:





$$P(\text{"yellow"}|\text{"žlutý"}) = \frac{c(\text{yellow} \rightarrow \text{žlutý})}{c(\text{žlutý})} = \frac{2}{2} = 1$$

$$P(*|\text{"žlutý"}) = 0$$

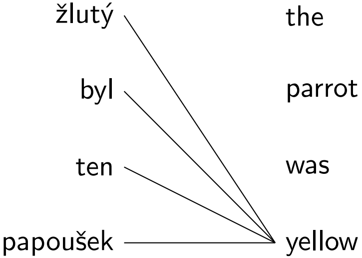
We will align the English words to Czech words.

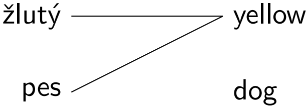
Estimation of IBM Model 1

žlutý	the		
byl	parrot	žlutý	yellow
ten	was	pes	dog
papoušek	yellow		

Our approach: distribute our one alignment link among all words.

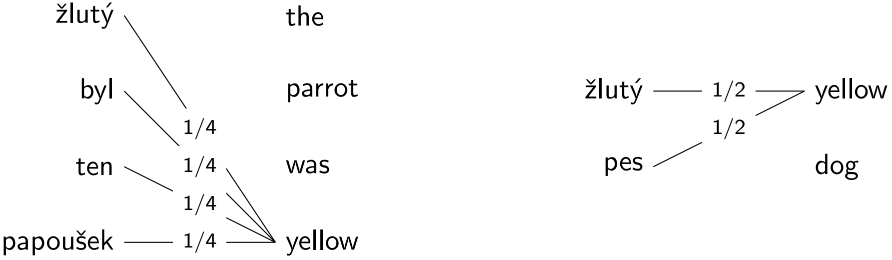
Estimation of IBM Model 1





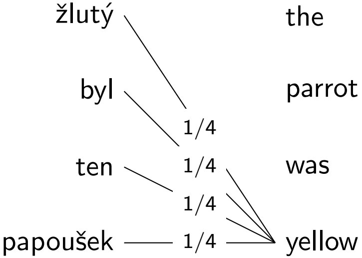
How to weight these "partial" links? Use translation probability $P(\mathbf{e}|\mathbf{f})$.

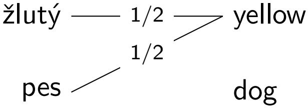
Estimation of IBM Model 1



Initially, we don't know anything ⇒ start with uniform estimates.

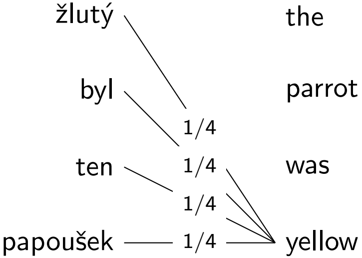
Estimation of IBM Model 1

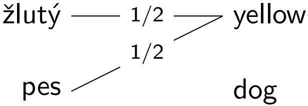




Let's sum up the evidence that "yellow" aligns to "žlutý":
$c(\text{yellow} \rightarrow \text{žlutý}) = 1/4 + 1/2 = 3/4$

Estimation of IBM Model 1





...and the evidence that "yellow" aligns to other words...

$c(\text{yellow} \rightarrow \text{žlutý}) = 3/4$

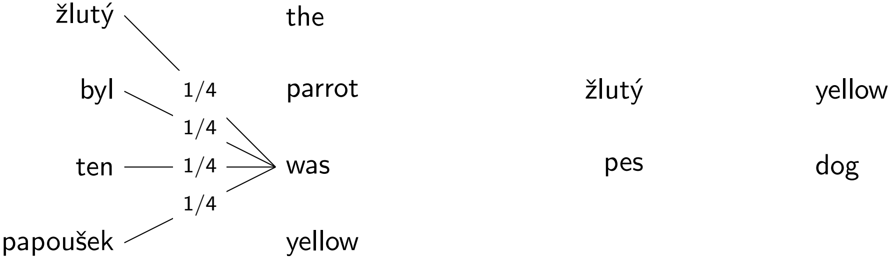
$c(\text{yellow} \rightarrow \text{byl}) = 1/4$

$c(\text{yellow} \rightarrow \text{ten}) = 1/4$

$c(\text{yellow} \rightarrow \text{papoušek}) = 1/4$

$c(\text{yellow} \rightarrow \text{pes}) = 1/2$

Estimation of IBM Model 1

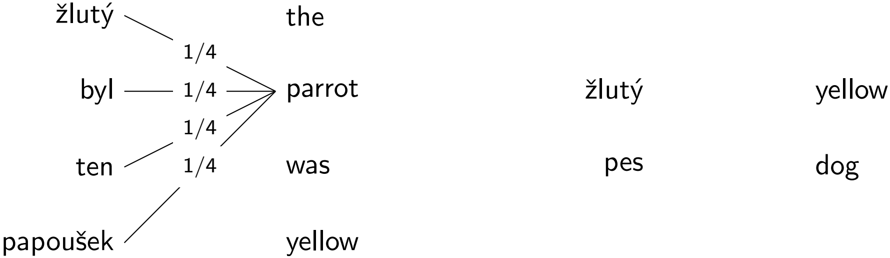


...and do the same for the other "partial" alignment links...

$c(\text{yellow} \rightarrow \text{žlutý}) = 3/4, c(\text{yellow} \rightarrow \text{byl}) = 1/4, \ldots$

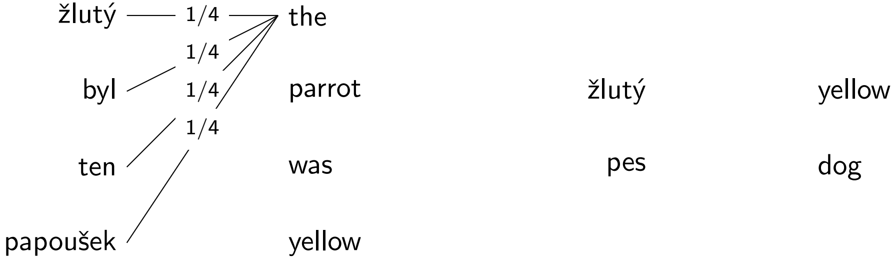
$c(\text{was} \rightarrow \text{žlutý}) = 1/4, c(\text{was} \rightarrow \text{byl}) = 1/4, \ldots$

Estimation of IBM Model 1



...and do the same for the other "partial" alignment links...
$c(\text{yellow} \rightarrow \text{žlutý}) = 3/4, c(\text{yellow} \rightarrow \text{byl}) = 1/4, \ldots$
$c(\text{was} \rightarrow \text{žlutý}) = 1/4, c(\text{was} \rightarrow \text{byl}) = 1/4, \ldots$
$c(\text{parrot} \rightarrow \text{žlutý}) = 1/4, c(\text{parrot} \rightarrow \text{byl}) = 1/4, \ldots$

Estimation of IBM Model 1



...and do the same for the other "partial" alignment links...

$c(\text{yellow} \rightarrow \text{žlutý}) = 3/4, c(\text{yellow} \rightarrow \text{byl}) = 1/4, \dots$

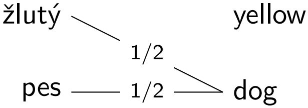
$c(\text{was} \rightarrow \text{žlutý}) = 1/4, c(\text{was} \rightarrow \text{byl}) = 1/4, \dots$

$c(\text{parrot} \rightarrow \text{žlutý}) = 1/4, c(\text{parrot} \rightarrow \text{byl}) = 1/4, \dots$

$c(\text{the} \rightarrow \text{žlutý}) = 1/4, c(\text{the} \rightarrow \text{byl}) = 1/4, \dots$

Estimation of IBM Model 1

žlutý	the
byl	parrot
ten	was
papoušek	yellow



...and do the same for the other "partial" alignment links...

$c(\text{yellow} \rightarrow \text{žlutý}) = 3/4, c(\text{yellow} \rightarrow \text{byl}) = 1/4, \dots$

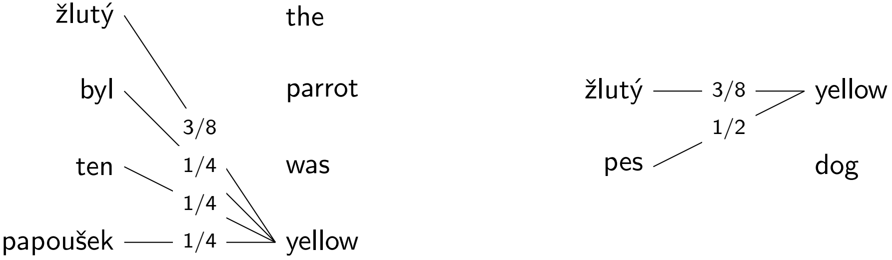
$c(\text{was} \rightarrow \text{žlutý}) = 1/4, c(\text{was} \rightarrow \text{byl}) = 1/4, \dots$

$c(\text{parrot} \rightarrow \text{žlutý}) = 1/4, c(\text{parrot} \rightarrow \text{byl}) = 1/4, \dots$

$c(\text{the} \rightarrow \text{žlutý}) = 1/4, c(\text{the} \rightarrow \text{byl}) = 1/4, \dots$

$c(\text{dog} \rightarrow \text{žlutý}) = 1/2, c(\text{dog} \rightarrow \text{pes}) = 1/2$

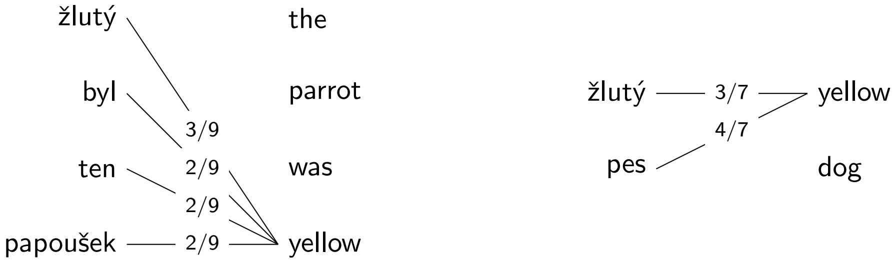
Estimation of IBM Model 1



Normalize to get the conditional probability distributions:

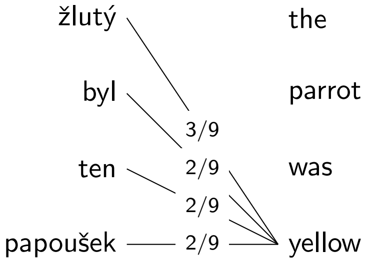
$$
\begin{array}{rrcr}
P(\text{yellow}|\text{žlutý}) = 3/8 & P(\text{yellow}|\text{byl}) = 1/4 & & P(\text{yellow}|\text{pes}) = 1/2 \\
P(\text{was}|\text{žlutý}) = 1/8 & P(\text{was}|\text{byl}) = 1/4 & & P(\text{was}|\text{pes}) = 0 \\
P(\text{parrot}|\text{žlutý}) = 1/8 & P(\text{parrot}|\text{byl}) = 1/4 & \cdots & P(\text{parrot}|\text{pes}) = 0 \\
P(\text{the}|\text{žlutý}) = 1/8 & P(\text{the}|\text{byl}) = 1/4 & & P(\text{the}|\text{pes}) = 0 \\
P(\text{dog}|\text{žlutý}) = 2/8 & P(\text{dog}|\text{byl}) = 0 & & P(\text{dog}|\text{pes}) = 1/2
\end{array}
$$

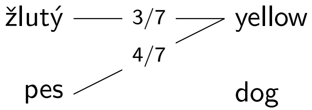
Estimation of IBM Model 1



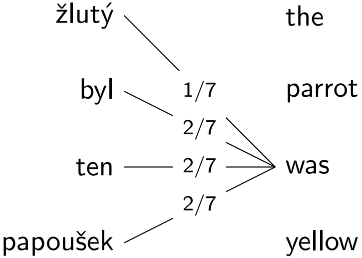
What next? Iterate.

Estimation of IBM Model 1



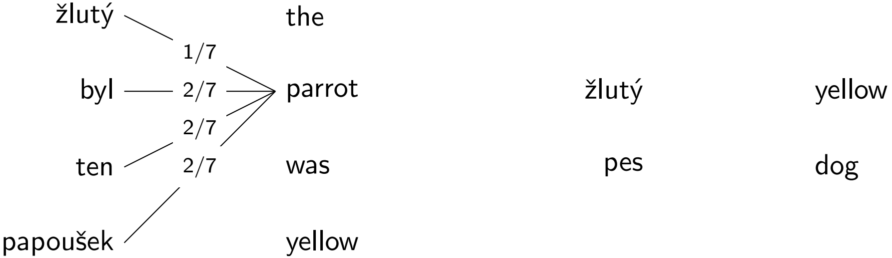


Estimation of IBM Model 1

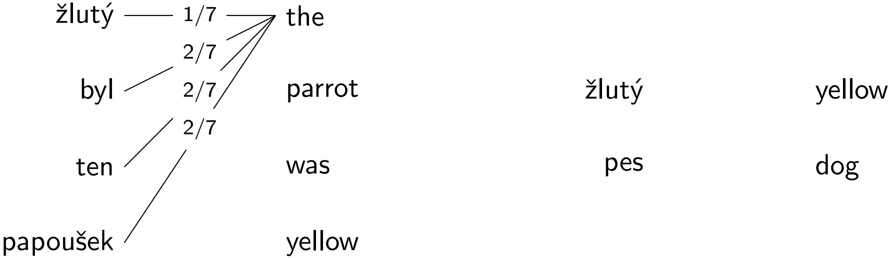


žlutý	yellow
pes	dog

Estimation of IBM Model 1

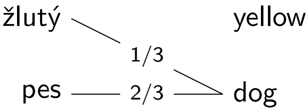


Estimation of IBM Model 1

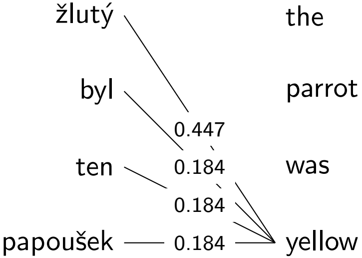


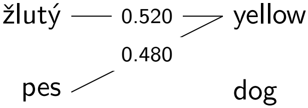
Estimation of IBM Model 1

žlutý	the
byl	parrot
ten	was
papoušek	yellow



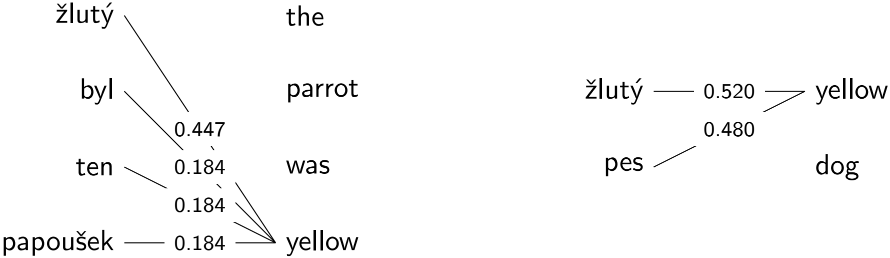
Estimation of IBM Model 1





$$\begin{aligned} P(\text{yellow}|\text{žlutý}) &= 0.5 \\ P(\text{was}|\text{žlutý}) &= 0.094 \\ P(\text{parrot}|\text{žlutý}) &= 0.094 \\ P(\text{the}|\text{žlutý}) &= 0.094 \\ P(\text{dog}|\text{žlutý}) &= 0.219 \end{aligned} \quad \begin{aligned} P(\text{yellow}|\text{byl}) &= 0.206 \\ P(\text{was}|\text{byl}) &= 0.265 \\ P(\text{parrot}|\text{byl}) &= 0.265 \\ P(\text{the}|\text{byl}) &= 0.265 \\ P(\text{dog}|\text{byl}) &= 0 \end{aligned} \quad \cdots \quad \begin{aligned} P(\text{yellow}|\text{pes}) &= 0.462 \\ P(\text{was}|\text{pes}) &= 0 \\ P(\text{parrot}|\text{pes}) &= 0 \\ P(\text{the}|\text{pes}) &= 0 \\ P(\text{dog}|\text{pes}) &= 0.538 \end{aligned}$$

Estimation of IBM Model 1



The algorithm: expectation maximization (EM)

1. Initialize the model with uniform probabilities.
2. Apply the model to the data (**expectation** step).
3. Re-estimate the model from the data (**maximization** step).
4. Go to 2 and repeat until probabilities stop changing.

Word-Based Models

- IBM Models 1-5 (increasing model complexity)
- Brown et al. (1993): The Mathematics of Statistical Machine Translation: Parameter Estimation
- Originally developed for word-based translation
- Higher models account for:
  - word position (IBM 1 only models the lexical translation probability)
  - fertility (number of English words aligned to a foreign word)
- Today: used for **word alignment**

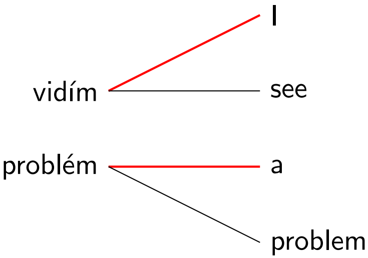
IBM Model 1

- We treat the alignment between words as a hidden variable.
- Alignment is a function; each English word (position) picks a foreign counterpart, e.g. $a(4) = 1$ ("yellow" aligns to "žlutý" in the first sentence).
- IBM Model 1 only models lexical translation probability, so formally, the probability of sentence $\mathbf{e} = (e_1, \ldots, e_m)$ given $\mathbf{f} = (f_1, \ldots, f_n)$ is:

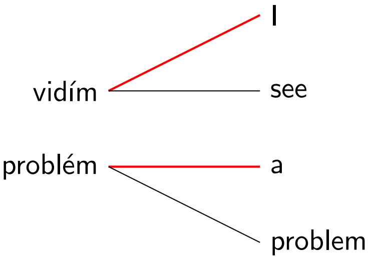
$$P(\mathbf{e}|\mathbf{f}) = \sum_{a_1=0}^{n} \cdots \sum_{a_m=0}^{n} \frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} t(e_j|f_{a_j}) = \frac{\epsilon}{(n+1)^m} \prod_{j=1}^{m} \sum_{i=0}^{n} t(e_j|f_i)$$

- EM finds such an alignment which maximizes the (log) likelihood of our data.

NULL Token

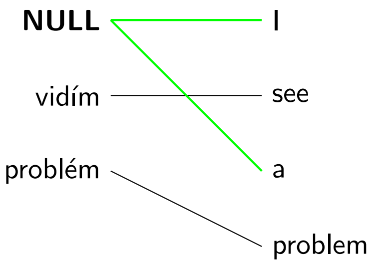


NULL Token



Do we align the indefinite article to all Czech nouns?

NULL Token



Align words which are dropped in Czech to NULL.

From IBM Models to Word Alignment

IBM models 1 to 5 are learned in a sequence; estimated parameters of one model are used to initialize the next model.

At the end of training, we can obtain the most likely alignment of the data.

From IBM Models to Word Alignment

IBM models 1 to 5 are learned in a sequence; estimated parameters of one model are used to initialize the next model.
At the end of training, we can obtain the most likely alignment of the data.
Alignment is a function $\Rightarrow$ one English word cannot align to more than one foreign word.

psací	typewriter	vysavač	vacuum
stroj			cleaner

From IBM Models to Word Alignment

IBM models 1 to 5 are learned in a sequence; estimated parameters of one model are used to initialize the next model.

At the end of training, we can obtain the most likely alignment of the data.

Alignment is a function $\Rightarrow$ one English word cannot align to more than one foreign word.

From IBM Models to Word Alignment

IBM models 1 to 5 are learned in a sequence; estimated parameters of one model are used to initialize the next model.

At the end of training, we can obtain the most likely alignment of the data.

Alignment is a function $\Rightarrow$ one English word cannot align to more than one foreign word.

There is no way that we can get this word alignment with our current models.

From IBM Models to Word Alignment

IBM models 1 to 5 are learned in a sequence; estimated parameters of one model are used to initialize the next model.
At the end of training, we can obtain the most likely alignment of the data.
Alignment is a function $\Rightarrow$ one English word cannot align to more than one foreign word.

There is no way that we can get this word alignment with our current models.
Solution: run the alignment **in both directions** (i.e., train all the models twice, English→Czech and Czech→English) and **symmetrize** the alignment.

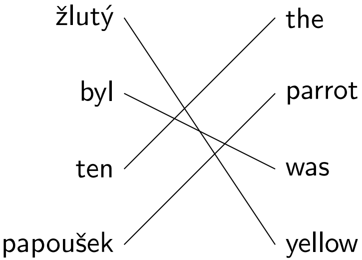
Alignment Symmetrization

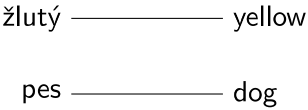
- A heuristic procedure, several possible strategies.
- Start with an intersection of the alignment links.
- Gradually add links from the union which are allowed by the chosen criteria.

Alignment Symmetrization

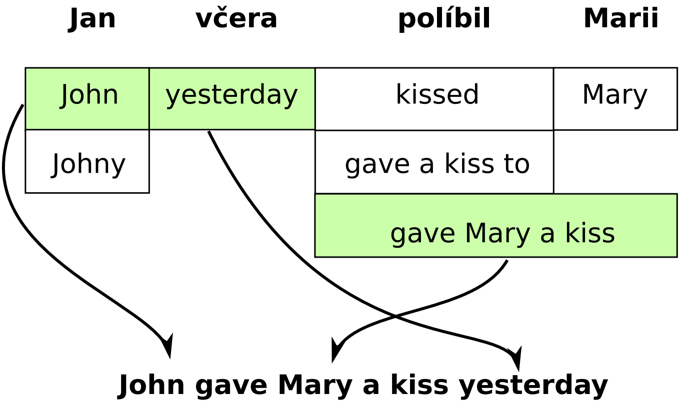
- A heuristic procedure, several possible strategies.
- Start with an intersection of the alignment links.
- Gradually add links from the union which are allowed by the chosen criteria.

Progress Check

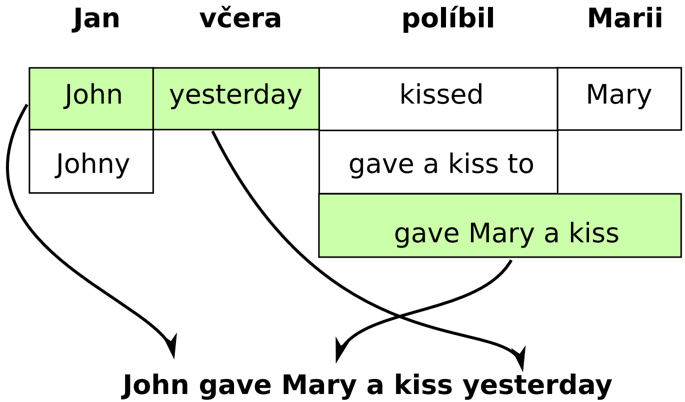




Progress Check

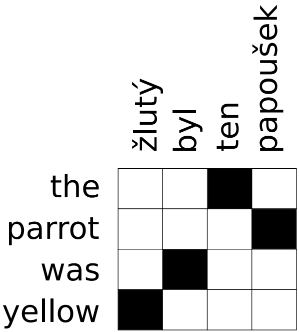


Progress Check

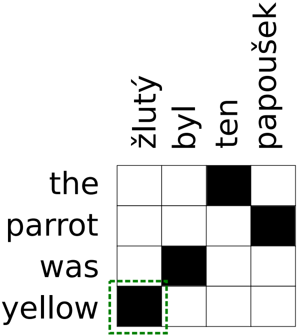


Let's go from words to phrases.

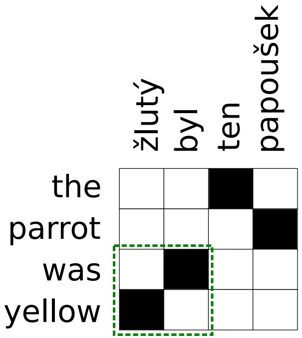
Phrase Extraction



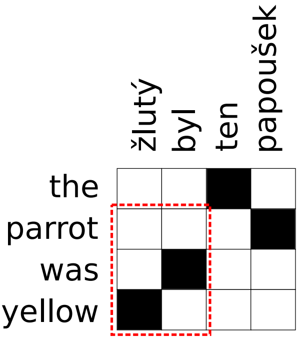
Phrase Extraction



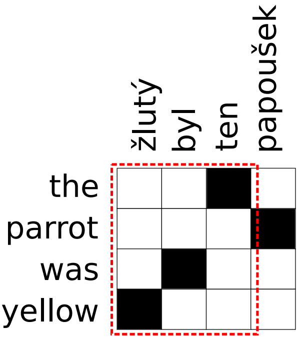
Phrase Extraction



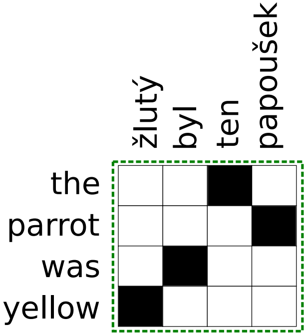
Phrase Extraction



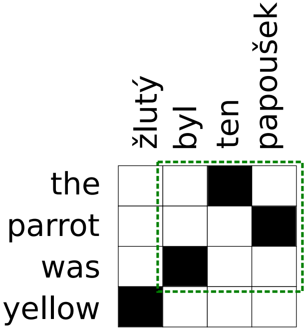
Phrase Extraction



Phrase Extraction



Phrase Extraction



Building a Phrase Table (Translation Model)

- Obtain some parallel data (sentence aligned).

Building a Phrase Table (Translation Model)

- Obtain some parallel data (sentence aligned).
- Run word alignment (IBM models) in both directions, symmetrize.

Building a Phrase Table (Translation Model)

- Obtain some parallel data (sentence aligned).
- Run word alignment (IBM models) in both directions, symmetrize.
- Extract admissible phrase pairs up to a certain length (typically around 7 words).

Building a Phrase Table (Translation Model)

- Obtain some parallel data (sentence aligned).
- Run word alignment (IBM models) in both directions, symmetrize.
- Extract admissible phrase pairs up to a certain length (typically around 7 words).
- Count phrase (co-)occurrences to estimate phrase translation probabilities:

$$P(\mathbf{e}|\mathbf{f}) = \frac{c(\mathbf{e}\&\mathbf{f})}{c(\mathbf{f})}$$

Building a Phrase Table (Translation Model)

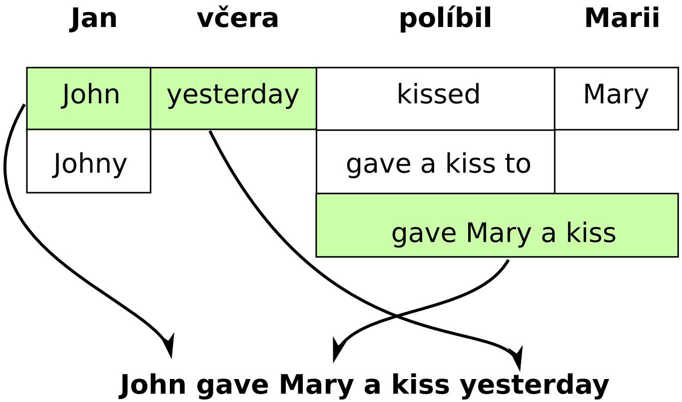
- Obtain some parallel data (sentence aligned).
- Run word alignment (IBM models) in both directions, symmetrize.
- Extract admissible phrase pairs up to a certain length (typically around 7 words).
- Count phrase (co-)occurrences to estimate phrase translation probabilities:

$$P(\mathbf{e}|\mathbf{f}) = \frac{c(\mathbf{e}\&\mathbf{f})}{c(\mathbf{f})}$$

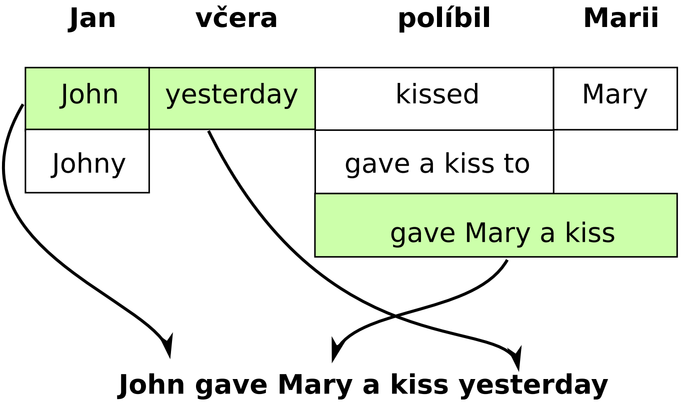
Tiny example:

```
žlutý papoušek ||| a yellow parrot ||| 0.1
žlutý papoušek ||| yellow parakeet ||| 0.1
žlutý papoušek ||| yellow parrot ||| 0.6
žlutý papoušek ||| yellowish parrot ||| 0.2
```

Progress Check



Progress Check



How do we decide which of these translations is best?

The Noisy Channel Model

Warren Weaver (1955):

> *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.'*

The Noisy Channel Model

Warren Weaver (1955):

> *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.'*

The Noisy Channel Model

Warren Weaver (1955):

> *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.'*

We are looking for the most probable original English sentence (which we received in Russian due to "noise").

The Noisy Channel Model

Warren Weaver (1955):

> *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.'*

We are looking for the most probable original English sentence (which we received in Russian due to "noise").

$$\begin{aligned}\hat{\mathbf{e}} &= \arg\max_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} \frac{P(\mathbf{f}|\mathbf{e})P(\mathbf{e})}{P(\mathbf{f})} \\ &= \arg\max_{\mathbf{e}} \underbrace{P(\mathbf{f}|\mathbf{e})}_{\textit{Translation model}} \underbrace{P(\mathbf{e})}_{\textit{Language model}}\end{aligned}$$

Noisy Channel Model

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} P(\mathbf{f}|\mathbf{e})P(\mathbf{e})$$

- $P(\mathbf{e})$ is the language model (LM).
- $P(\mathbf{f}|\mathbf{e})$ depends on the application:
  - Automatic speech recognition: the acoustic model.
  - Spelling correction: the spelling error model.
  - Machine translation: the translation model (TM).

Noisy Channel Model

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} P(\mathbf{f}|\mathbf{e})P(\mathbf{e})$$

- $P(\mathbf{e})$ is the language model (LM).
- $P(\mathbf{f}|\mathbf{e})$ depends on the application:
  - Automatic speech recognition: the acoustic model.
  - Spelling correction: the spelling error model.
  - Machine translation: the translation model (TM).
- A useful decomposition:
  - TM: How accurately does the translation match the input? (Parallel data needed for training.)
  - LM: Is the translation is good (fluent) English? (Only requires monolingual data!)

Noisy Channel Model

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} P(\mathbf{f}|\mathbf{e})P(\mathbf{e})$$

- $P(\mathbf{e})$ is the language model (LM).
- $P(\mathbf{f}|\mathbf{e})$ depends on the application:
  - Automatic speech recognition: the acoustic model.
  - Spelling correction: the spelling error model.
  - Machine translation: the translation model (TM).
- A useful decomposition:
  - TM: How accurately does the translation match the input? (Parallel data needed for training.)
  - LM: Is the translation is good (fluent) English? (Only requires monolingual data!)
- So far, we only talked about half of the story. (And technically, in the wrong direction, given that we want to translate Czech into English.)

Language Model

- Don't miss the lecture on language modelling tomorrow (Kenneth Heafield).

Language Model

- Don't miss the lecture on language modelling tomorrow (Kenneth Heafield).
- The task: decide which sequences of words are good English.
  For any English sentence $\mathbf{e} = (e_1, \ldots, e_{l_e})$, estimate $P(\mathbf{e})$

Language Model

- Don't miss the lecture on language modelling tomorrow (Kenneth Heafield).
- The task: decide which sequences of words are good English.
  For any English sentence $\mathbf{e} = (e_1, \ldots, e_{l_e})$, estimate $P(\mathbf{e})$
- We need to decompose the joint probability somehow. The (usual) solution: *n*-gram language models.

Language Model

- Don't miss the lecture on language modelling tomorrow (Kenneth Heafield).
- The task: decide which sequences of words are good English.
  For any English sentence $\mathbf{e} = (e_1, \ldots, e_{l_e})$, estimate $P(\mathbf{e})$
- We need to decompose the joint probability somehow. The (usual) solution: *n*-gram language models.
- Side note – chain rule (example for 4 variables):

$$P(A, B, C, D) = P(D|A, B, C) \cdot P(C|A, B) \cdot P(B|A) \cdot P(A)$$

Language Model

- Don't miss the lecture on language modelling tomorrow (Kenneth Heafield).
- The task: decide which sequences of words are good English.
  For any English sentence $\mathbf{e} = (e_1, \ldots, e_{l_e})$, estimate $P(\mathbf{e})$
- We need to decompose the joint probability somehow. The (usual) solution: n-gram language models.
- Side note – chain rule (example for 4 variables):

$$P(A, B, C, D) = P(D|A, B, C) \cdot P(C|A, B) \cdot P(B|A) \cdot P(A)$$

- Let's formulate the n-gram LM:

$$\begin{aligned} P(\mathbf{e}) &= P(e_1)P(e_2|e_1)P(e_3|e_1, e_2) \ldots P(e_{l_e}|e_1, \ldots, e_{l_e-1}) \\ &\approx P(e_1)P(e_2|e_1) \ldots P(e_{l_e}|e_{l_e-n+1}, \ldots, e_{l_e-1}) \end{aligned}$$

Language Model: Example

A 3-gram language model (only depend on 2 previous words).

$$
\begin{aligned}
P(\text{"thank you very much"}) = {} & P(\text{"thank"} \mid \text{"<s>"}) \\
& \times P(\text{"you"} \mid \text{"<s>thank"}) \\
& \times P(\text{"very"} \mid \text{"thank you"}) \\
& \times P(\text{"much"} \mid \text{"you very"})
\end{aligned}
$$

Language Model: Example

A 3-gram language model (only depend on 2 previous words).

$$
\begin{aligned}
P(\text{"thank you very much"}) = {} & P(\text{"thank"} \mid \text{"<s>"}) \\
& \times P(\text{"you"} \mid \text{"<s>thank"}) \\
& \times P(\text{"very"} \mid \text{"thank you"}) \\
& \times P(\text{"much"} \mid \text{"you very"})
\end{aligned}
$$

To estimate e.g. $P(\text{"very"} \mid \text{"thank you"})$, we go through the data and count:

- How many times we saw "thank you" followed by "very".
- How many times we saw "thank you" (followed by anything).

Language Model: Example

A 3-gram language model (only depend on 2 previous words).

$$
\begin{aligned}
P(\text{"thank you very much"}) = {} & P(\text{"thank"} | \text{"<s>"}) \\
& \times P(\text{"you"} | \text{"<s>thank"}) \\
& \times P(\text{"very"} | \text{"thank you"}) \\
& \times P(\text{"much"} | \text{"you very"})
\end{aligned}
$$

To estimate e.g. $P(\text{"very"}|\text{"thank you"})$, we go through the data and count:

- How many times we saw "thank you" followed by "very".
- How many times we saw "thank you" (followed by anything).

$$P(\text{"very"}|\text{"thank you"}) = \frac{c(\text{"thank you very"})}{c(\text{"thank you"})}$$

Language Model: Example

A 3-gram language model (only depend on 2 previous words).

$$\begin{aligned} P(\text{"thank you very much"}) = {} & P(\text{"thank"} | \text{"<s>"}) \\ & \times P(\text{"you"} | \text{"<s>thank"}) \\ & \times P(\text{"very"} | \text{"thank you"}) \\ & \times P(\text{"much"} | \text{"you very"}) \end{aligned}$$

To estimate e.g. $P(\text{"very"} | \text{"thank you"})$, we go through the data and count:

- How many times we saw "thank you" followed by "very".
- How many times we saw "thank you" (followed by anything).

$$P(\text{"very"} | \text{"thank you"}) = \frac{c(\text{"thank you very"})}{c(\text{"thank you"})}$$

Smoothing!

Log-Linear Model

Begin with the noisy channel model:

$$\begin{aligned}\hat{e} &= \arg\max_e P(\mathbf{f}|\mathbf{e})P(\mathbf{e}) \\ &= \arg\max_e \log(P(\mathbf{f}|\mathbf{e})P(\mathbf{e})) \\ &= \arg\max_e \log(P(\mathbf{f}|\mathbf{e})) + \log(P(\mathbf{e}))\end{aligned}$$

Log-Linear Model

Begin with the noisy channel model:

$$
\begin{aligned}
\hat{e} &= \arg\max_{e} P(\mathbf{f}|\mathbf{e})P(\mathbf{e}) \\
&= \arg\max_{e} \log(P(\mathbf{f}|\mathbf{e})P(\mathbf{e})) \\
&= \arg\max_{e} \log(P(\mathbf{f}|\mathbf{e})) + \log(P(\mathbf{e}))
\end{aligned}
$$

Perhaps the importance of LM vs. TM should be weighted differently?

$$
\hat{e} = \arg\max_{e} \lambda_{TM} \log(P(\mathbf{f}|\mathbf{e})) + \lambda_{LM} \log(P(\mathbf{e}))
$$

Log-Linear Model

Begin with the noisy channel model:

$$\begin{aligned}\hat{e} &= \arg\max_e P(\mathbf{f}|\mathbf{e})P(\mathbf{e}) \\ &= \arg\max_e \log(P(\mathbf{f}|\mathbf{e})P(\mathbf{e})) \\ &= \arg\max_e \log(P(\mathbf{f}|\mathbf{e})) + \log(P(\mathbf{e}))\end{aligned}$$

Perhaps the importance of LM vs. TM should be weighted differently?

$$\hat{e} = \arg\max_e \lambda_{TM} \log(P(\mathbf{f}|\mathbf{e})) + \lambda_{LM} \log(P(\mathbf{e}))$$

We could add other features (besides LM and TM), so generally:

$$\hat{e} = \arg\max_e \sum_i \lambda_i f_i(\mathbf{e}, \mathbf{f})$$

Log-Linear Model: Features

We now have the freedom to add new features. In PBMT, we typically use:

- Phrase translation probability, both direct and inverse:
  - $P(\mathbf{e}|\mathbf{f})$
  - $P(\mathbf{f}|\mathbf{e})$
- Lexical translation probability (direct and inverse):
  - $P_{lex}(\mathbf{e}|\mathbf{f})$
  - $P_{lex}(\mathbf{f}|\mathbf{e})$
- Language model probability:
  - $P(\mathbf{e})$
- Phrase penalty.
- Word penalty.
- Distortion penalty.

Lexical Weights (P_{lex})

The problem: many extracted phrases are rare.
(Esp. long phrases might only be seen once in the parallel corpus.)

Lexical Weights (P_{lex})

The problem: many extracted phrases are rare.
(Esp. long phrases might only be seen once in the parallel corpus.)

$$P(\text{"modrý autobus přistál na Marsu"} \mid \text{"a blue bus lands on Mars"}) = 1$$
$$P(\text{"a blue bus lands on Mars"} \mid \text{"modrý autobus přistál na Marsu"}) = 1$$

Is that a reliable probability estimate?

Lexical Weights (P_{lex})

The problem: many extracted phrases are rare.
(Esp. long phrases might only be seen once in the parallel corpus.)

$$P(\text{"; distortion carried - over"} | \text{"; zkreslení"}) = 1$$
$$P(\text{"; zkreslení"} | \text{"; distortion carried - over"}) = 1$$

Data from the "wild" are noisy. Word alignment contains errors.
This is a real phrase pair from our best English-Czech system.
Both $P(\mathbf{e}|\mathbf{f})$ and $P(\mathbf{f}|\mathbf{e})$ say that this is a perfect translation.

Lexical Weights (P_{lex})

Decompose the phrase pair into word pairs. Look at the word-level translation probabilities.

Lexical Weights (P_{lex})

Decompose the phrase pair into word pairs. Look at the word-level translation probabilities.

Several possible definitions, e.g.:

Lexical Weights (P_{lex})

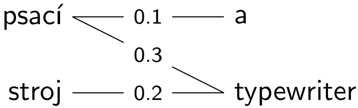
Decompose the phrase pair into word pairs. Look at the word-level translation probabilities.
Several possible definitions, e.g.:

$$P_{lex}(\mathbf{e}|\mathbf{f}, a) = \prod_{j=1}^{l_e} \frac{1}{|i|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(e_j, f_i)$$

Lexical Weights (P_{lex})

Decompose the phrase pair into word pairs. Look at the word-level translation probabilities.
Several possible definitions, e.g.:

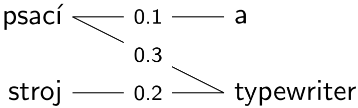
$$P_{lex}(\mathbf{e}|\mathbf{f}, a) = \prod_{j=1}^{l_e} \frac{1}{|i|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(e_j, f_i)$$



Lexical Weights (P_{lex})

Decompose the phrase pair into word pairs. Look at the word-level translation probabilities.
Several possible definitions, e.g.:

$$P_{lex}(\mathbf{e}|\mathbf{f}, a) = \prod_{j=1}^{l_e} \frac{1}{|i|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(e_j, f_i)$$



$$P_{lex}(\text{"a typewriter"}|\text{"psací stroj"}) = \left[\frac{1}{1} \cdot 0.1\right] \cdot \left[\frac{1}{2} \cdot (0.3 + 0.2)\right] = 0.025$$

Word Penalty

Not all languages use the same number of words on average.

```
vidím problém ||| I can see a problem
```

Word Penalty

Not all languages use the same number of words on average.

```
vidím problém ||| I can see a problem
```

- We want to control how many words are generated.

Word Penalty

Not all languages use the same number of words on average.

```
vidím problém ||| I can see a problem
```

- We want to control how many words are generated.
- Word penalty simply adds 1 for each produced word in the translation.

Word Penalty

Not all languages use the same number of words on average.

```
vidím problém ||| I can see a problem
```

- We want to control how many words are generated.
- Word penalty simply adds 1 for each produced word in the translation.
- Depending on the λ for word penalty, we will either generate shorter or longer outputs.

Word Penalty

Not all languages use the same number of words on average.

```
vidím problém ||| I can see a problem
```

- We want to control how many words are generated.
- Word penalty simply adds 1 for each produced word in the translation.
- Depending on the λ for word penalty, we will either generate shorter or longer outputs.

$$\hat{e} = \arg\max_{e} \sum_{i} \lambda_i f_i(\mathbf{e}, \mathbf{f})$$

Phrase Penalty

- Add 1 for each produced *phrase* in the translation.

Phrase Penalty

- Add 1 for each produced *phrase* in the translation.
- Varying the λ for phrase penalty can lead to more literal (word-by-word) translations (made from a lot of short phrases) or to more idiomatic outputs (use fewer, longer phrases – if available).

Distortion Penalty

- The simplest way to capture **phrase reordering**.
- Can be sufficient for some language pairs (our English→Czech systems use it).
- Several possible definitions, e.g.:
  - Distance between the end of the previous phrase (on the source side) and the beginning of the current phrase.

Model Weights

- How to get λs for our feature functions?

Model Weights

- How to get λs for our feature functions?
- Usual approach: set them so that we translate some held-out data as well as possible.

Model Weights

- How to get λs for our feature functions?
- Usual approach: set them so that we translate some held-out data as well as possible.

"Tuning"

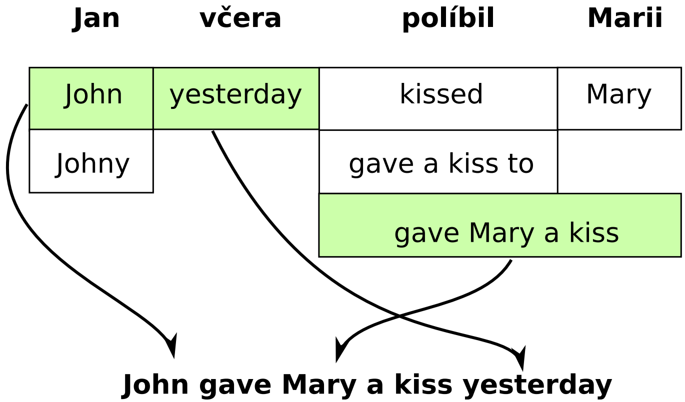
Model Weights

- How to get λs for our feature functions?
- Usual approach: set them so that we translate some held-out data as well as possible.

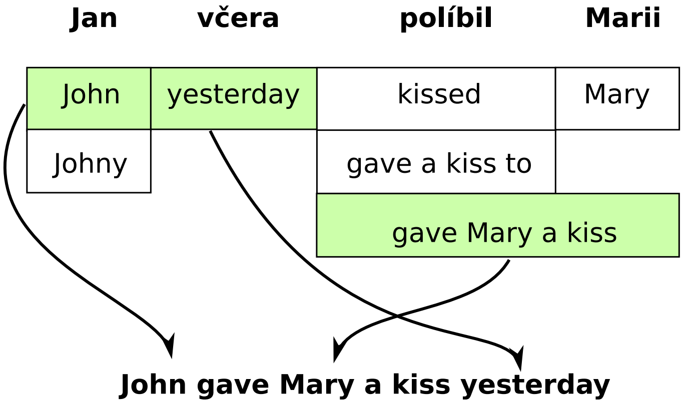
"Tuning"

- See the lecture tomorrow: Discriminative Training (Miloš Stanojević)

Progress Check



Progress Check



Search for the best translation.

Translation Process: Generate Translation Options

Jan	**včera**	**políbil**	**Marii**
John	yesterday	kissed	Mary
Johny		gave a kiss to	
		gave Mary a kiss	

Translation Process: Beam Search