Natural Language Processing Applications

Lecture 09: Machine Translation

fit@hcmus
KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

- Introduction to Machine Translation
- Word-based models
- Phrase-based models
- Evaluation

NLPA- Machine Translation

INTRODUCTION TO MACHINE TRANSLATION

INTRODUCTION TO MACHINE TRANSLATION

fit@hcmus

NLPA- Machine Translation

Word-based models

Word-based models

- **Lexical Translation:**
  - How to translate a word -> look up in dictionary
    - Example for German-English dictionary:

      Haus - house, building, home, household, shell

  - Multiple translations:
    - Some more frequent than others
    - For instance: house, and building most common
    - Special cases: Haus of a snail is its shell
  - Note: In all lectures, we translate from a foreign language into English

Word-based models

- **Collect Statistics**
  - Look at a parallel corpus (German text along with English translation)

Translation of *Haus*	Count
house	8,000
building	1,600
home	200
household	150
shell	50

Word-based models

- Estimate Translation Probabilities

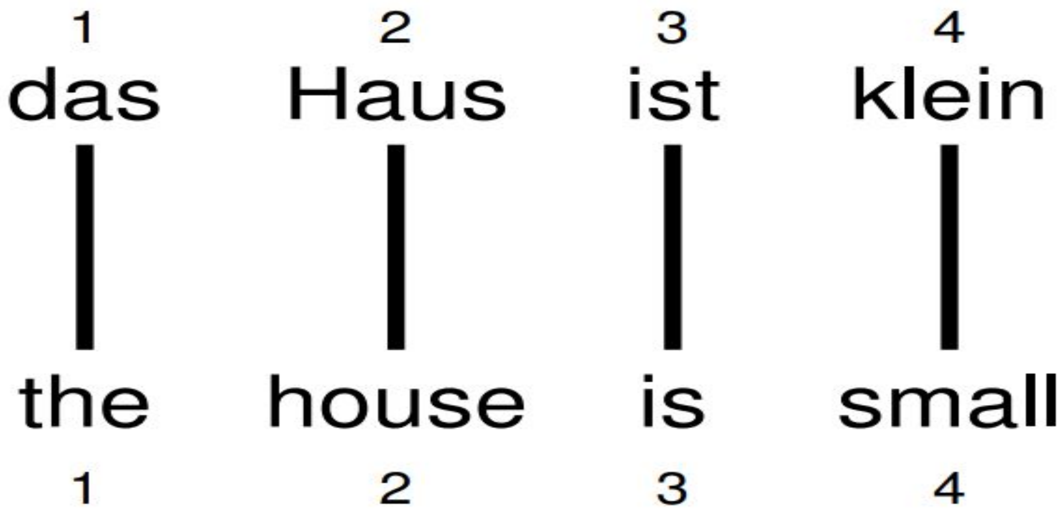
Maximum likelihood estimation

$$p_f(e) = \begin{cases} 0.8 & \text{if } e = \text{house}, \\ 0.16 & \text{if } e = \text{building}, \\ 0.02 & \text{if } e = \text{home}, \\ 0.015 & \text{if } e = \text{household}, \\ 0.005 & \text{if } e = \text{shell}. \end{cases}$$

fit@hcmus

Word-based models

- **Alignment**
  - In a parallel text (or when we translate), we align words in one language with the words in the other



- Word positions are numbered 1-4

Word-based models

- **Alignment Function**
  - Formalizing alignment with an alignment function
  - Mapping an English target word at position **i** to a German source word at position **j** with a function:

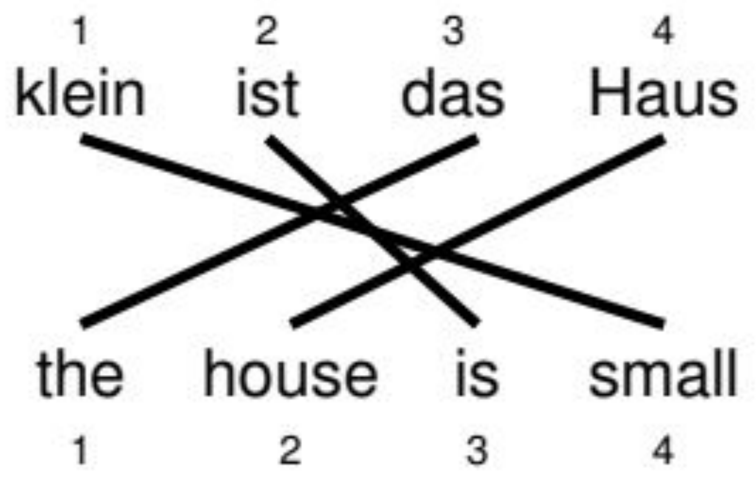
a: i -> j

  - Example:

a: {1 -> 1, 2 -> 2, 3 -> 3, 4 -> 4}

Word-based models

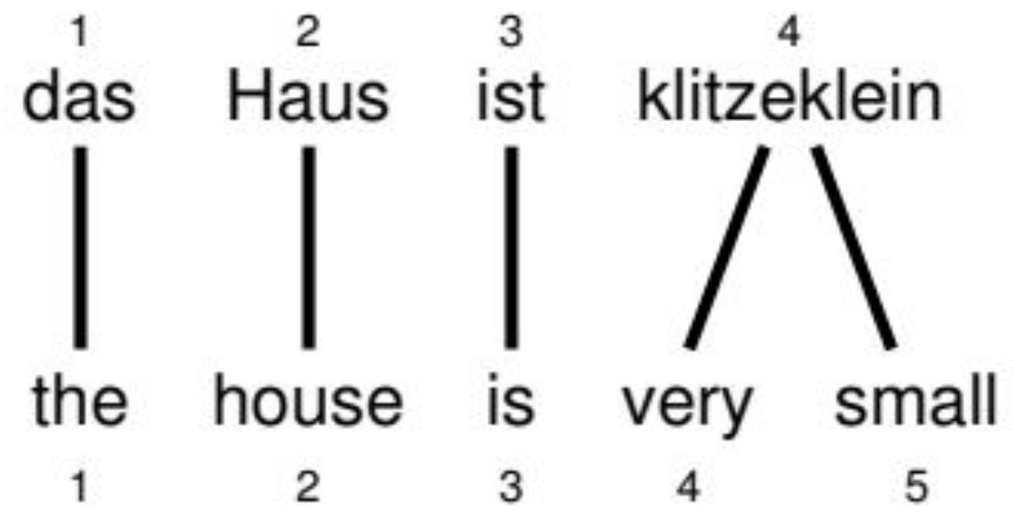
- **Reordering**
  - Words may be reordered during translation



- a: { 1 -> 3, 2 -> 4, 3 -> 2, 4 -> 1}

Word-based models

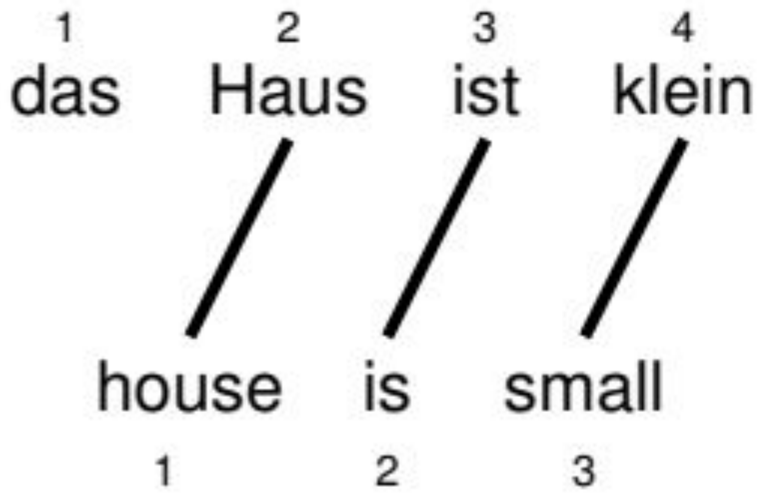
- **One-to-many Translation**
  - A source word may translate into multiple target words:



a: {1 -> 1, 2 -> 2, 3 -> 3, 4 -> 4, 5 -> 4}

Word-based models

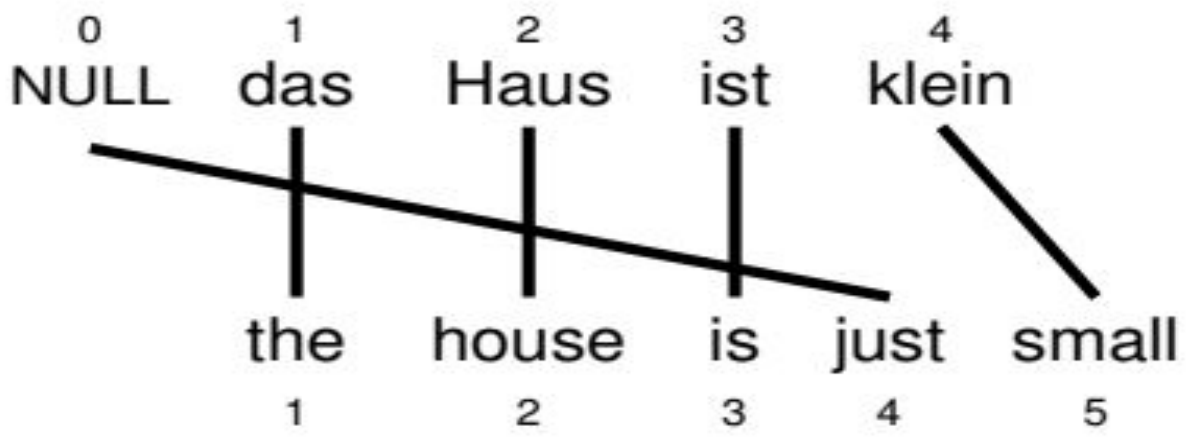
- **Dropping Words**
  - Words may be dropped when translated (German article **das** is dropped)



a: {1 -> 2, 2 -> 3, 3 -> 4}

Word-based models

- **Inserting Words**
  - Words may be added during translation
    - The English **just** does not have an equivalent in German
    - We still need to map it to something: special **NULL** token



a: {1 -> 1, 2 -> 2, 3 -> 3, 4 -> 0, 5-> 4}

Word-based models

- **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

Word-based models

Example

das

e	$t(e\|f)$
the	0.7
that	0.15
which	0.075
who	0.05
this	0.025

Haus

e	$t(e\|f)$
house	0.8
building	0.16
home	0.02
household	0.015
shell	0.005

ist

e	$t(e\|f)$
is	0.8
's	0.16
exists	0.02
has	0.015
are	0.005

klein

e	$t(e\|f)$
small	0.4
little	0.4
short	0.1
minor	0.06
petty	0.04

$$\begin{aligned} p(e,a|f) &= \frac{\epsilon}{5^4} \times t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein}) \\ &= \frac{\epsilon}{5^4} \times 0.7 \times 0.8 \times 0.8 \times 0.4 \\ &= 0.0029\epsilon \end{aligned}$$

Word-based models

- **Learning Lexical Translation Models**
  - We would like to estimate the lexical translation probabilities $t(e|f)$ from a parallel corpus
  - … but we do not have the alignments
  - Chicken and egg problem
    - if we had the *alignments*,

      -> we could estimate the parameters of our generative model
    - if we had the *parameters*,

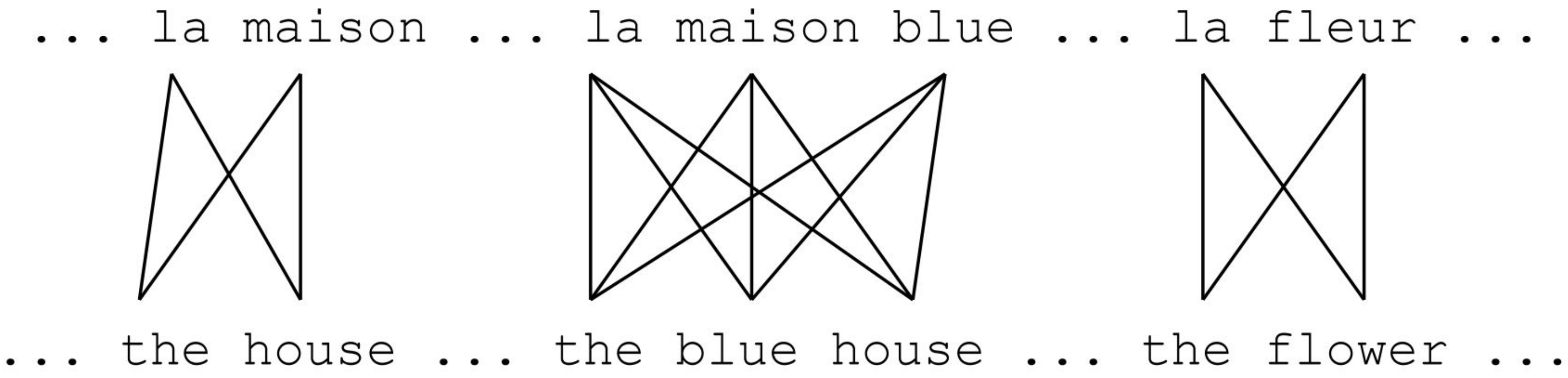
      -> we could estimate the alignments

Word-based models

- **EM Algorithm (Expectation Maximization Algorithm):**
  - Incomplete data
    - if we had complete data, would could estimate model
    - if we had model, we could fill in the gaps in the data
  - Expectation Maximization (EM) in a nutshell
    1. initialize model parameters (e.g. uniform)
    2. assign probabilities to the missing data.
    3. estimate model parameters from completed data
    4. iterate steps 2 - 3 until convergence

Word-based models

EM Algorithm

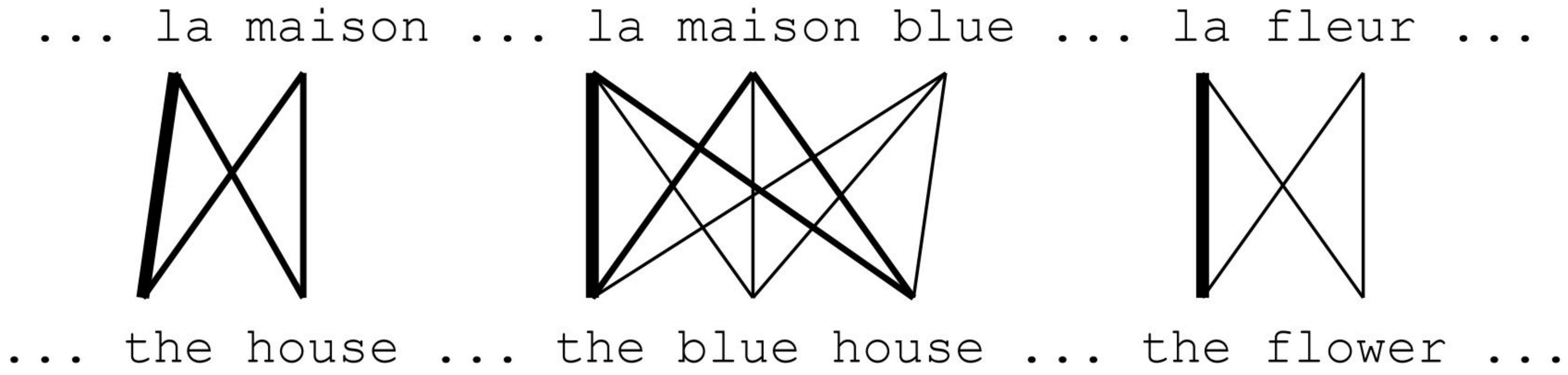


- Initial step: all alignments equally likely
- Model learns that, e.g., **la** is often aligned with **the**

fit@hcmus

Word-based models

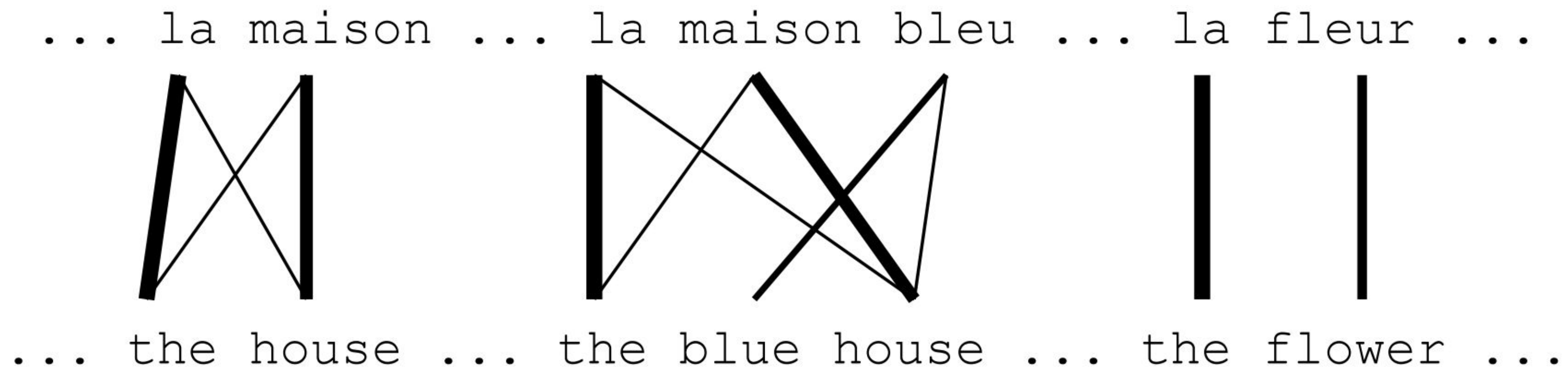
EM Algorithm



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

Word-based models

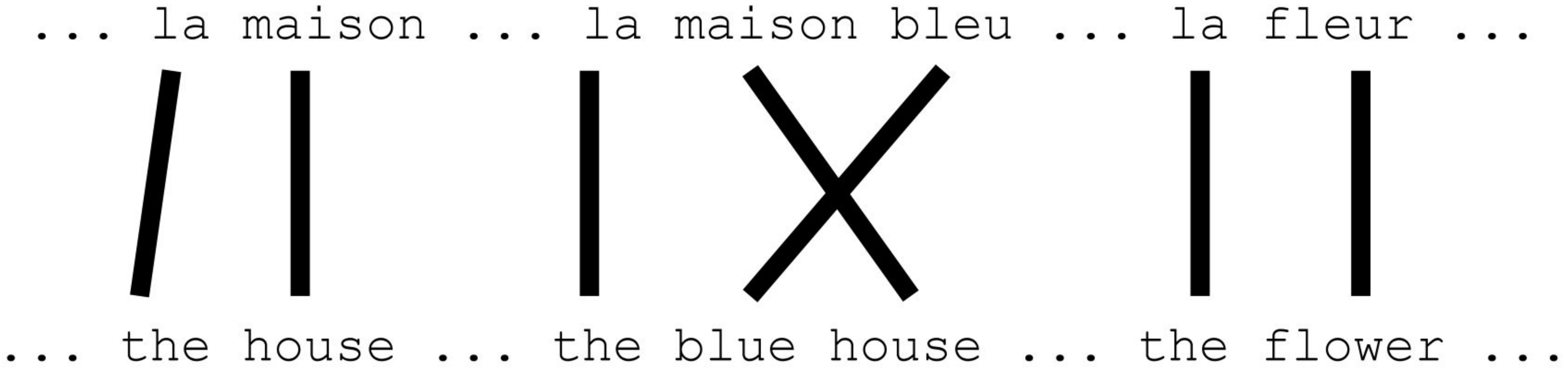
EM Algorithm



- After another iteration
- It becomes apparent that alignments, e.g., between fleur and flower are more likely (pigeon hole principle)

Word-based models

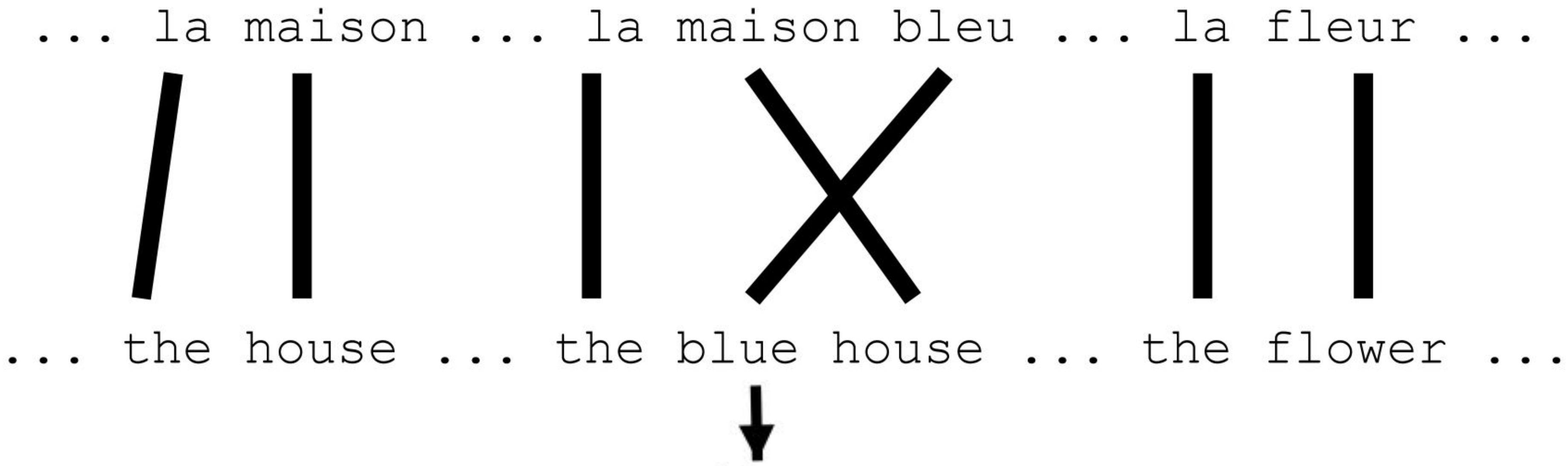
EM Algorithm



- Convergence
- Inherent hidden structure revealed by EM

Word-based models

EM Algorithm



p (la|the) = 0.453
p (le|the) = 0.334
p (maison|house) = 0.876
p (bleu|blue) = 0.563

Parameter estimation from the aligned corpus

Word-based models

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

Word-based models

IBM Model 1 and EM

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

fit@hcmus

Word-based models

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:

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

Word-based models

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 $p(\mathbf{e},a|\mathbf{f})$ (definition of Model 1)

Word-based models

IBM Model 1 and EM: Expectation Step

- We need to compute $p(\mathbf{e}|\mathbf{f})$

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

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} p(\mathbf{e}, a|\mathbf{f})$$

$$= \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

Word-based models

IBM Model 1 và EM: Expectation Step

$$p(\mathbf{e}|\mathbf{f}) = \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} \cdots \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

$$= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)$$

Note the trick in the last line

- Removes the need for an exponential number of products

-> This makes IBM Model 1 estimation tractable

Word-based models

The Trick

(case $l_e = l_f = 2$)

$$\sum_{a(1)=0}^{2} \sum_{a(2)=0}^{2} = \frac{\epsilon}{3^2} \prod_{j=1}^{2} t(e_j|f_{a(j)}) =$$

$$\begin{aligned}
&= t(e_1|f_0)\ t(e_2|f_0) + t(e_1|f_0)\ t(e_2|f_1) + t(e_1|f_0)\ t(e_2|f_2) + \\
&\quad + t(e_1|f_1)\ t(e_2|f_0) + t(e_1|f_1)\ t(e_2|f_1) + t(e_1|f_1)\ t(e_2|f_2) + \\
&\quad + t(e_1|f_2)\ t(e_2|f_0) + t(e_1|f_2)\ t(e_2|f_1) + t(e_1|f_2)\ t(e_2|f_2) = \\
&= t(e_1|f_0)\ (t(e_2|f_0) + t(e_2|f_1) + t(e_2|f_2)) + \\
&\quad + t(e_1|f_1)\ (t(e_2|f_1) + t(e_2|f_1) + t(e_2|f_2)) + \\
&\quad + t(e_1|f_2)\ (t(e_2|f_2) + t(e_2|f_1) + t(e_2|f_2)) = \\
&= \ (t(e_1|f_0) + t(e_1|f_1) + t(e_1|f_2))\ (t(e_2|f_2) + t(e_2|f_1) + t(e_2|f_2))
\end{aligned}$$

fit@hcmus

Word-based models

IBM Model 1 and EM: Expectation Step

- Combine what we have:

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

$$= \frac{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})}{\frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)}$$

$$= \prod_{j=1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

Word-based models

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:

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

Word-based models

IBM Model 1 and EM: Maximization Step

- After collecting these counts over a corpus, we can estimate the model:

$$t(e|f;\mathbf{e},\mathbf{f}) = \frac{\sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}{\sum_e \sum_{(\mathbf{e},\mathbf{f})} c(e|f;\mathbf{e},\mathbf{f}))}$$

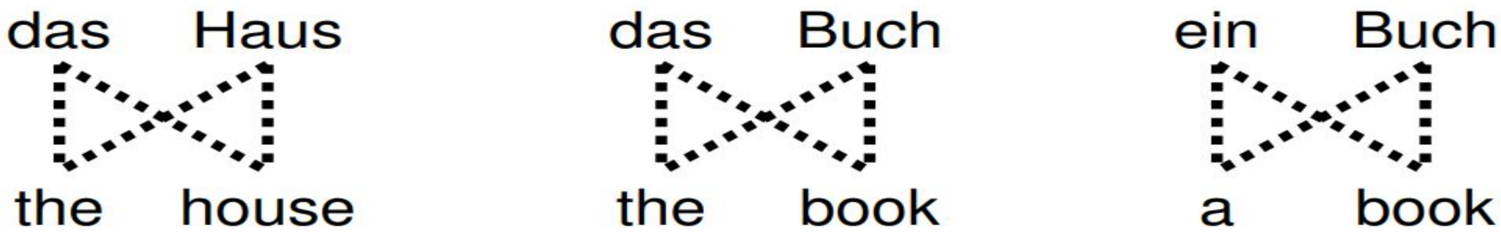
Word-based models

IBM Model 1 and EM: Pseudocode

```
Input: set of sentence pairs (e, f)
Output: translation prob. t(e|f)
initialize t(e|f) uniformly
while not converged do
    // initialize
    count(e|f) = 0 for all e, f
    total(f) = 0 for all f
    for all sentence pairs (e,f) do
        // compute normalization
        for all words e in e do
            s-total(e) = 0
            for all words f in f do
                s-total(e) += t(e|f)
            end for
        end for
        // collect counts
        for all words e in e do
            for all words f in f do
                count(e|f) += t(e|f) / s-total(e)
                total(f) += t(e|f) / s-total(e)
            end for
        end for
    end for
    // estimate probabilities
    for all foreign words f do
        for all English words e do
            t(e|f) = count(e|f) / total(f)
        end for
    end for
end while
```

Word-based models

Convergence



e	f	initial	1st it.	2nd it.	3rd it.	...	final
the	das	0.25	0.5	0.6364	0.7479	...	1
book	das	0.25	0.25	0.1818	0.1208	...	0
house	das	0.25	0.25	0.1818	0.1313	...	0
the	buch	0.25	0.25	0.1818	0.1208	...	0
book	buch	0.25	0.5	0.6364	0.7479	...	1
a	buch	0.25	0.25	0.1818	0.1313	...	0
book	ein	0.25	0.5	0.4286	0.3466	...	0
a	ein	0.25	0.5	0.5714	0.6534	...	1
the	haus	0.25	0.5	0.4286	0.3466	...	0
house	haus	0.25	0.5	0.5714	0.6534	...	1

Word-based models

Perplexity

- How well does the model fit the data?
- Perplexity: derived from probability of the training data according to the model

$$\log_2 PP = -\sum_s \log_2 p(\mathbf{e}_s|\mathbf{f}_s)$$

Example (ϵ=1)

	initial	1st it.	2nd it.	3rd it.	...	final
$p(\text{the haus}\mid\text{das haus})$	0.0625	0.1875	0.1905	0.1913	...	0.1875
$p(\text{the book}\mid\text{das buch})$	0.0625	0.1406	0.1790	0.2075	...	0.25
$p(\text{a book}\mid\text{ein buch})$	0.0625	0.1875	0.1907	0.1913	...	0.1875
perplexity	4095	202.3	153.6	131.6	...	113.8

fit@hcmus

Word-based models

Ensuring Fluent Output

- Our translation model cannot decide between **small** and **little**
- Sometime one is preferred over the other:
  - **small step**: 2,070,000 occurrences in the Google index
  - **little step:** 257,000 occurrences in the Google index
- Language model (LM)
  - estimate how likely a string is English
  - based on n-gram statistics

$$
\begin{aligned}
p(\mathbf{e}) &= p(e_1, e_2, ..., e_n) \\
&= p(e_1)p(e_2|e_1)...p(e_n|e_1, e_2, ..., e_{n-1}) \\
&\simeq p(e_1)p(e_2|e_1)...p(e_n|e_{n-2}, e_{n-1})
\end{aligned}
$$

Word-based models

Noisy Channel Model

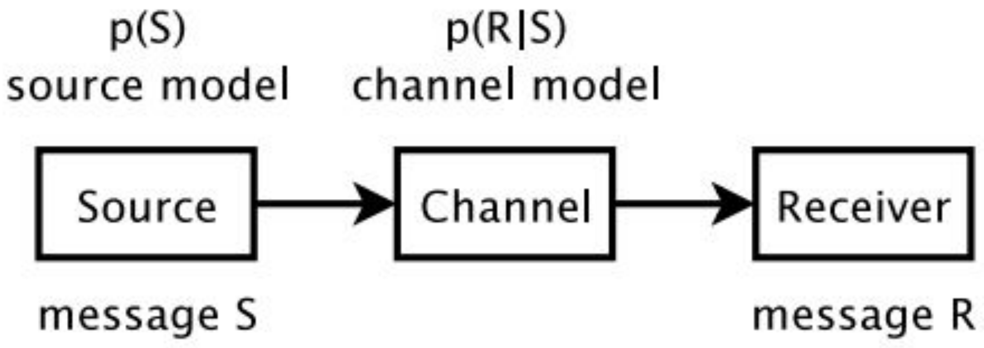
- We would like to integrate a language model
- Bayes rule

$$\text{argmax}_{\mathbf{e}}\, p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}} \frac{p(\mathbf{f}|\mathbf{e})\, p(\mathbf{e})}{p(\mathbf{f})}$$

$$= \text{argmax}_{\mathbf{e}}\, p(\mathbf{f}|\mathbf{e})\, p(\mathbf{e})$$

Word-based models

Noisy Channel Model



- Applying Bayes rule also called noisy channel model
  - we observe a distorted message R (here: a foreign string f)
  - we have a model on how the message is distorted (here: translation model)
  - we have a model on what messages are probably (here: language model)
  - we want to recover the original message S (here: an English string e)

Word-based models

Higher IBM Models

IBM Model 1	lexical translation
IBM Model 2	adds absolute reordering model
IBM Model 3	adds fertility model
IBM Model 4	relative reordering model
IBM Model 5	fixes deficiency

- Only IBM Model 1 has global maximum
    - Training of a higher IBM model builds on previous model
- Computationally biggest change in Model 3:
    - Trick to simplify estimation does not work anymore

-> exhaustive count collection becomes computationally too expensive

- sampling over high probability alignments is used instead

fit@hcmus

Word-based models

Reminder: 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

NLPA - Machine Translation

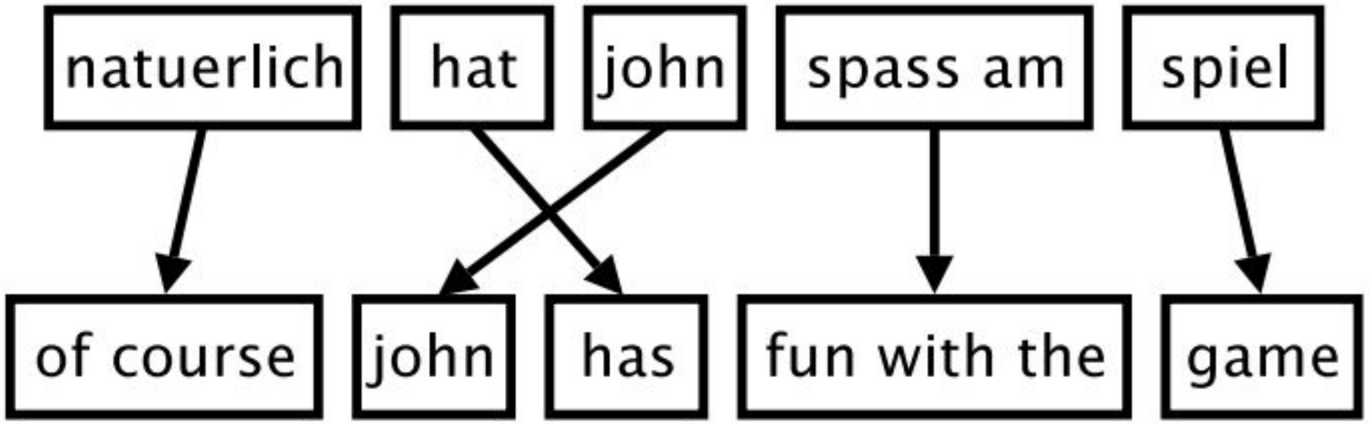
PHRASE-BASED MODELS

Phrase-based models

- **Motivation:**
  - Word-Based Models translate words as atomic units
  - Phrase-Based Models translate phrases as atomic units
  - Advantages
    - many-to-many translation can handle non-compositional phrases
    - use of local context in translation
    - the more data, the longer phrases can be learned
  - "Standard Model", used by Google Translate and others

Phrase-based models

- **Phrase-Based Model:**



- Foreign input is segmented in phrases
- Each phrase is translated into English
- Phrases are reordered

Phrase-based models

Phrase Translation Table:

- Main knowledge source: table with phrase translations and their probabilities
- Example: phrase translations for **natuerlich**

Translation	Probability $\phi(\bar{e}\mid f)$
of course	0.5
naturally	0.3
of course ,	0.15
, of course ,	0.05

Phrase-based models

Real Example

- Phrase translations for **den Vorschlag** learned from the Europarl corpus:

English	$\phi(\bar{e}\|\bar{f})$	English	$\phi(\bar{e}\|\bar{f})$
the proposal	0.6227	the suggestions	0.0114
's proposal	0.1068	the proposed	0.0114
a proposal	0.0341	the motion	0.0091
the idea	0.0250	the idea of	0.0091
this proposal	0.0227	the proposal ,	0.0068
proposal	0.0205	its proposal	0.0068
of the proposal	0.0159	it	0.0068
the proposals	0.0159	...	...

- lexical variation (proposal vs suggestions)
- morphological variation (proposal vs proposals)
- included function words (the,a,...)
- noise (it)

Phrase-based models

- **Linguistic Phrases?**
  - Model is not limited to linguistic phrases

  (noun phrases, verb phrases, prepositional phrases, ...)

  - Example **non-linguistic** phrase pair

  spass am -> fun with the

  - Prior noun often helps with translation of preposition
  - Experiments show that limitation to linguistic phrases hurts quality

Phrase-based models

- **Probabilistic Model**
  - Bayes rule

$$\mathbf{e}_{\text{best}} = \text{argmax}_{\mathbf{e}}\, p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}}\, p(\mathbf{f}|\mathbf{e})\, p_{\text{LM}}(\mathbf{e})$$

    - translation model **p(e|f)**
    - language model $p_{\text{LM}}(\mathbf{e})$

- Decomposition of the translation model

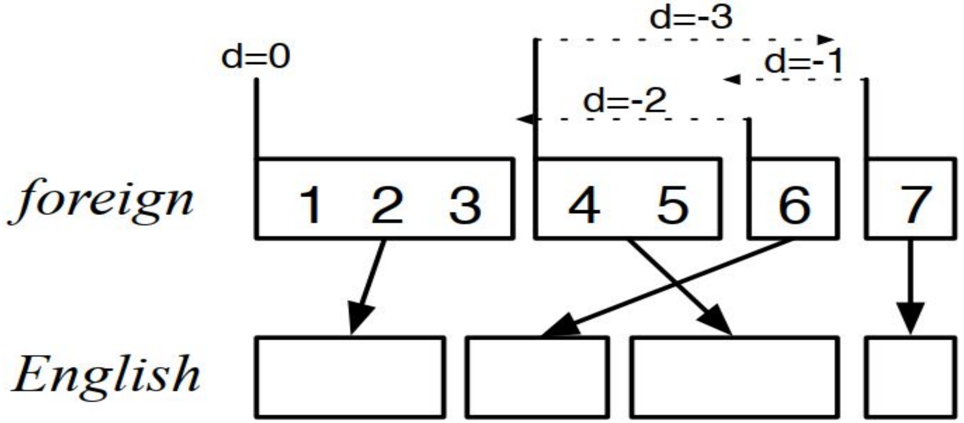
$$p(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\, d(start_i - end_{i-1} - 1)$$

  - phrase translation probability ϕ
  - reordering probability ***d***

fit@hcmus

Phrase-based models

- Distance-Based Reordering



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

Scoring function: $d(x) = \alpha^{|x|}$ -exponential with distance

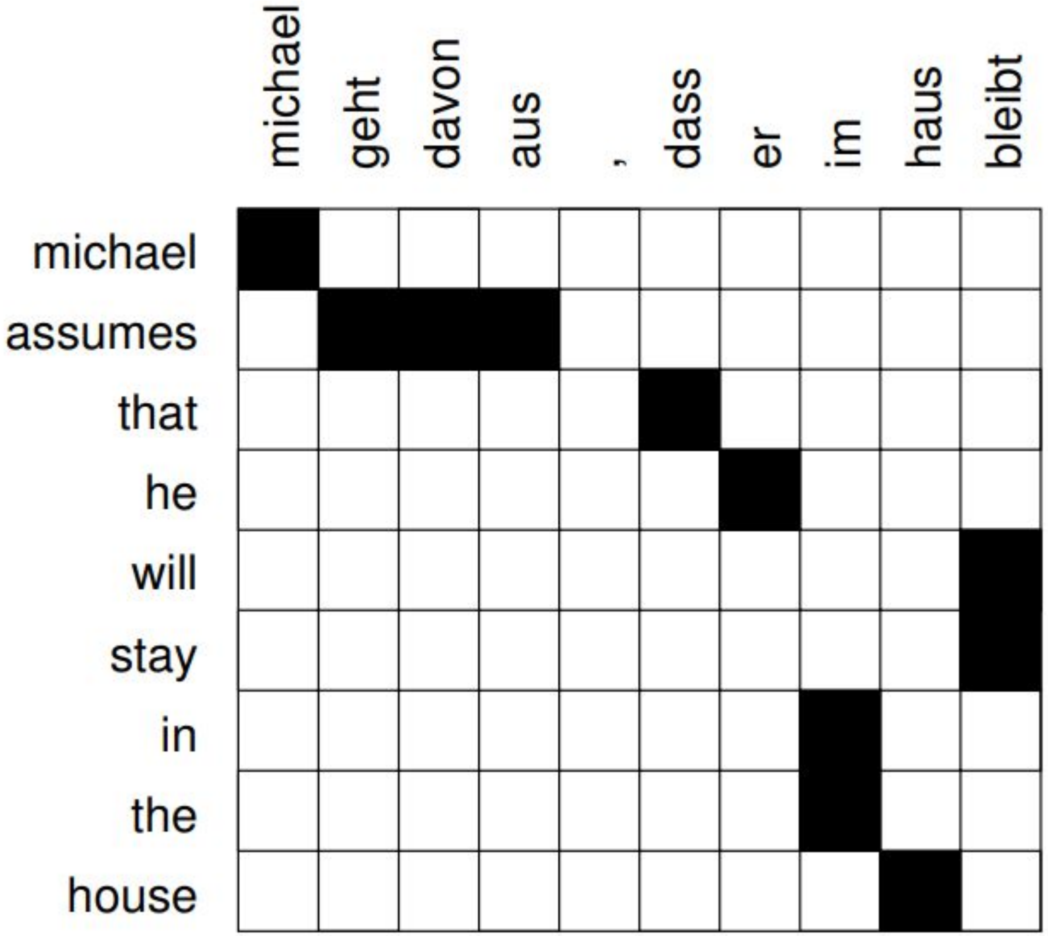
Phrase-based models

- Learn a Phrase Translation Table
  - Task: learn the model from a parallel corpus
  - Three stages:
    - word alignment: using IBM models or other method
    - extraction of phrase pairs
    - scoring phrase pairs

fit@hcmus

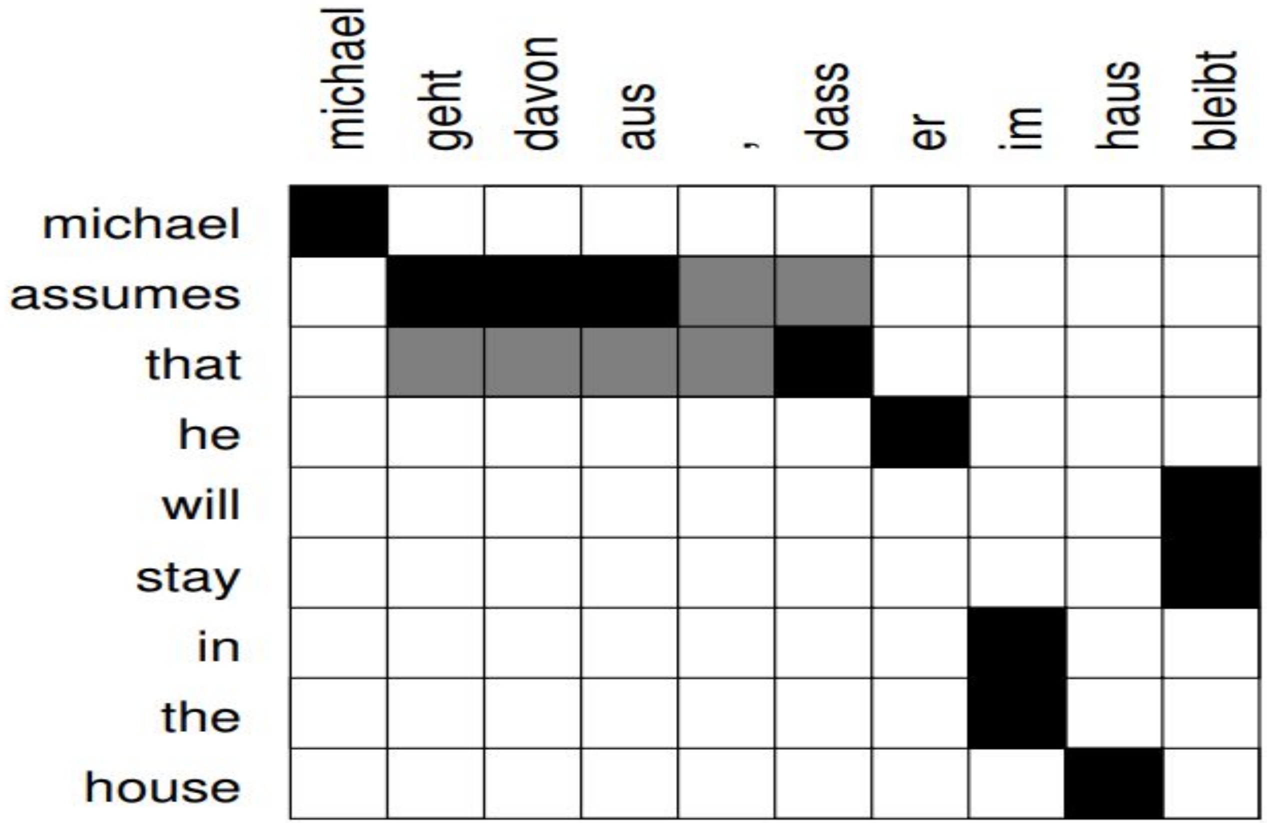
Phrase-based models

Word Alignment



Phrase-based models

Extracting Phrase Pairs

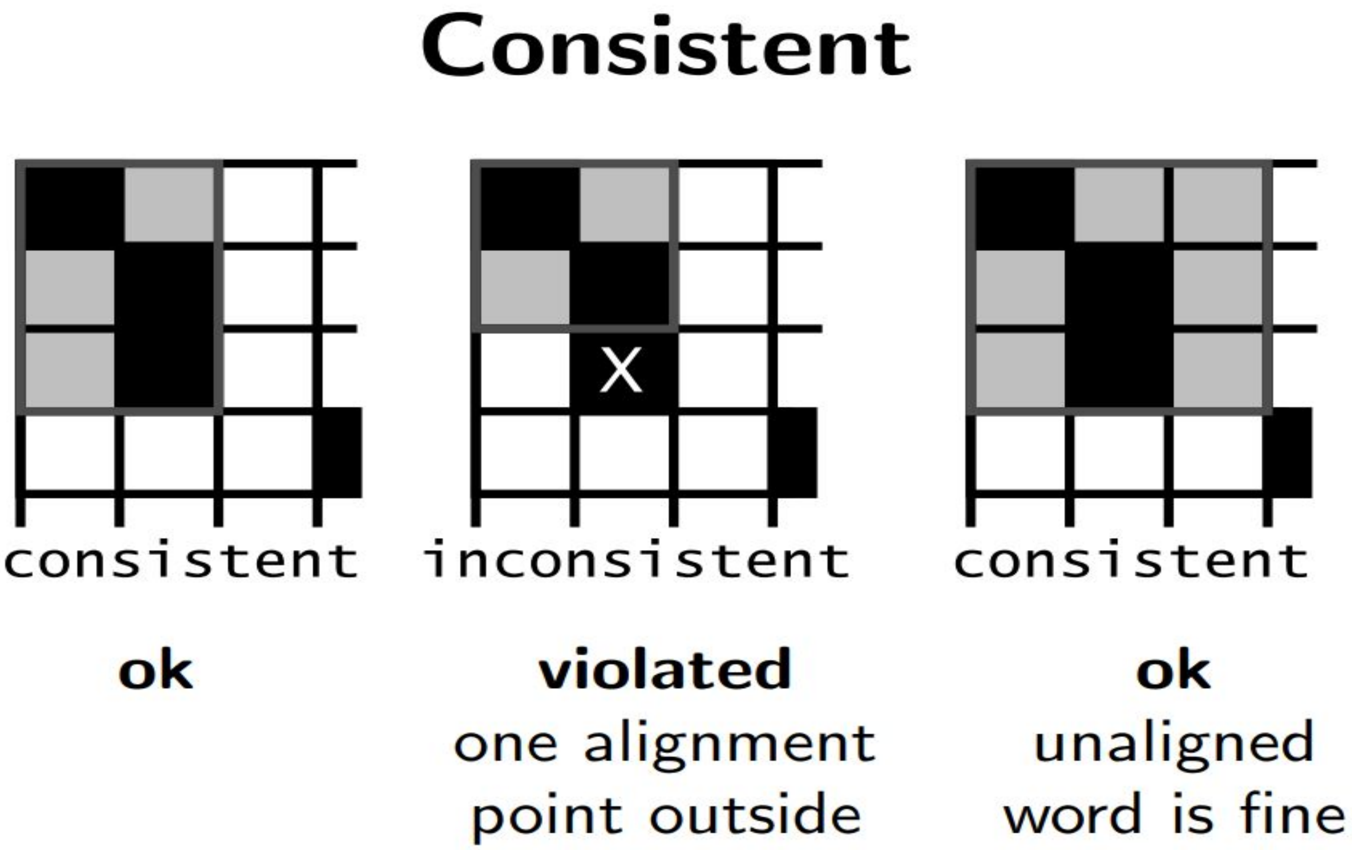


extract phrase pair consistent with word alignment:

assumes that / geht davon aus , dass

Phrase-based models

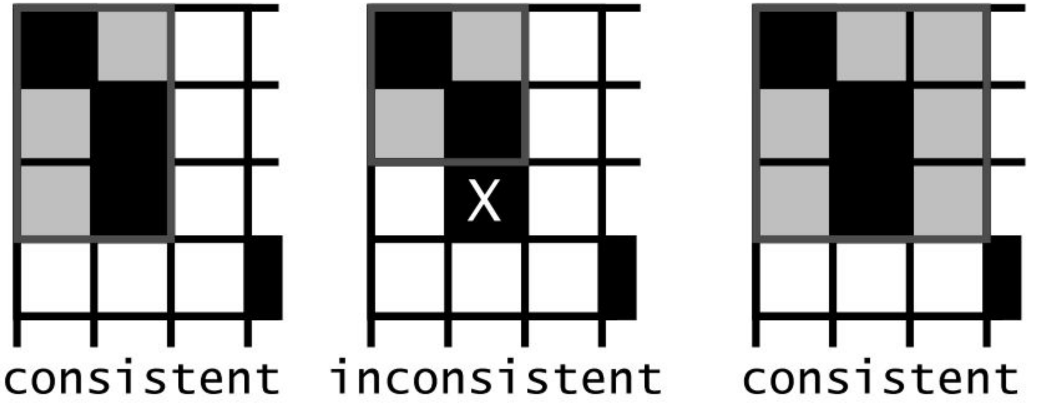
Consistent



All words of the phrase pair have to align to each other.

Phrase-based models

Consistent



Phrase pair $(\bar{e}, \bar{f})$ consistent with an alignment **A,** if all words $f_1, \ldots, f_n$ in $\bar{f}$ that have alignment points in **A** have these with words $e_1, \ldots, e_n$ in $\bar{e}$ and vice versa:

$$(\bar{e}, \bar{f}) \text{ consistent with } A \Leftrightarrow$$

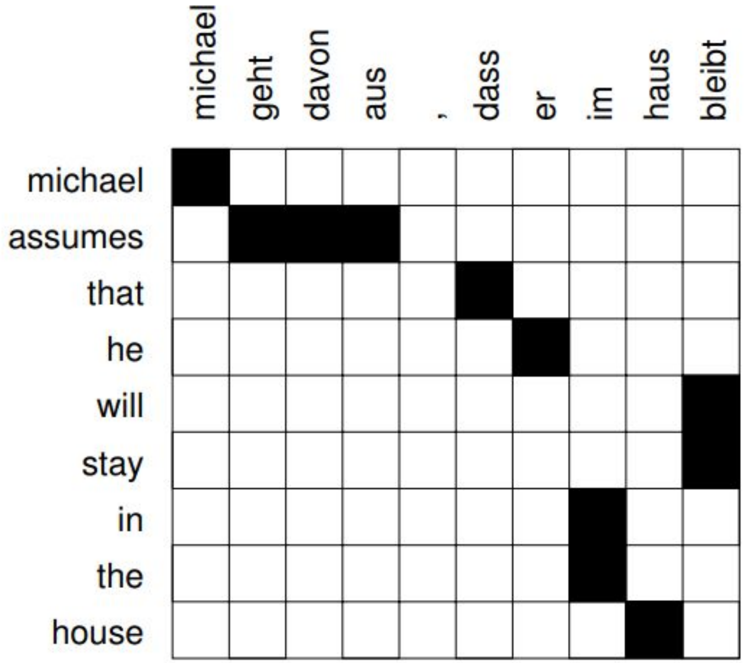
$$\forall e_i \in \bar{e} : (e_i, f_j) \in A \rightarrow f_j \in \bar{f}$$

$$\text{AND } \forall f_j \in \bar{f} : (e_i, f_j) \in A \rightarrow e_i \in \bar{e}$$

$$\text{AND } \exists e_i \in \bar{e}, f_j \in \bar{f} : (e_i, f_j) \in A$$

Phrase-based models

Phrase Pair Extraction



unaligned words (here: German comma) lead to multiple translations

Smallest phrase pairs:

michael — michael

assumes — geht davon aus / geht davon aus ,

that — dass / , dass

he — er

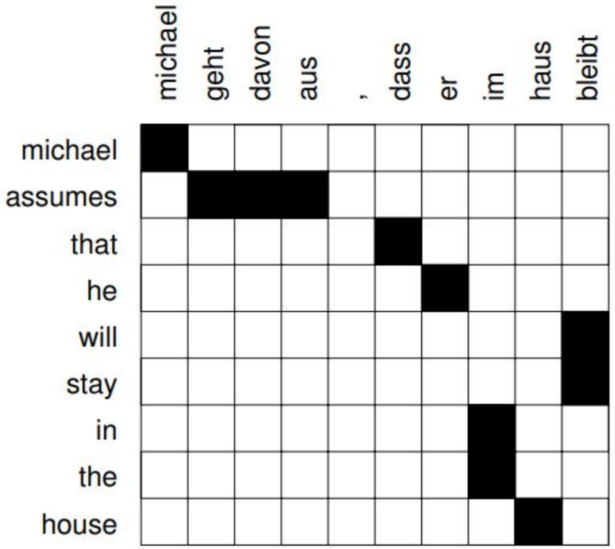
will stay — bleibt

in the — im

house — haus

Phrase-based models

Larger Phrase Pairs



michael assumes — michael geht davon aus / michael geht davon aus ,
assumes that — geht davon aus , dass ; assumes that he — geht davon aus , dass er
that he — dass er / , dass er ; in the house — im haus
michael assumes that — michael geht davon aus , dass
michael assumes that he — michael geht davon aus , dass er
michael assumes that he will stay in the house — michael geht davon aus , dass er im haus bleibt
assumes that he will stay in the house — geht davon aus , dass er im haus bleibt
that he will stay in the house — dass er im haus bleibt ; dass er im haus bleibt ,
he will stay in the house — er im haus bleibt ; will stay in the house — im haus bleibt

Phrase-based models

- Scoring Phrase Translations
  - Phrase pair extraction: collect all phrase pairs from the data
  - Phrase pair scoring: assign probabilities to phrase translations
  - Score by relative frequency:

$$\phi(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{e}, f)}{\sum_{\bar{f}_i} \text{count}(\bar{e}, \bar{f}_i)}$$

Phrase-based models

- Size of the Phrase Table
  - Phrase translation table typically bigger than corpus, even with limits on phrase lengths (e.g., max 7 words)

    -> Too big to store in memory?
  - Solution for training
    - extract to disk, sort, construct for one source phrase at a time
  - Solutions for decoding
    - on-disk data structures with index for quick look-ups
    - suffix arrays to create phrase pairs on demand

NLPA - Machine Translation

EVALUATION

Evaluation

- Automatic Evaluation Metrics
  - Goal: computer program that computes the quality of translations
  - Advantages: low cost, tunable, consistent
  - Basic strategy
    - given: machine translation output
    - given: human reference translation
    - task: compute similarity between them

Evaluation

❑ Precision and Recall of Words

SYSTEM A: Israeli officials ~~responsibility~~ ~~of~~ airport ~~safety~~

REFERENCE: Israeli officials are responsible for airport security

- Precision $$\frac{correct}{output\text{-}length} = \frac{3}{6} = 50\%$$

- Recall $$\frac{correct}{reference\text{-}length} = \frac{3}{7} = 43\%$$

- F-measure $$\frac{precision \times recall}{(precision + recall)/2} = \frac{.5 \times .43}{(.5 + .43)/2} = 46\%$$

fit@hcmus

Evaluation

- Precision and Recall

SYSTEM A: Israeli officials ~~responsibility~~ ~~of~~ airport ~~safety~~

REFERENCE: Israeli officials are responsible for airport security

SYSTEM B: airport security Israeli officials are responsible

Metric	System A	System B
precision	50%	100%
recall	43%	100%
f-measure	46%	100%

flaw: no penalty for reordering

Evaluation

- **Word Error Rate:**
  - Minimum number of editing steps to transform output to reference
    - match: words match, no cost
    - substitution: replace one word with another
    - insertion: add word
    - deletion: drop word
  - Levenshtein distance

$$\text{WER} = \frac{\textit{substitutions} + \textit{insertions} + \textit{deletions}}{\textit{reference-length}}$$

Evaluation

- **Word Error Rate:**

Example

		Israeli	officials	responsibility	of	airport	safety
	0	1	2	3	4	5	6
Israeli	1	0	1	2	3	4	5
officials	2	1	0	1	2	3	4
are	3	2	1	1	2	3	4
responsible	4	3	2	2	2	3	4
for	5	4	3	3	3	3	4
airport	6	5	4	4	4	3	4
security	7	6	5	5	5	4	4

		airport	security	Israeli	officials	are	responsible
	0	1	2	3	4	5	6
Israeli	1	1	2	2	3	4	5
officials	2	2	2	3	2	3	4
are	3	3	3	3	3	2	3
responsible	4	4	4	4	4	3	2
for	5	5	5	5	5	4	3
airport	6	5	6	6	6	5	4
security	7	6	5	6	7	6	5

Metric	System A	System B
word error rate (WER)	57%	71%

fit@hcmus

Evaluation

- **BLEU (Bilingual Language Evaluation Understudy):**
  - N-gram overlap between machine translation output and reference translation
  - Compute precision for n-grams of size 1 to 4
  - Add brevity penalty (for too short translations)

$$\text{BLEU} = \min\left(1, \frac{\textit{output-length}}{\textit{reference-length}}\right) \left(\prod_{i=1}^{4} \textit{precision}_i\right)^{\frac{1}{4}}$$

  - Typically computed over the entire corpus, not single sentences

fit@hcmus

Evaluation

BLEU:

Example

SYSTEM A: Israeli officials responsibility of airport safety
2-GRAM MATCH 1-GRAM MATCH

REFERENCE: Israeli officials are responsible for airport security

SYSTEM B: airport security Israeli officials are responsible
2-GRAM MATCH 4-GRAM MATCH

Metric	System A	System B
precision (1gram)	3/6	6/6
precision (2gram)	1/5	4/5
precision (3gram)	0/4	2/4
precision (4gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0%	52%

Evaluation

- **Multiple Reference Translations**
  - To account for variability, use multiple reference translations
    - n-grams may match in any of the references
    - closest reference length used
    - Example

SYSTEM: Israeli officials responsibility of airport safety

2-GRAM MATCH 2-GRAM MATCH 1-GRAM

REFERENCES:

Israeli officials are responsible for airport security

Israel is in charge of the security at this airport

The security work for this airport is the responsibility of the Israel government

Israeli side was in charge of the security of this airport

Evaluation

- **METEOR: flexible matching**
  - Partial credit for matching stems

SYSTEM	Jim went home
REFERENCE	Joe goes home

  - Partial credit for matching synonyms

SYSTEM	Jim walks home
REFERENCE	Joe goes home

  - Use of paraphrases

fit@hcmus

Evaluation

- Critique of Automatic Metrics
  - Ignore relevance of words

    (names and core concepts more important than determiners and punctuation)
  - Operate on local level

    (do not consider overall grammaticality of the sentence or sentence meaning)
  - Scores are meaningless (scores very test-set specific, absolute value not informative)
  - Human translators score low on BLEU

    (possibly because of higher variability, different word choices)

Evaluation

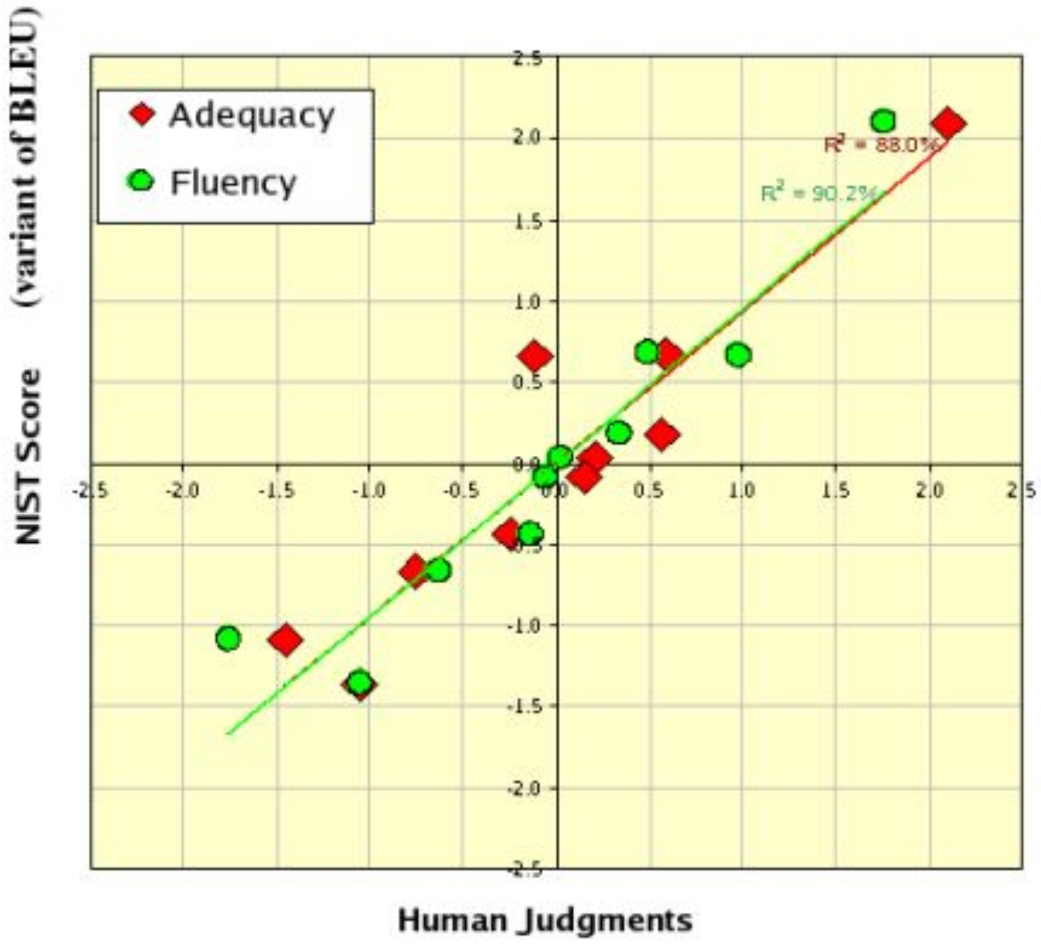
Evaluation of Evaluation Metrics

- Automatic metrics are low cost, tunable, consistent
- But are they correct?

-> Yes, if they correlate with human judgement

Evaluation

- Correlation with Human Judgement



fit@hcmus

Evaluation

- Pearson's Correlation Coefficient
  - Two variables: automatic score x, human judgment y
  - Multiple systems : $(x_1, y_1), (x_2, y_2), \ldots$
  - Pearson's correlation coefficient r_{xy}

$$r_{xy} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{(n-1)\, s_x\, s_y}$$

  - Note:

$$\text{mean} \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\text{variance} \quad s_x^2 = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2$$

Evaluation

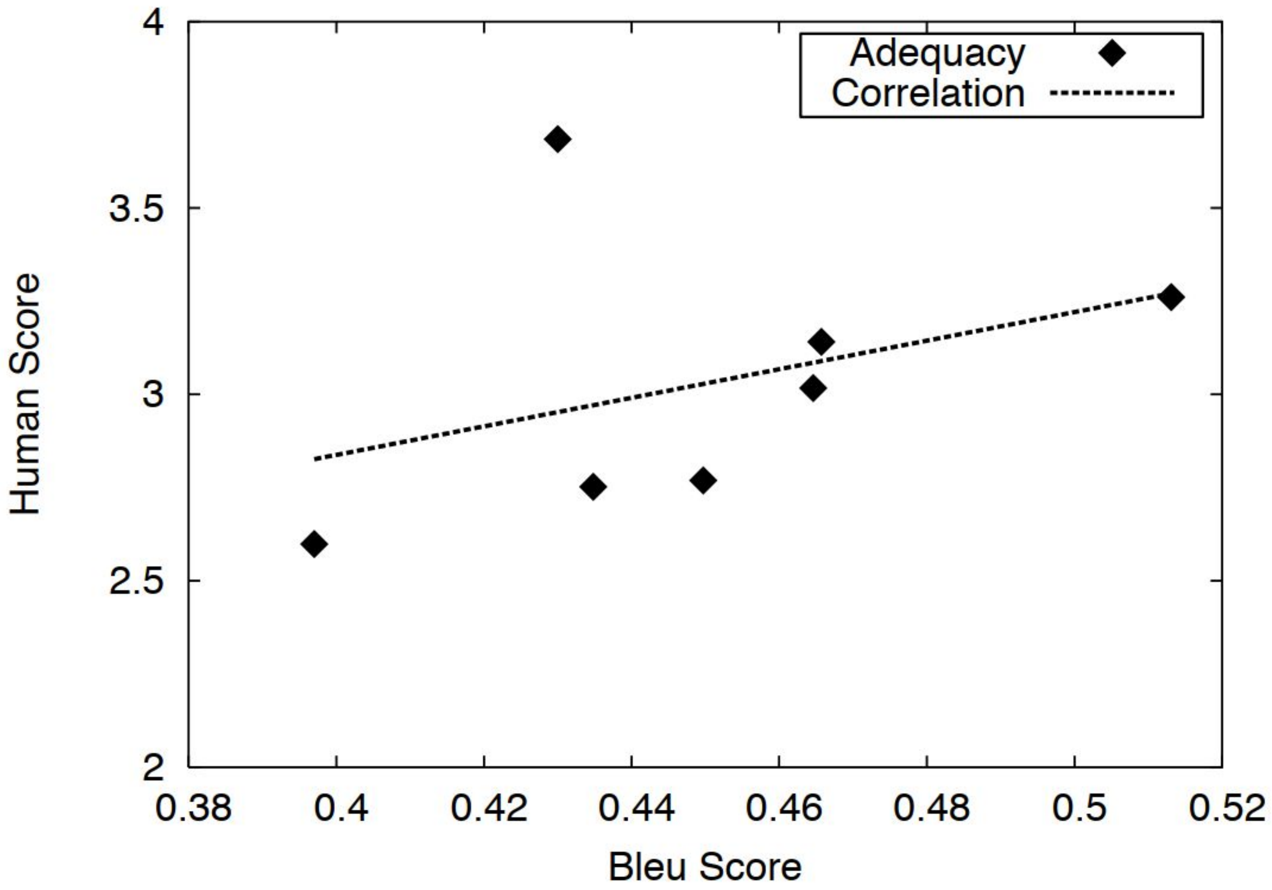
- Metric Research
  - Active development of new metrics
    - syntactic similarity
    - semantic equivalence or entailment
    - metrics targeted at reordering
    - trainable metrics
    - etc.
  - Evaluation campaigns that rank metrics

    (using Pearson's correlation coefficient)

Evaluation

❑ Evidence of Shortcomings of Automatic Metrics

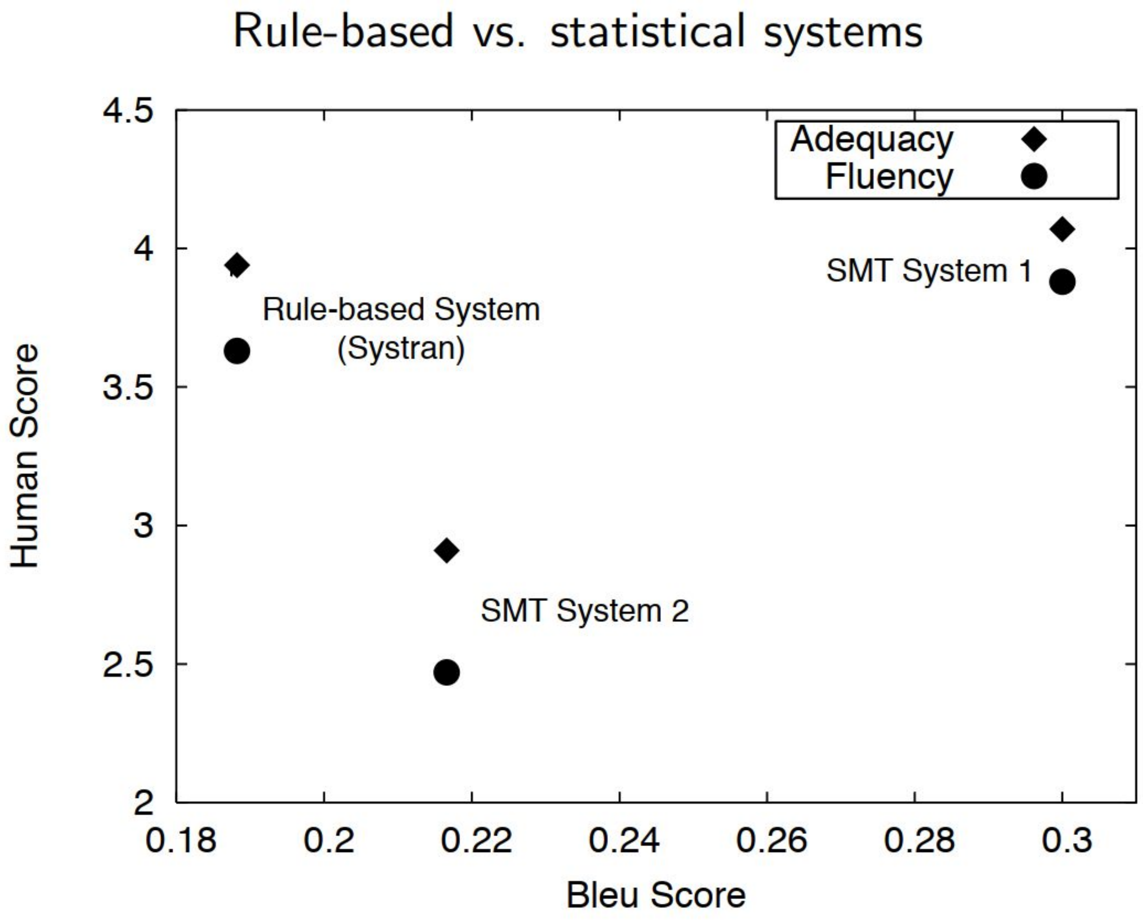
Post-edited output vs. statistical systems (NIST 2005)



fit@hcmus

Evaluation

❑ Evidence of Shortcomings of Automatic Metrics



fit@hcmus

Evaluation

- **Automatic Metrics: Conclusions**
  - Automatic metrics essential tool for system development
  - Not fully suited to rank systems of different types
  - Evaluation metrics still open challenge