Philipp Koehn and Chris Callison-Burch

Statistical Machine Translation

Course Material. 20th European Summer School in Logic, Language and Information (ESSLLI 2008), Freie und Hansestadt Hamburg, Germany, 4–15 August 2008

Intro to Statistical MT

EuroMatrix

MT Marathon

Chris Callison-Burch

Various approaches

- Word-for-word translation
- Syntactic transfer
- Interlingual approaches
- Controlled language
- Example-based translation
- Statistical translation

Advantages of SMT

- Data driven
- Language independent
- No need for staff of linguists of language experts
- Can prototype a new system quickly and at a very low cost

Statistical machine translation

- Find most probable English sentence given a foreign language sentence
- Automatically align words and phrases within sentence pairs in a parallel corpus
- Probabilities are determined automatically by training a statistical model using the parallel corpus

Parallel corpus

what is more , the relevant cost dynamic is completely under control.	im übrigen ist die diesbezügliche kostenentwicklung völlig unter kontrolle .
sooner or later we will have to be sufficiently progressive in terms of own resources as a basis for this fair tax system .	früher oder später müssen wir die notwendige progressivität der eigenmittel als grundlage dieses gerechten steuersystems zur sprache bringen .
we plan to submit the first accession partnership in the autumn of this year .	wir planen , die erste beitrittspartnerschaft im herbst dieses jahres vorzulegen .
it is a question of equality and solidarity .	hier geht es um gleichberechtigung und solidarität .
the recommendation for the year 1999 has been formulated at a time of favourable developments and optimistic prospects for the european economy .	die empfehlung für das jahr 1999 wurde vor dem hintergrund günstiger entwicklungen und einer für den kurs der europäischen wirtschaft positiven perspektive abgegeben .
that does not , however , detract from the deep appreciation which we have for this report .	im übrigen tut das unserer hohen wertschätzung für den vorliegenden bericht keinen abbruch .

Probabilities

- Find most probable English sentence given a foreign language sentence

$$p(e|f)$$

$$\hat{e} = \arg\max_e \; p(e|f)$$

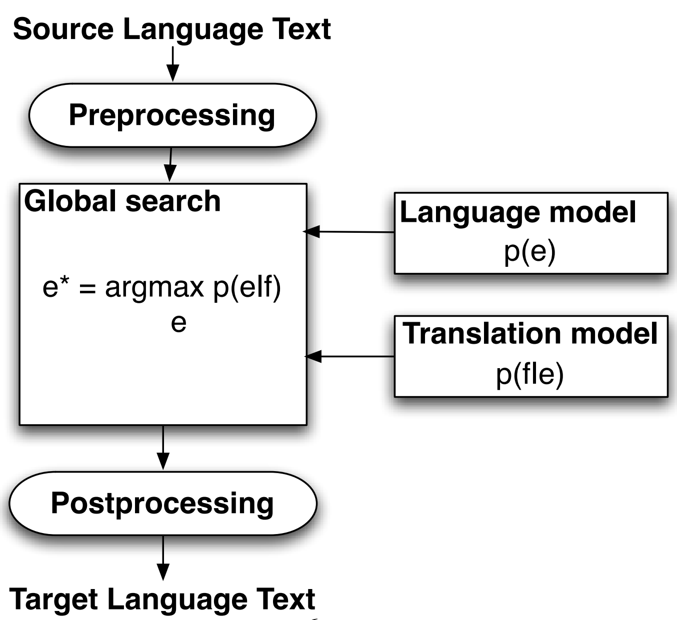
$$p(e|f) = \frac{p(e)p(f|e)}{p(f)}$$

$$\hat{e} = \arg\max_e \; p(e)p(f|e)$$

What the probabilities represent

- p(e) is the "Language model"
  - Assigns a higher probability to fluent / grammatical sentences
  - Estimated using monolingual corpora
- p(f|e) is the "Translation model"
  - Assigns higher probability to sentences that have corresponding meaning
  - Estimated using bilingual corpora

For people who don't like equations



Language Model

- Component that tries to ensure that words come in the right order
- Some notion of grammaticality
- Standardly calculated with a trigram language model, as in speech recognition
- Could be calculated with a statistical grammar such as a PCFG

Trigram language model

- p(I like bungee jumping off high bridges) =
  p(I | <s> <s>) *
  p(like | I <s>) *
  p(bungee | I like) *
  p(jumping | like bungee) *
  p(off | bungee jumping) *
  p(high | jumping off) *
  p(bridges | off high) *
  p(</s> | high bridges) *
  p(</s> | bridges </s>)

Calculating Language Model Probabilities

- Unigram probabilities

$$p(w_1) = \frac{count(w_1)}{total\ words\ observed}$$

Calculating Language Model Probabilities

- Bigram probabilities

$$p(w_2|w_1) = \frac{count(w_1 w_2)}{count(w_1)}$$

Calculating Language Model Probabilities

- Trigram probabilities

$$p(w_3|w_1w_2) = \frac{count(w_1w_2w_3)}{count(w_1w_2)}$$

Calculating Language Model Probabilities

- Can take this to increasingly long sequences of n-grams
- As we get longer sequences it's less likely that we'll have ever observed them

Backing off

- Sparse counts are a big problem
- If we haven't observed a sequence of words then the count = 0
- Because we're multiplying the n-gram probabilities to get the probability of a sentence the whole probability = 0

Backing off

$$
\begin{aligned}
&.8 * p(w_3|w_1 w_2) + \\
&.15 * p(w_3|w_2) + \\
&.049 * p(w_3) + \\
&.001
\end{aligned}
$$

- Avoids zero probs

Translation model

- p(f|e)... the probability of some foreign language string given a hypothesis English translation
- f = Ces gens ont grandi, vécu et oeuvré des dizaines d'années dans le domaine agricole.
- e = *Those people have grown up, lived and worked many years in a farming district.*
- e = *I like bungee jumping off high bridges.*

Translation model

- How do we assign values to p(f|e)?

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

- Impossible because sentences are novel, so we'd never have enough data to find values for all sentences.

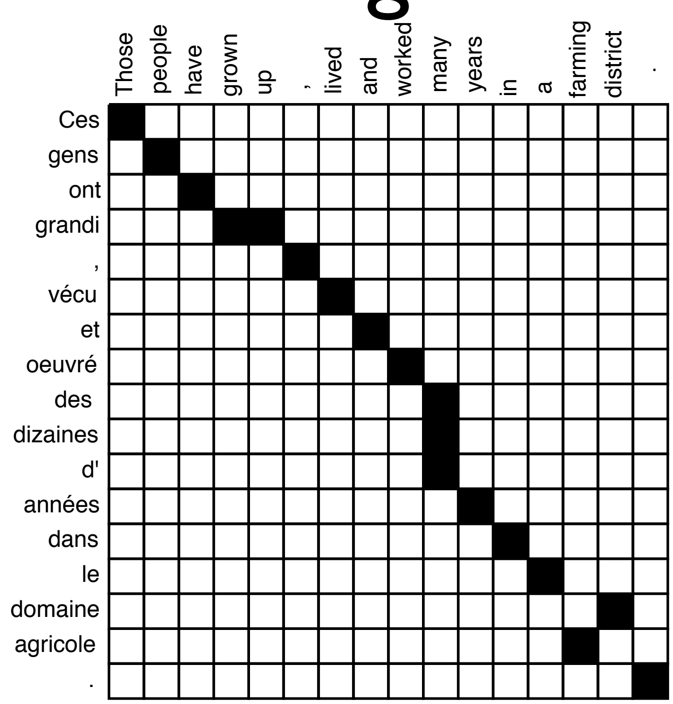
Translation model

- Decompose the sentences into smaller chunks, like in language modeling

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

- Introduce another variable *a* that represents alignments between the individual words in the sentence pair

Word alignment



Alignment probabilities

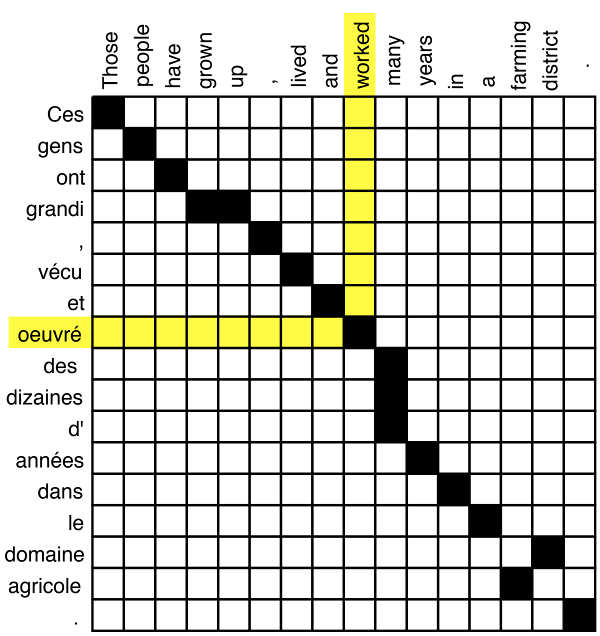
- So we can calculate translation probabilities by way of these alignment probabilities

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

- Now we need to define p(a, f | e)

$$p(a, f|e) = \prod_{j=1}^{m} t(f_j|e_i)$$

Calculating $t(f_j|e_i)$



- Counting! I told you probabilities were easy!

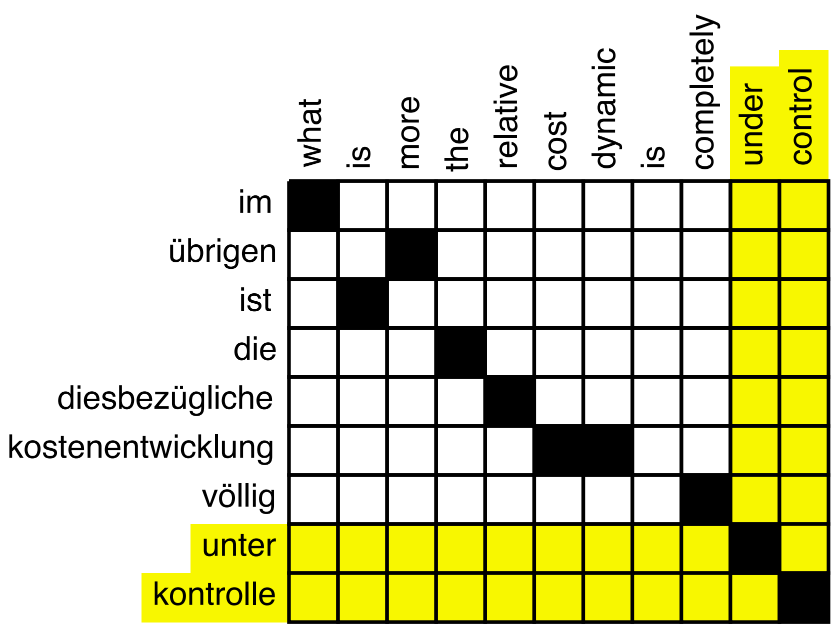
$$= \frac{count(f_j, e_i)}{count(e_i)}$$

- **worked... fonctionné, travaillé, marché, oeuvré**
- 100 times total 13 with this f. 13%

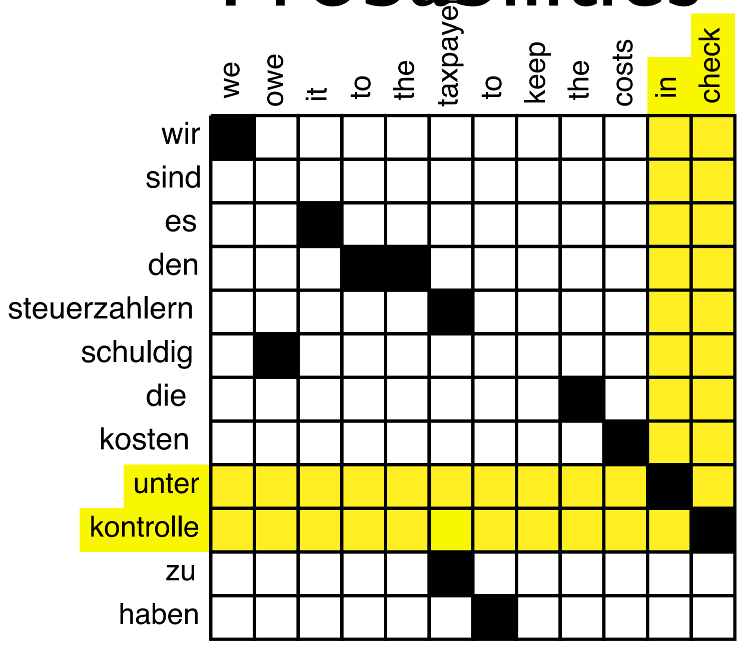
Calculating $t(f_j|e_i)$

- Unfortunately we don't have word aligned data, so we can't do this directly.
- OK, so it's not quite as easy as I said.
- There will be another lecture on how to do word alignments later in the week.

Phrase Translation Probabilities



Phrase Translation Probabilities

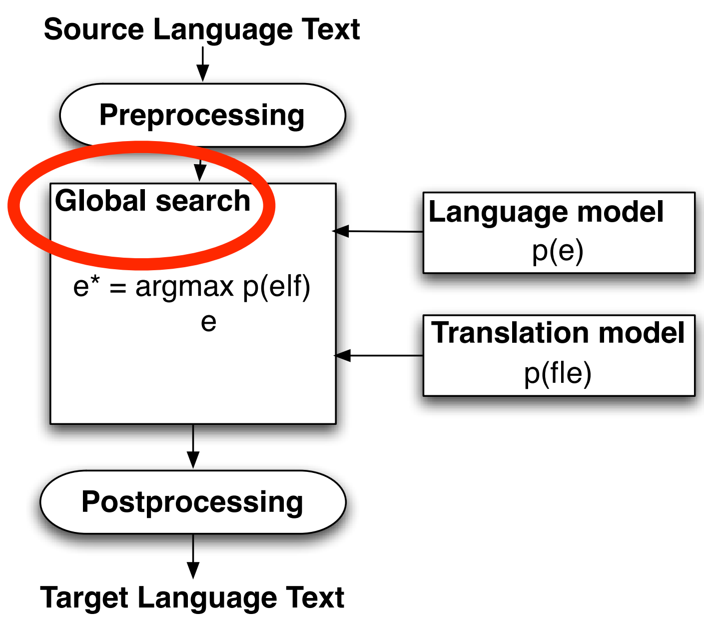


Phrase Table

- Exhaustive table of source language phrases paired with their possible translations into the target language, along with probabilities

das thema	the issue	.51
	the point	.38
	the subject	.21

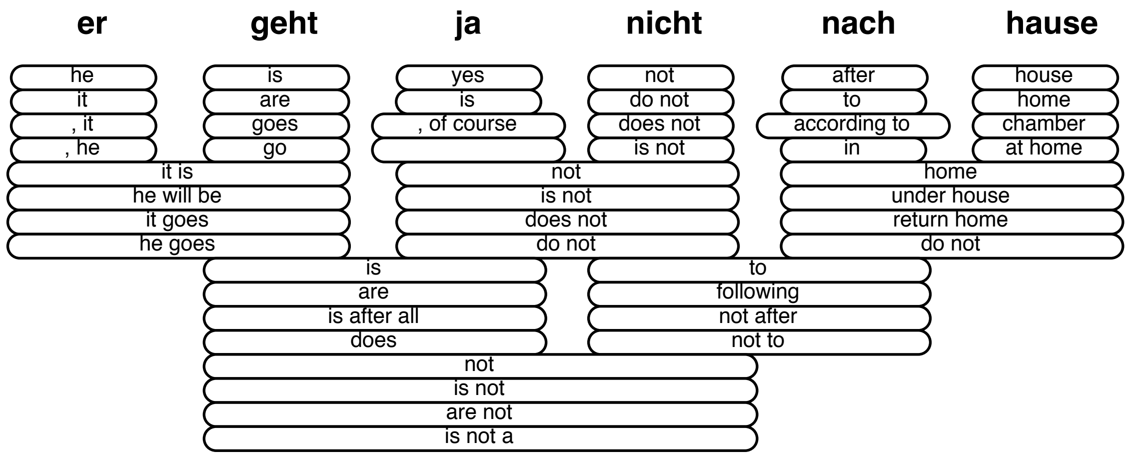
``Diagram Number 1"



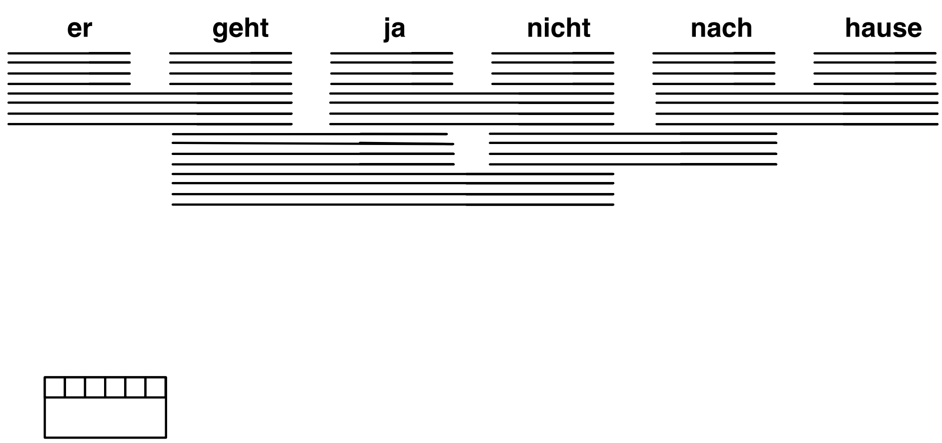
The Search Process AKA ``Decoding"

- Look up all translations of every source phrase, using the phrase table
- Recombine the target language phrases that maximizes the translation model probability * the language model probability
- This search over all possible combinations can get very large so we need to find ways of limiting the search space

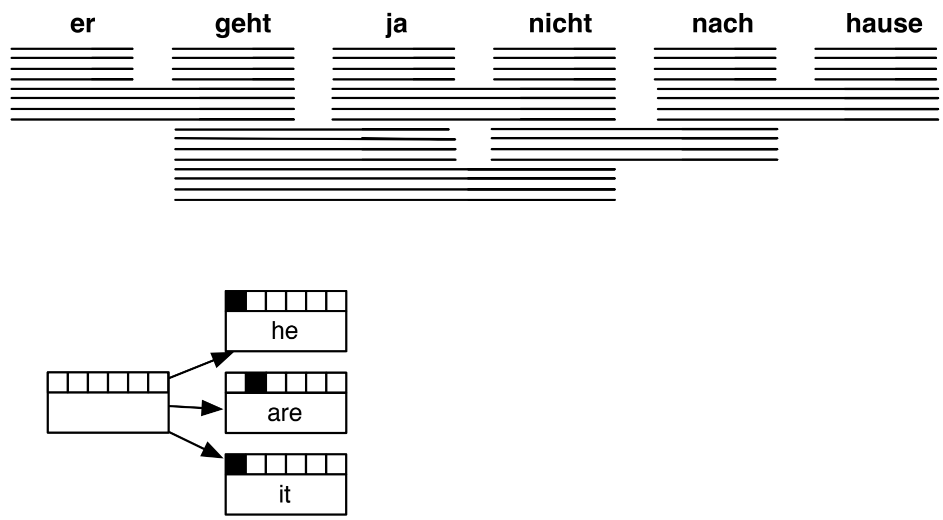
Translation Options



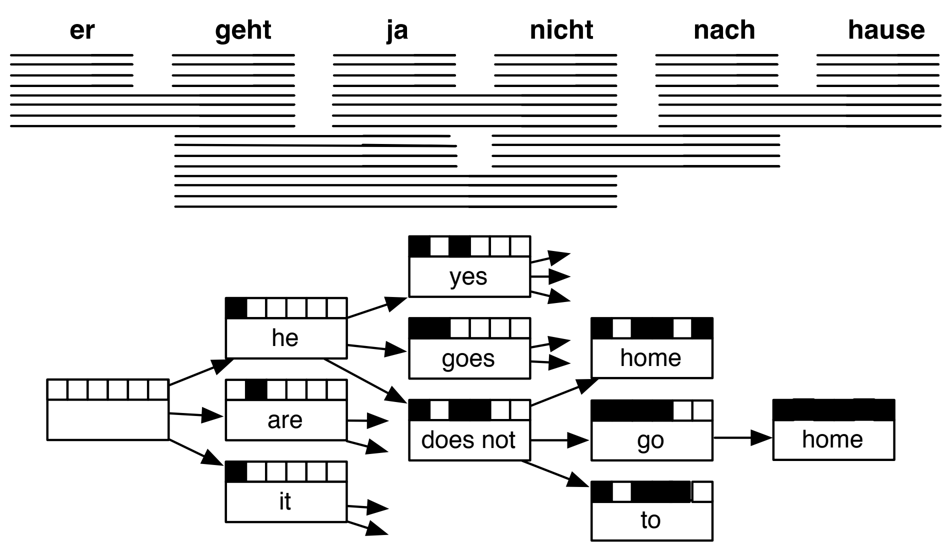
Search



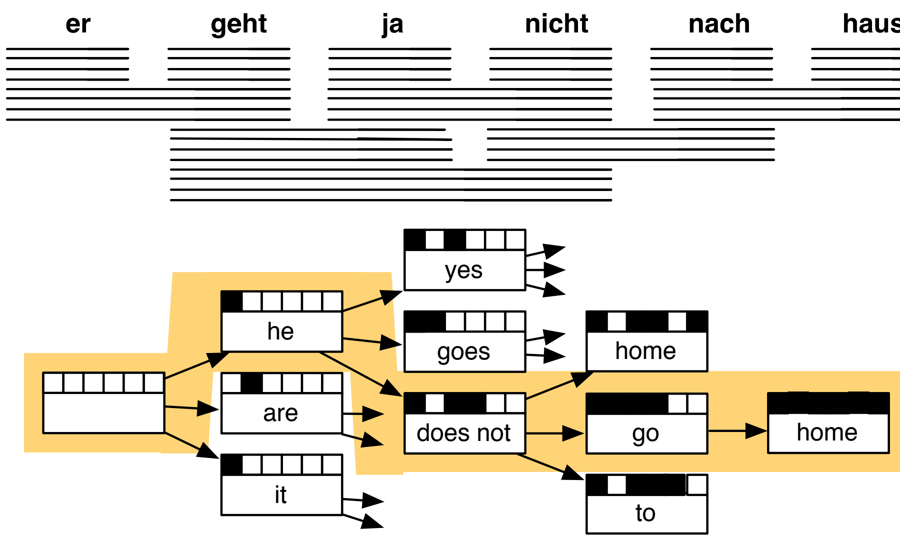
Search



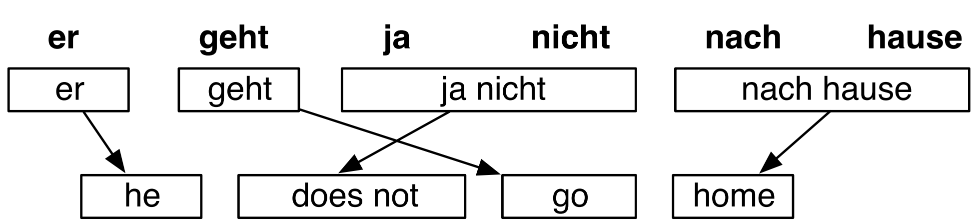
Search



Search



Best Translation



The Search Space

- In the end the item which covers all of the source words and which has the highest probability wins!
- That's our best translation

$$\hat{e} = \arg\max_{e} \; p(e)p(f|e)$$

- And there was much rejoicing!

Wrap-up: SMT is data driven

- Learns translations of words and phrases from parallel corpora
- Associate probabilities with translations empirically by counting co-occurrences in the data
- Estimates of probabilities get more accurate as size of the data increases

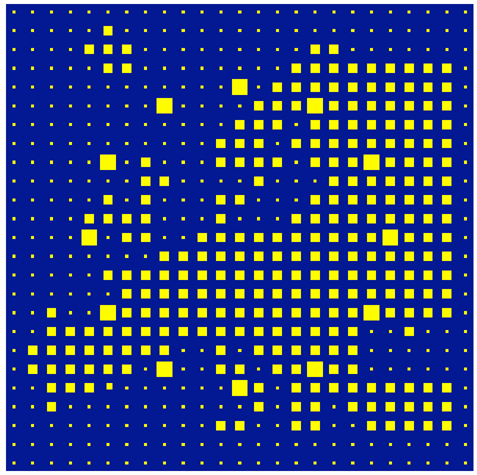
Wrap-up: SMT is language independent

- Can be applied to any language pairs that we have a parallel corpus for
- The only linguistic thing that we need to know is how to split into sentences, words
- Don't need linguists and language experts to hand craft rules because it's all derived from the data

Wrap-up: SMT is cheap and quick to produce

- Low overhead since we aren't employing anyone
- Computers do all the heavy lifting / statistical analysis of the data for us
- Can build a system in hours or days rather than months or years

Evaluating Translation Quality



EuroMatrix
MT Marathon
Chris Callison-Burch

Evaluating MT Quality

- Why do we want to do it?
  - Want to rank systems
  - Want to evaluate incremental changes
- How not to do it
  - ``Back translation"
  - The vodka is *not* good

Evaluating Human Translation Quality

- Why?
  - Quality control
  - Decide whether to re-hire freelance translators
  - Career promotion

DLPT-CRT

- Defense Language Proficiency Test/ Constructed Response Test
- Read texts of varying difficulty, take test
- Structure of test
  - Limited responses for questions
  - Not multiple choice, not completely open
  - Test progresses in difficulty
  - Designed to assign level at which examinee fails to sustain proficiency

DLPT-CRT

- Level 1: Contains short, discrete, simple sentences. Newspaper announcements.
- Level 2: States facts with purpose of conveying information. Newswire stories.
- Level 3: Has denser syntax, convey opinions with implications. Editorial articles / opinion.
- Level 4: Often has highly specialized terminology. Professional journal articles.

Human Evaluation of Machine Translation

- One group has tried applying DLPT-CRT to machine translation
  - Translate texts using MT system
  - Have monolingual individuals take test
  - See what level they perform at
- Much more common to have human evaluators simply assign a scale directly using fluency / adequacy scales

Fluency

- 5 point scale
- 5) Flawless English
  4) Good English
  3) Non-native English
  2) Disfluent
  1) Incomprehensible

Adequacy

- This text contains how much of the information in the reference translation:
- 5) All
  4) Most
  3) Much
  2) Little
  1) None

Human Evaluation of MT v. Automatic Evaluation

- Human evaluation is
  - Ultimately what we're interested in, *but*
  - Very time consuming
  - Not re-usable
- Automatic evaluation is
  - Cheap and reusable, *but*
  - Not necessarily reliable

Goals for Automatic Evaluation

- No cost evaluation for incremental changes
- Ability to rank systems
- Ability to identify which sentences we're doing poorly on, and categorize errors
- Correlation with human judgments
- Interpretability of the score

Methodology

- Comparison against reference translations
- Intuition: closer we get to human translations, the better we're doing
- Could use WER like in speech recognition

Word Error Rate

- Levenshtein Distance (also "edit distance")
- Minimum number of insertions, substitutions, and deletions needed to transform one string into another
- Useful measure in speech recognition
  - *Shows how easy it is to recognize speech*
  - *Shows how easy it is to wreck a nice beach*

Problems with WER

- Unlike speech recognition we don't have the assumptions of
  - linearity
  - exact match against the reference
- In machine translation there can be many possible (and equally valid) ways of translating a sentence
- Also, clauses can move around, since we're not doing transcription

Solutions

- Compare against lots of test sentences
- Use multiple reference translations for each test sentence
- Look for phrase / n-gram matches, allow movement

Metrics

- Exact sentence match
- WER
- PI-WER
- Bleu
- Precision / Recall
- Meteor

Bleu

- Use multiple reference translations
- Look for n-grams that occur anywhere in the sentence
- Also has ``brevity penalty''
- Goal: Distinguish which system has better quality (correlation with human judgments)

Example Bleu

R1: It is a guide to action that ensures that the military will forever heed Party commands.
R2: It is the Guiding Principle which guarantees the military forces always being under the command of the Party.
R3: It is the practical guide for the army always to heed the directions of the party.

C1: It is to insure the troops forever hearing the activity guidebook that party direct.
C2: It is a guide to action which ensures that the military always obeys the command of the party.

Example Bleu

R1: It is a guide to action that ensures that the military will forever heed Party commands.
R2: It is the Guiding Principle which guarantees the military forces always being under the command of the Party.
R3: It is the practical guide for the army always to heed the directions of the party.

C1: It is to insure the troops forever hearing the activity guidebook that party direct.

Example Bleu

R1: It is a guide to action that ensures that the military will forever heed Party commands.
R2: It is the Guiding Principle which guarantees the military forces always being under the command of the Party.
R3: It is the practical guide for the army always to heed the directions of the party.

C2: It is a guide to action which ensures that the military always obeys the command of the party.

Automated evaluation

- Because **C2** has more n-grams and longer n-grams than **C1** it receives a higher score
- Bleu has been shown to correlate with human judgments of translation quality
- Bleu has been adopted by DARPA in its annual machine translation evaluation

Interpretability of the score

- How many errors are we making?
- How much better is one system compared to another?
- How useful is it?
- How much would we have to improve to be useful?

Evaluating an evaluation metric

- How well does it correlate with human judgments?
  - On a system level
  - On a per sentence level
- Data for testing correlation with human judgments of translation quality

NIST MT Evaluation

- Annual Arabic-English and Chinese-English competitions
- 10 systems
- 1000+ sentences each
- Scored by Bleu and human judgments
- Human judgments for translations produced by each system

Final thoughts on Evaluation

When writing a paper

- If you're writing a paper that claims that
  - one approach to machine translation is better than another, or that
  - some modification you've made to a system has improved translation quality
- Then you need to back up that claim
- Evaluation metrics can help, but good experimental design is also critical

Experimental Design

- Importance of separating out training / test / development sets
- Importance of standardized data sets
- Importance of standardized evaluation metric
- Error analysis
- Statistical significance tests for differences between systems

Invent your own evaluation metric

- If you think that Bleu is inadequate then invent your own automatic evaluation metric
- Can it be applied automatically?
- Does it correlate better with human judgment?
- Does it give a finer grained analysis of mistakes?

Evaluation drives MT research

- Metrics can drive the research for the topics that they evaluate
- NIST MT Eval / DARPA Sponsorship
- Bleu has lead to a focus on phrase-based translation
- Minimum error rate training
- Other metrics may similarly change the community's focus

Afternoon Exercise

- Evaluation exercise this afternoon
- Examine translations from state-of-the-art systems (in the language of your choice!)
- Manually evaluate quality!
- Perform error analysis!
- Develop ideas about how to improve SMT!

ESSLLI Summer School 2008

Day 2: Word-based models and the EM algorithm

Philipp Koehn, University of Edinburgh

Day 2

Lexical translation

- How to translate a word → look up in 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: During all the lectures, we will translate from a foreign language into English

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

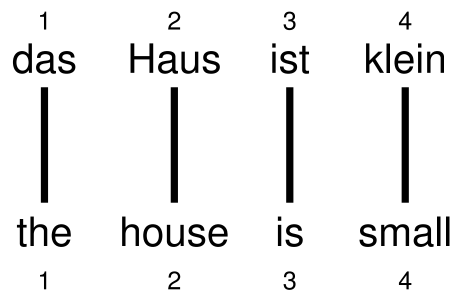
Estimate translation probabilities

- *Maximum likelihood estimation*

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

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

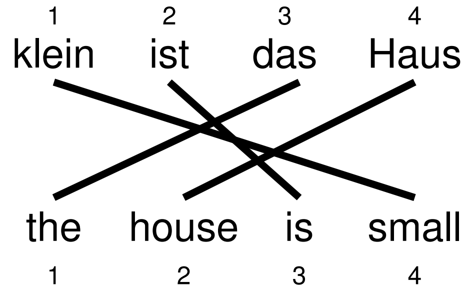
Alignment function

- 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 $a : i \rightarrow j$
- Example

$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4\}$$

Reordering

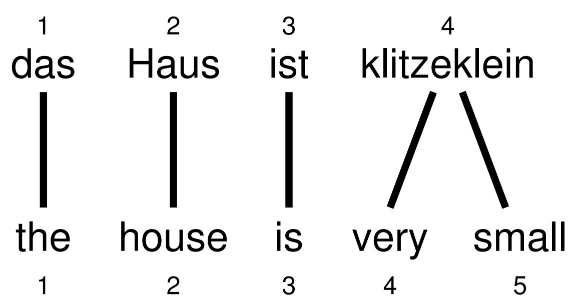
- Words may be **reordered** during translation



$$a : \{1 \rightarrow 3, 2 \rightarrow 4, 3 \rightarrow 2, 4 \rightarrow 1\}$$

One-to-many translation

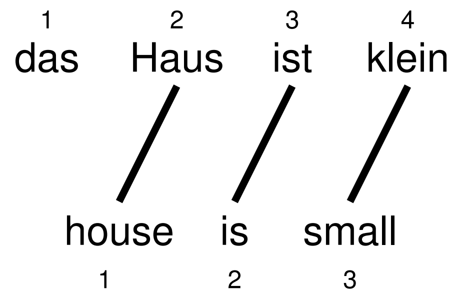
- A source word may translate into **multiple** target words



$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4, 5 \rightarrow 4\}$$

Dropping words

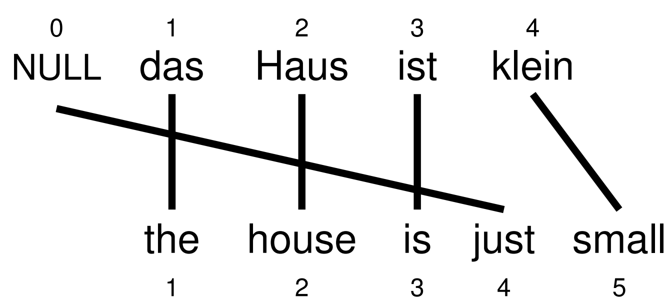
- Words may be **dropped** when translated
  - The German article *das* is dropped



$$a : \{1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4\}$$

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



$$a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 0, 5 \rightarrow 4\}$$

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 **f** $= (f_1, ..., f_{l_f})$ of length l_f
  – to an English sentence **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 $a : j \rightarrow i$

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

  – parameter ϵ is a *normalization constant*

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

$$p(e,a|f) = \frac{\epsilon}{4^3} \times t(\text{the}|\text{das}) \times t(\text{house}|\text{Haus}) \times t(\text{is}|\text{ist}) \times t(\text{small}|\text{klein})$$
$$= \frac{\epsilon}{4^3} \times 0.7 \times 0.8 \times 0.8 \times 0.4$$
$$= 0.0028\epsilon$$

Learning lexical translation models

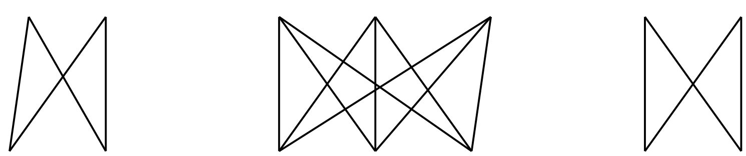
- We would like to *estimate* the lexical translation probabilities $t(e|f)$ from a parallel corpus
- ... but we do not have the alignments
- **Chicken and egg problem**
  - if we had the *alignments*,
    $\rightarrow$ we could estimate the *parameters* of our generative model
  - if we had the *parameters*,
    $\rightarrow$ we could estimate the *alignments*

EM 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
  - initialize model parameters (e.g. uniform)
  - assign probabilities to the missing data
  - estimate model parameters from completed data
  - iterate

EM algorithm

... la maison ... la maison blue ... la fleur ...

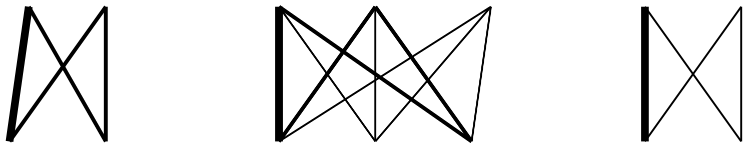


... the house ... the blue house ... the flower ...

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

EM algorithm

... la maison ... la maison blue ... la fleur ...

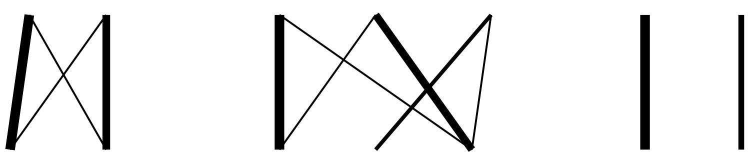


... the house ... the blue house ... the flower ...

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

EM algorithm

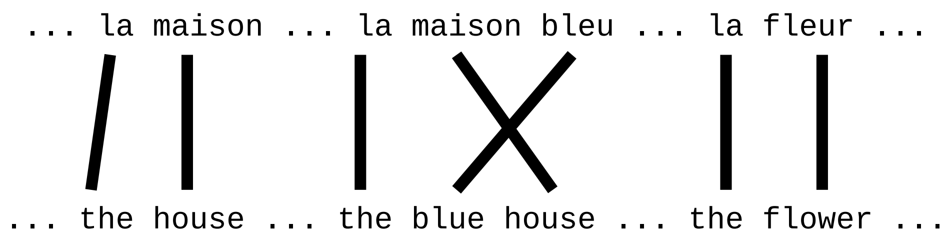
... la maison ... la maison bleu ... la fleur ...



... the house ... the blue house ... the flower ...

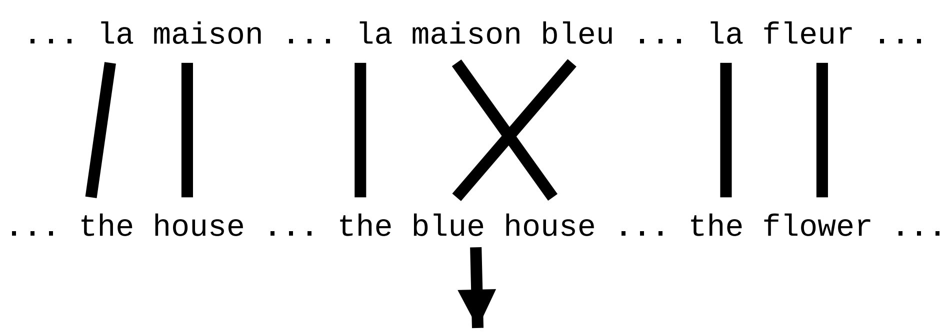
- After another iteration
- It becomes apparent that alignments, e.g., between *fleur* and *flower* are more likely (**pigeon hole principle**)

EM algorithm



- Convergence
- Inherent hidden structure revealed by EM

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

IBM Model 1 and EM

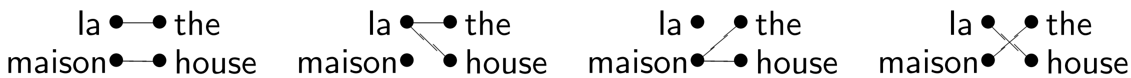
- EM Algorithm consists of two steps
- **Expectation-Step**: Apply model to the data
  - parts of the model are hidden (here: alignments)
  - using the model, assign probabilities to possible values
- **Maximization-Step**: Estimate model from data
  - take assign values as fact
  - collect counts (weighted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

IBM Model 1 and EM

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

IBM Model 1 and EM

- **Probabilities**
  $p(\text{the}|\text{la}) = 0.7$ $\quad p(\text{house}|\text{la}) = 0.05$
  $p(\text{the}|\text{maison}) = 0.1$ $\quad p(\text{house}|\text{maison}) = 0.8$
- **Alignments**



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

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

- **Counts**
  $c(\text{the}|\text{la}) = 0.824 + 0.052$ $\quad c(\text{house}|\text{la}) = 0.052 + 0.007$
  $c(\text{the}|\text{maison}) = 0.118 + 0.007$ $\quad c(\text{house}|\text{maison}) = 0.824 + 0.118$

IBM Model 1 and EM: Expectation Step

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

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

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

IBM Model 1 and EM: Expectation Step

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

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

IBM Model 1 and EM: Expectation Step

$$
\begin{aligned}
p(\mathbf{e}|\mathbf{f}) &= \sum_{a(1)=0}^{l_f} ... \sum_{a(l_e)=0}^{l_f} \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\
&= \frac{\epsilon}{(l_f+1)^{l_e}} \sum_{a(1)=0}^{l_f} ... \sum_{a(l_e)=0}^{l_f} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \\
&= \frac{\epsilon}{(l_f+1)^{l_e}} \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_i)
\end{aligned}
$$

- Note the trick in the last line
  - removes the need for an *exponential* number of products

$\rightarrow$ this makes IBM Model 1 estimation **tractable**

The trick

(case $l_e = l_f = 2$)

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

$$= t(e_1|f_0)\ t(e_2|f_0) + t(e_1|f_0)\ t(e_2|f_1) + t(e_1|f_0)\ t(e_2|f_2) +$$
$$+ t(e_1|f_1)\ t(e_2|f_0) + t(e_1|f_1)\ t(e_2|f_1) + t(e_1|f_1)\ t(e_2|f_2) +$$
$$+ t(e_1|f_2)\ t(e_2|f_0) + t(e_1|f_2)\ t(e_2|f_1) + t(e_1|f_2)\ t(e_2|f_2) =$$
$$= t(e_1|f_0)\,(t(e_2|f_0) + t(e_2|f_1) + t(e_2|f_2)) +$$
$$+ t(e_1|f_1)\,(t(e_2|f_1) + t(e_2|f_1) + t(e_2|f_2)) +$$
$$+ t(e_1|f_2)\,(t(e_2|f_2) + t(e_2|f_1) + t(e_2|f_2)) =$$
$$= (t(e_1|f_0) + t(e_1|f_1) + t(e_1|f_2))\,(t(e_2|f_2) + t(e_2|f_1) + t(e_2|f_2))$$

IBM Model 1 and EM: Expectation Step

- Combine what we have:

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

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

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

IBM Model 1 and EM: Maximization Step

- Now we have to *collect counts*
- Evidence from a sentence pair **e,f** that word e is a translation of word f:

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

- With the same simplication as before:

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

IBM Model 1 and EM: Maximization Step

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

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

IBM Model 1 and EM: Pseudocode

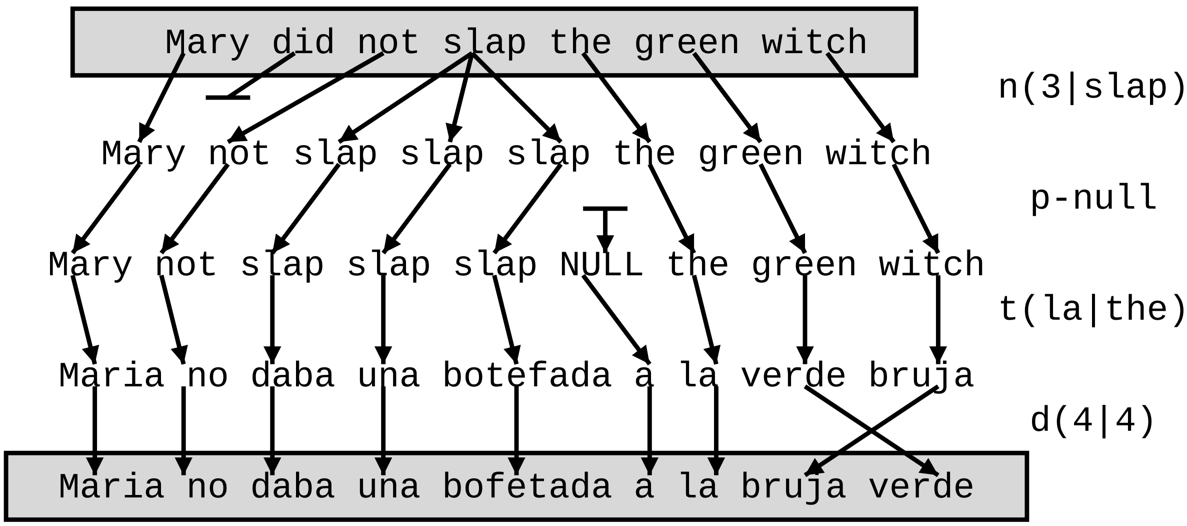
```
initialize t(e|f) uniformly
do until convergence
  set count(e|f) to 0 for all e,f
  set total(f) to 0 for all f
  for all sentence pairs (e_s,f_s)
    for all words e in e_s
      total_s(e) = 0
      for all words f in f_s
        total_s(e) += t(e|f)
    for all words e in e_s
      for all words f in f_s
        count(e|f) += t(e|f) / total_s(e)
        total(f)   += t(e|f) / total_s(e)
  for all f
    for all e
      t(e|f) = count(e|f) / total(f)
```


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**

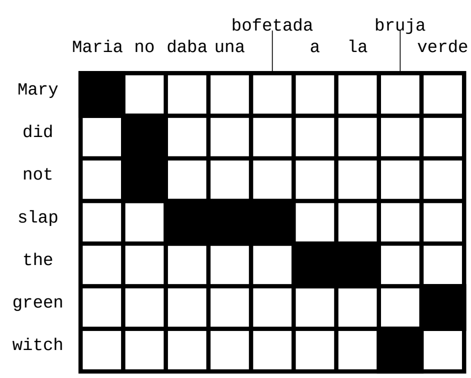
- Only IBM Model 1 has *global maximum*
  – training of a higher IBM model builds on previous model
- Compuationally biggest change in Model 3
  – trick to simplify estimation does not work anymore
  → *exhaustive* count collection becomes computationally too expensive
  – **sampling** over high probability alignments is used instead

IBM Model 4



Word alignment

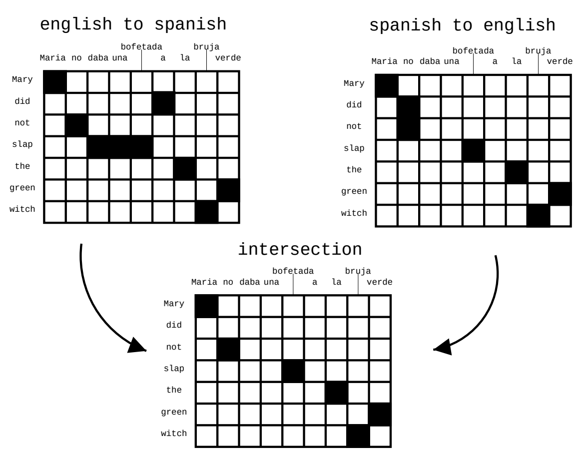
- Notion of **word alignment** valuable
- Shared task at NAACL 2003 and ACL 2005 workshops



Word alignment with IBM models

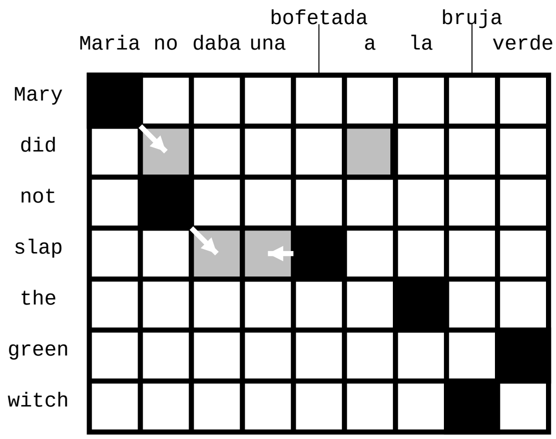
- IBM Models create a *many-to-one* mapping
  - words are aligned using an **alignment function**
  - a function may return the same value for different input (one-to-many mapping)
  - a function can not return multiple values for one input (*no many-to-one* mapping)
- But we need *many-to-many* mappings

Symmetrizing word alignments



- *Intersection* of GIZA++ bidirectional alignments

Symmetrizing word alignments



- *Grow* additional alignment points [Och and Ney, CompLing2003]

Growing heuristic

```
GROW-DIAG-FINAL(e2f,f2e):
  neighboring = ((-1,0),(0,-1),(1,0),(0,1),(-1,-1),(-1,1),(1,-1),(1,1))
  alignment = intersect(e2f,f2e);
  GROW-DIAG(); FINAL(e2f); FINAL(f2e);

GROW-DIAG():
  iterate until no new points added
    for english word e = 0 ... en
      for foreign word f = 0 ... fn
        if ( e aligned with f )
          for each neighboring point ( e-new, f-new ):
            if ( ( e-new not aligned and f-new not aligned ) and
                 ( e-new, f-new ) in union( e2f, f2e ) )
              add alignment point ( e-new, f-new )
FINAL(a):
  for english word e-new = 0 ... en
    for foreign word f-new = 0 ... fn
      if ( ( e-new not aligned or f-new not aligned ) and
           ( e-new, f-new ) in alignment a )
        add alignment point ( e-new, f-new )
```

More Recent Work

- Symmetrization during training
  - symmetrize after each iteration of IBM Models
  - integrate symmetrization into models
- Discriminative training methods
  - supervised learning based on labeled data
  - semi-supervised learning with limited labeled data
- Better generative models
  - see talk by Alexander Fraser

ESSLLI Summer School 2008

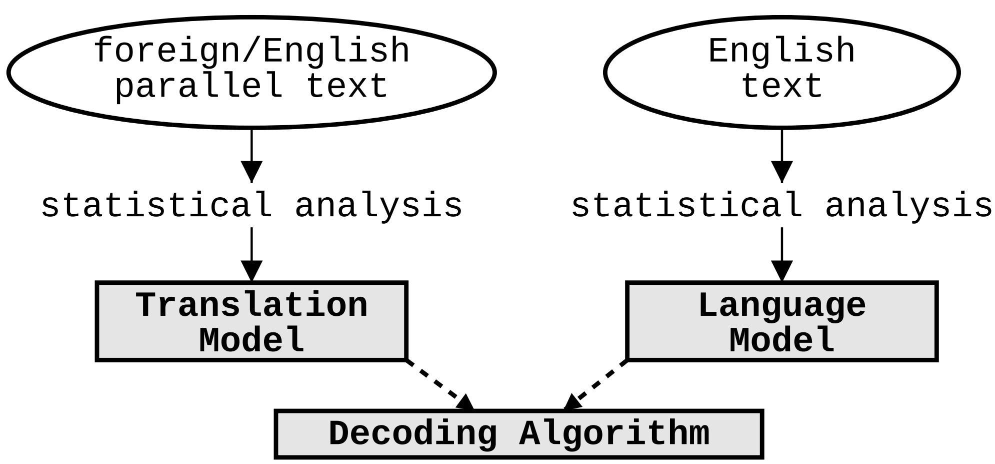
Day 3: Decoding / Phrase-based models

Philipp Koehn, University of Edinburgh

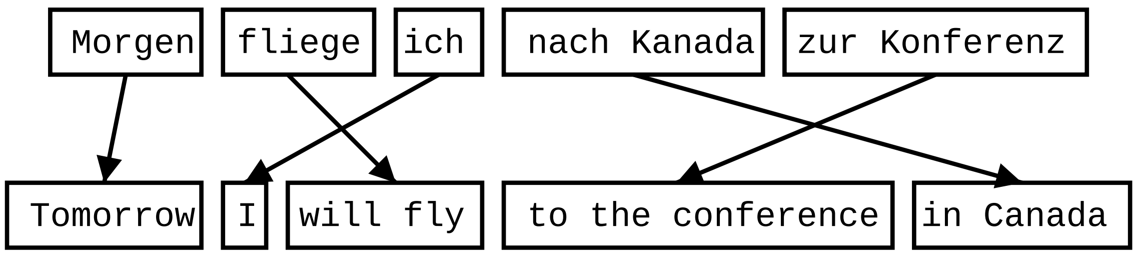
Day 3

Statistical Machine Translation

- Components: Translation model, language model, decoder



Phrase-Based Translation



- Foreign input is segmented in phrases
  - any sequence of words, not necessarily linguistically motivated
- Each phrase is translated into English
- Phrases are reordered

Phrase Translation Table

- Phrase Translations for "den Vorschlag":

English	$\phi(\mathbf{e} \mid \mathbf{f})$	**English**	$\phi(\mathbf{e} \mid \mathbf{f})$
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	...	...

Decoding Process

Maria	no	dio	una	bofetada	a	la	bruja	verde

- Build translation left to right
  - *select foreign* words to be translated

Decoding Process

Maria	no	dio	una	bofetada	a	la	bruja	verde

Mary

- Build translation *left to right*
  - select foreign words to be translated
  - *find English* phrase translation
  - *add English* phrase to end of partial translation

Decoding Process

Maria	no	dio	una	bofetada	a	la	bruja	verde

Mary

- Build translation left to right
  - select foreign words to be translated
  - find English phrase translation
  - add English phrase to end of partial translation
  - *mark foreign* words as translated

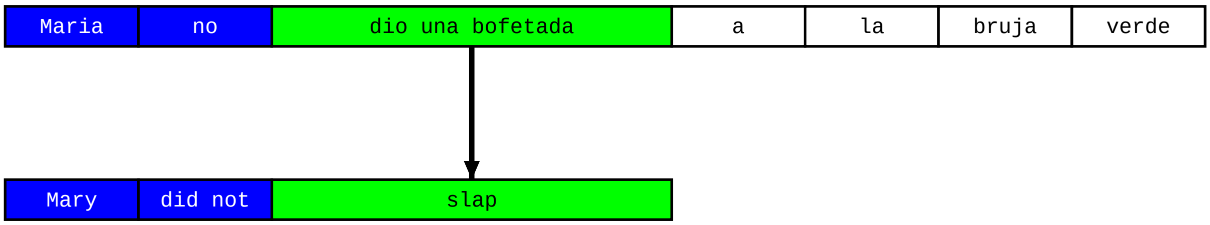
Decoding Process

Maria	no	dio	una	bofetada	a	la	bruja	verde

Mary	did not

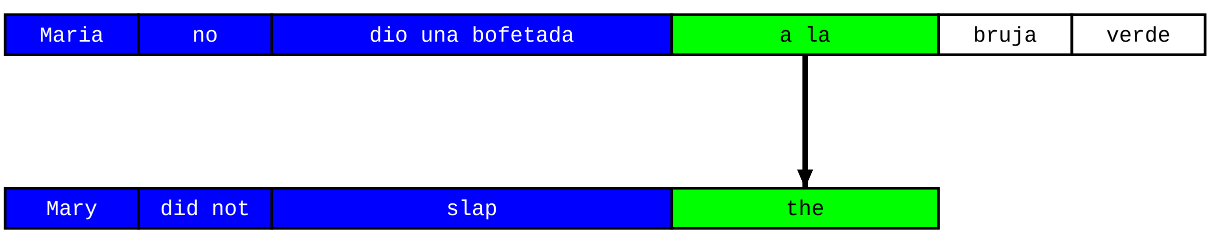
- *One to many* translation

Decoding Process



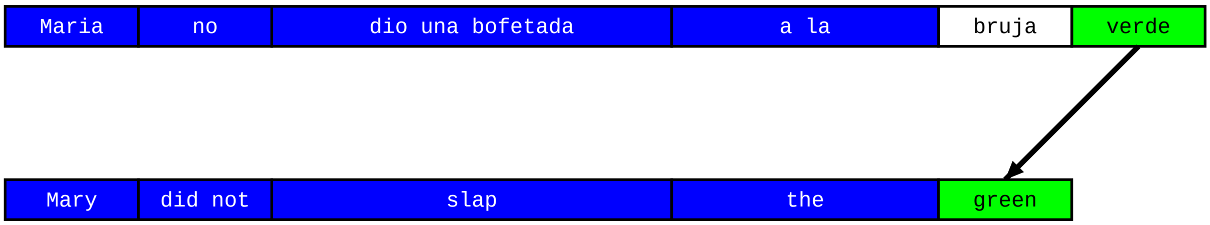
- Many to one translation

Decoding Process



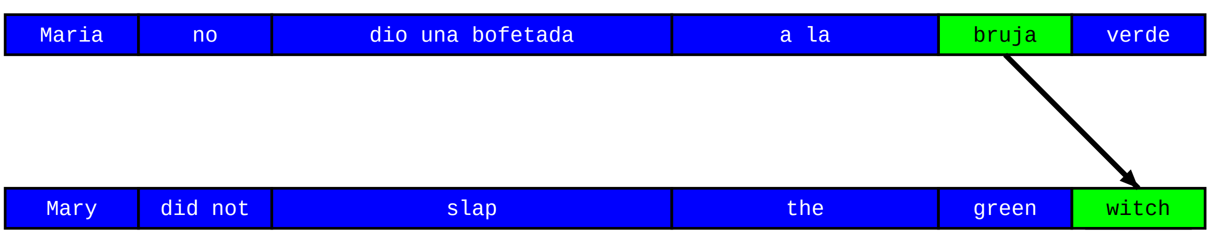
- *Many to one* translation

Decoding Process



- *Reordering*

Decoding Process

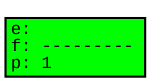


- Translation *finished*

Translation Options

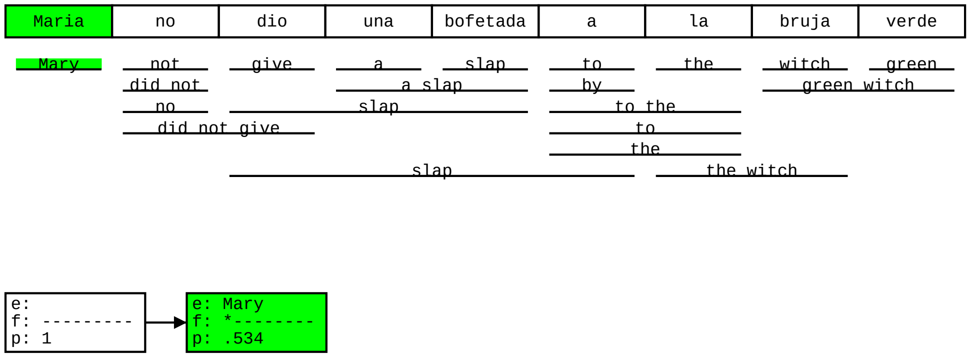
- Look up *possible phrase translations*
  - many different ways to *segment* words into phrases
  - many different ways to *translate* each phrase

Hypothesis Expansion



- Start with **empty hypothesis**
  - e: no English words
  - f: no foreign words covered
  - p: probability 1

Hypothesis Expansion

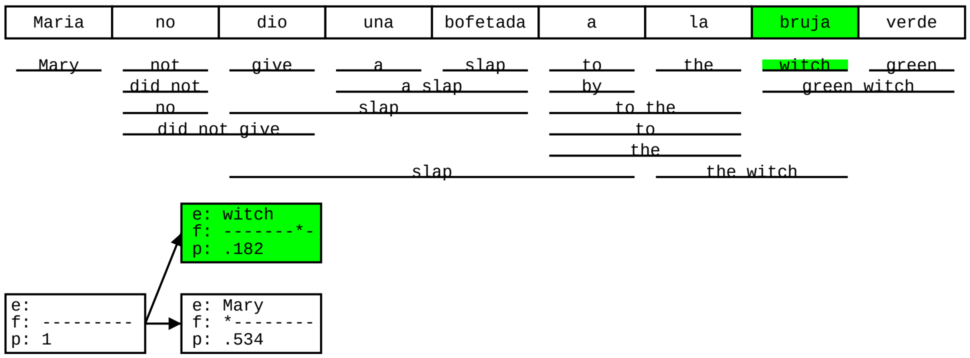


- Pick *translation option*
- Create *hypothesis*
  - e: add English phrase `Mary`
  - f: first foreign word covered
  - p: probability 0.534

A Quick Word on Probabilities

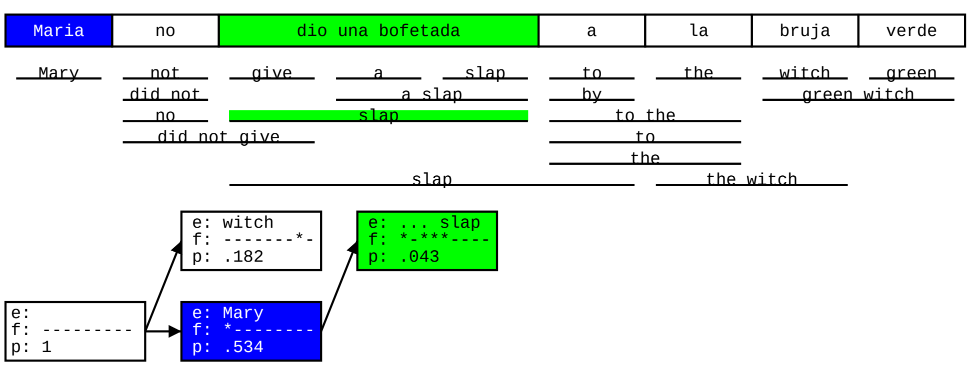
- Not going into detail here, but...
- *Translation Model*
  - phrase translation probability p(Mary|Maria)
  - reordering costs
  - phrase/word count costs
  - ...
- *Language Model*
  - uses trigrams:
  - p(Mary did not) $=$
    p(Mary|START) $\times p$(did|Mary,START) $\times$ p(not|Mary did)

Hypothesis Expansion



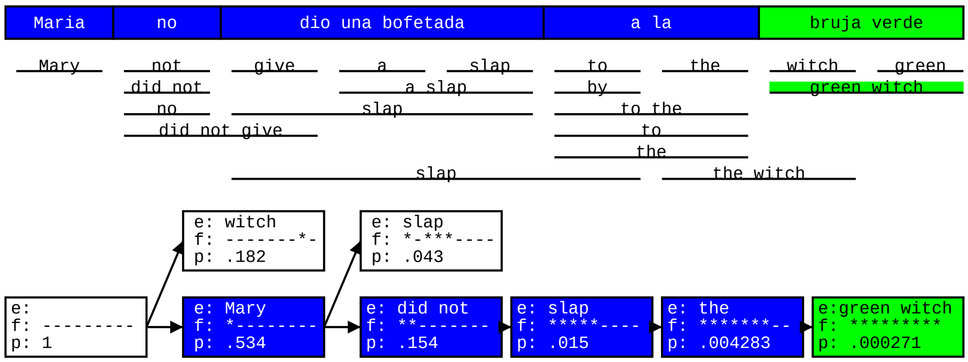
- Add another *hypothesis*

Hypothesis Expansion



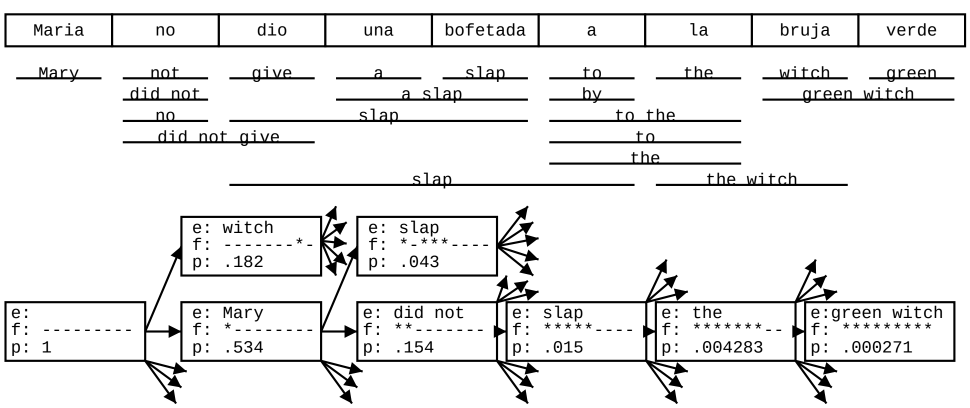
- Further *hypothesis expansion*

Hypothesis Expansion



- ... until all foreign words *covered*
  - find *best hypothesis* that covers all foreign words
  - *backtrack* to read off translation

Hypothesis Expansion



- Adding more hypothesis

⇒ *Explosion* of search space

Explosion of Search Space

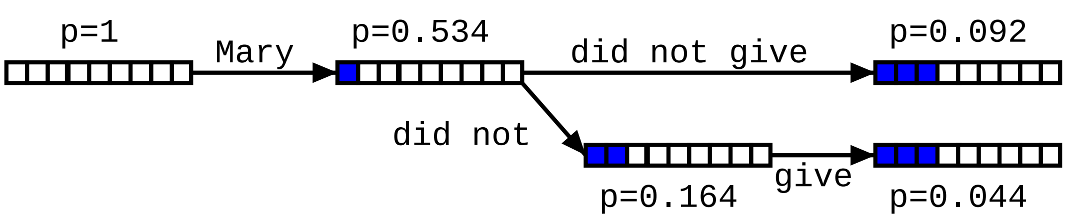
- Number of hypotheses is *exponential* with respect to sentence length

⇒ Decoding is NP-complete [Knight, 1999]

⇒ Need to *reduce search space*

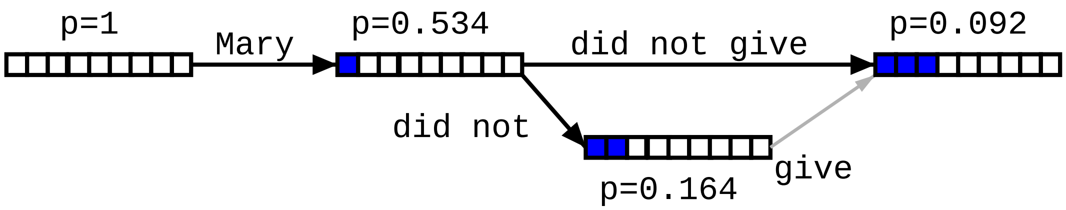
  – risk free: hypothesis **recombination**
  – risky: **histogram/threshold pruning**

Hypothesis Recombination



- Different paths to the *same* partial translation

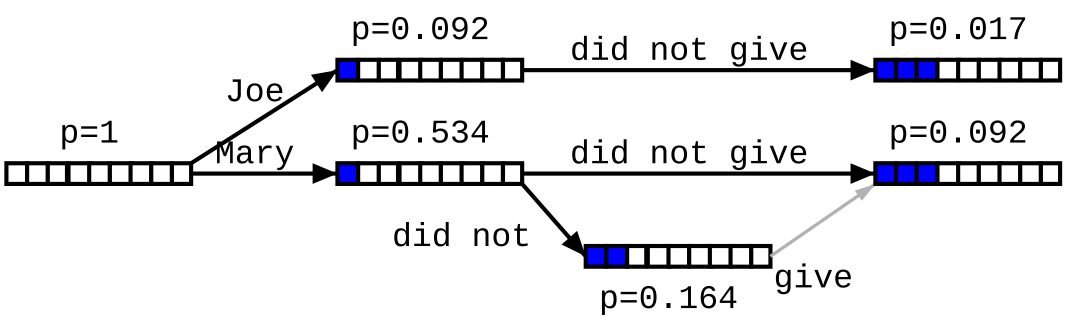
Hypothesis Recombination



- Different paths to the same partial translation

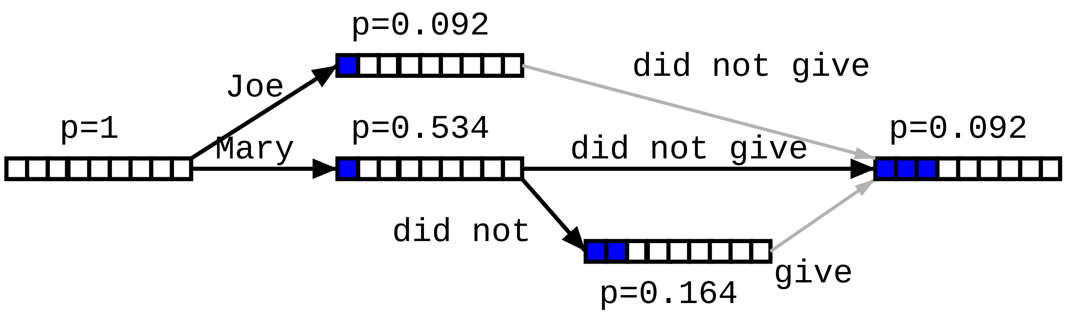
⇒ *Combine paths*
  - *drop weaker* path
  - keep pointer from weaker path (for lattice generation)

Hypothesis Recombination



- Recombined hypotheses do *not* have to *match completely*
- No matter what is added, weaker path can be dropped, if:
  - *last two English words* match (matters for language model)
  - *foreign word coverage* vectors match (effects future path)

Hypothesis Recombination



- Recombined hypotheses do not have to match completely
- No matter what is added, weaker path can be dropped, if:
  - last two English words match (matters for language model)
  - foreign word coverage vectors match (effects future path)

⇒ *Combine paths*

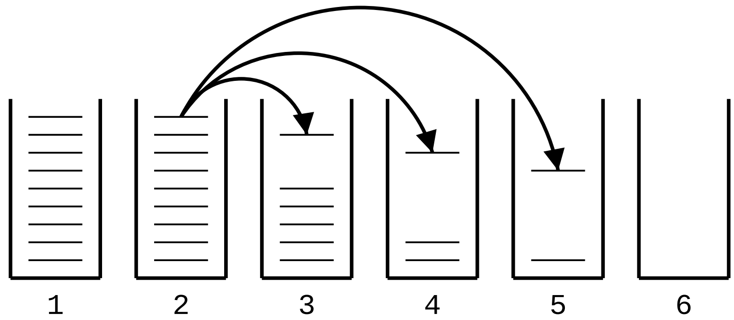
Pruning

- Hypothesis recombination is *not sufficient*

⇒ Heuristically *discard* weak hypotheses early

- Organize Hypothesis in **stacks**, e.g. by
  - *same* foreign words covered
  - *same number* of foreign words covered
- Compare hypotheses in stacks, discard bad ones
  - **histogram pruning**: keep top n hypotheses in each stack (e.g., n=100)
  - **threshold pruning**: keep hypotheses that are at most α times the cost of best hypothesis in stack (e.g., $\alpha = 0.001$)

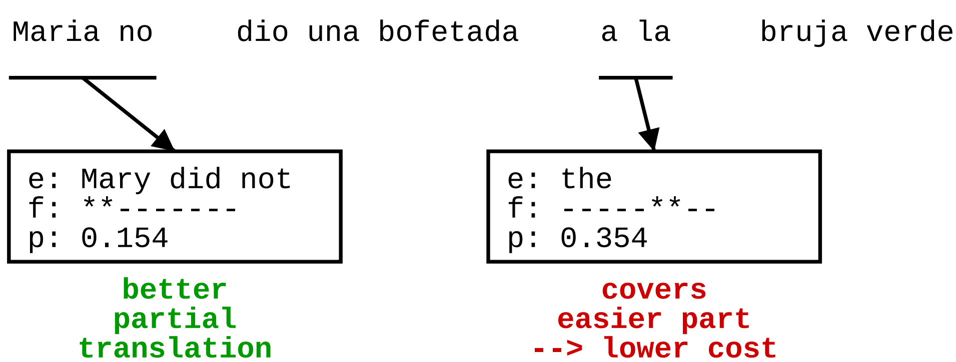
Hypothesis Stacks



- Organization of hypothesis into stacks
  - here: based on *number of foreign words* translated
  - during translation all hypotheses from one stack are expanded
  - expanded Hypotheses are placed into stacks

Comparing Hypotheses

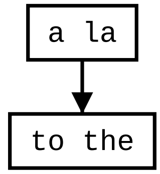
- Comparing hypotheses with *same number of foreign words* covered



- Hypothesis that covers *easy part* of sentence is preferred

⇒ Need to consider **future cost** of uncovered parts

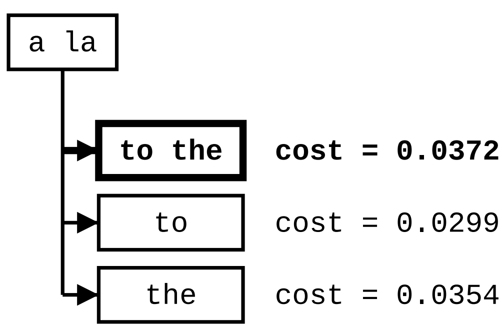
Future Cost Estimation



- *Estimate cost* to translate remaining part of input
- Step 1: estimate future cost for each *translation option*
  - look up translation model cost
  - estimate language model cost (no prior context)
  - ignore reordering model cost

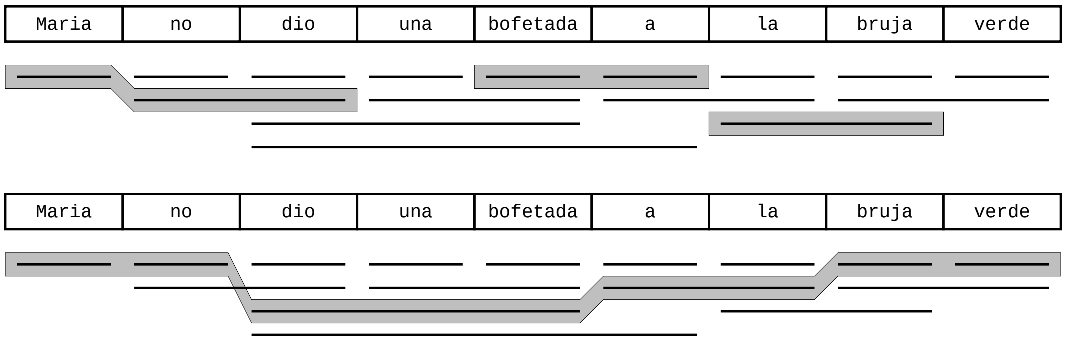
  → LM * TM = p(to) * p(the|to) * p(to the|a la)

Future Cost Estimation: Step 2



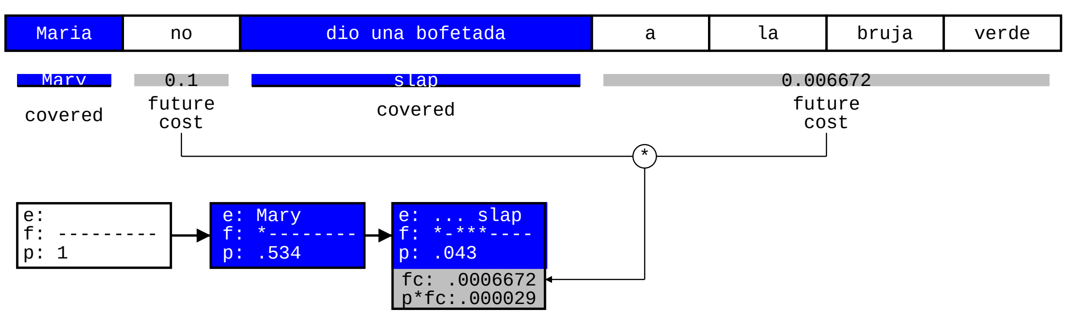
- Step 2: find *cheapest cost* among translation options

Future Cost Estimation: Step 3



- Step 3: find *cheapest future cost path* for each span
  - can be done *efficiently* by dynamic programming
  - future cost for every span can be *pre-computed*

Future Cost Estimation: Application



- Use future cost estimates when *pruning* hypotheses
- For each *uncovered contiguous span*:
  - look up *future costs* for each maximal contiguous uncovered span
  - *add* to actually accumulated cost for translation option for pruning

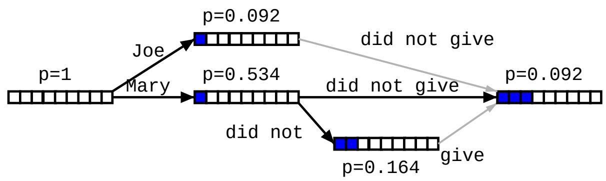
A* search

- Pruning might drop hypothesis that lead to the best path (**search error**)
- **A* search**: safe pruning
  - future cost estimates have to be accurate or underestimates
  - **lower bound** for probability is established early by **depth first search**: compute cost for one complete translation
  - if cost-so-far and future cost are worse than *lower bound*, hypothesis can be safely discarded
- Not commonly done, since not aggressive enough

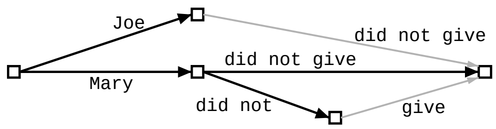
Limits on Reordering

- Reordering may be **limited**
  - **Monotone** Translation: No reordering at all
  - Only phrase movements of at most n words
- Reordering limits *speed* up search (polynomial instead of exponential)
- Current reordering models are weak, so limits *improve* translation quality

Word Lattice Generation



- **Search graph** can be easily converted into a **word lattice**
  - can be further mined for **n-best lists**
  - → enables **reranking** approaches
  - → enables **discriminative training**



Sample N-Best List

- Simple **N-best list**:

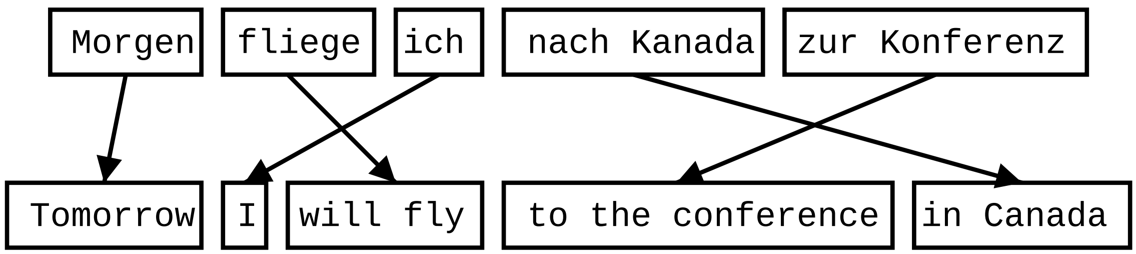
```
Translation ||| Reordering LM TM WordPenalty ||| Score
this is a small house ||| 0 -27.0908 -1.83258 -5 ||| -28.9234
this is a little house ||| 0 -28.1791 -1.83258 -5 ||| -30.0117
it is a small house ||| 0 -27.108 -3.21888 -5 ||| -30.3268
it is a little house ||| 0 -28.1963 -3.21888 -5 ||| -31.4152
this is an small house ||| 0 -31.7294 -1.83258 -5 ||| -33.562
it is an small house ||| 0 -32.3094 -3.21888 -5 ||| -35.5283
this is an little house ||| 0 -33.7639 -1.83258 -5 ||| -35.5965
this is a house small ||| -3 -31.4851 -1.83258 -5 ||| -36.3176
this is a house little ||| -3 -31.5689 -1.83258 -5 ||| -36.4015
it is an little house ||| 0 -34.3439 -3.21888 -5 ||| -37.5628
it is a house small ||| -3 -31.5022 -3.21888 -5 ||| -37.7211
this is an house small ||| -3 -32.8999 -1.83258 -5 ||| -37.7325
it is a house little ||| -3 -31.586 -3.21888 -5 ||| -37.8049
this is an house little ||| -3 -32.9837 -1.83258 -5 ||| -37.8163
the house is a little ||| -7 -28.5107 -2.52573 -5 ||| -38.0364
the is a small house ||| 0 -35.6899 -2.52573 -5 ||| -38.2156
is it a little house ||| -4 -30.3603 -3.91202 -5 ||| -38.2723
the house is a small ||| -7 -28.7683 -2.52573 -5 ||| -38.294
it 's a small house ||| 0 -34.8557 -3.91202 -5 ||| -38.7677
this house is a little ||| -7 -28.0443 -3.91202 -5 ||| -38.9563
it 's a little house ||| 0 -35.1446 -3.91202 -5 ||| -39.0566
this house is a small ||| -7 -28.3018 -3.91202 -5 ||| -39.2139
```

Moses: Open Source Toolkit

- **Open source** statistical machine translation system (developed from scratch 2006)
  - state-of-the-art *phrase-based* approach
  - novel methods: *factored translation models*, *confusion network decoding*
  - support for *very large models* through *memory-efficient* data structures
- Documentation, source code, binaries **available** at `http://www.statmt.org/moses/`
- Development also **supported by**
  - EC-funded *TC-STAR* project
  - *US* funding agencies DARPA, NSF
  - universities (Edinburgh, Maryland, MIT, ITC-irst, RWTH Aachen, ...)

Phrase-based models

Phrase-based translation



- Foreign input is segmented in phrases
  - any sequence of words, not necessarily linguistically motivated
- Each phrase is translated into English
- Phrases are reordered

Phrase-based translation model

- Major components of phrase-based model
  - **phrase translation model** $\phi(\mathbf{f}|\mathbf{e})$
  - **reordering model** $\omega^{d(\text{start}_i - \text{end}_{i-1} - 1)}$
  - **language model** $p_{\text{LM}}(\mathbf{e})$
- Bayes rule

$$\begin{aligned} \text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) &= \text{argmax}_{\mathbf{e}} p(\mathbf{f}|\mathbf{e}) p(\mathbf{e}) \\ &= \text{argmax}_{\mathbf{e}} \phi(\mathbf{f}|\mathbf{e}) \; p_{\text{LM}}(\mathbf{e}) \; \omega^{d(\text{start}_i - \text{end}_{i-1} - 1)} \end{aligned}$$

- Sentence $\mathbf{f}$ is decomposed into I phrases $\bar{f}_1^I = \bar{f}_1, ..., \bar{f}_I$
- Decomposition of $\phi(\mathbf{f}|\mathbf{e})$

$$\phi(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i) \; \omega^{d(\text{start}_i - \text{end}_{i-1} - 1)})$$

Advantages of phrase-based translation

- *Many-to-many* translation can handle non-compositional phrases
- Use of *local context* in translation
- The more data, the *longer phrases* can be learned

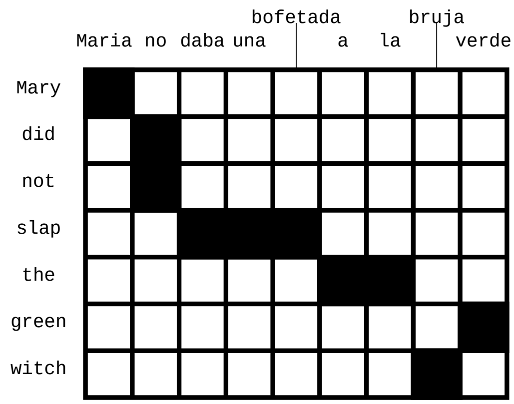
Phrase translation table

- Phrase translations for *den Vorschlag*

English	ϕ(e|f)	English	ϕ(e|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	...	...

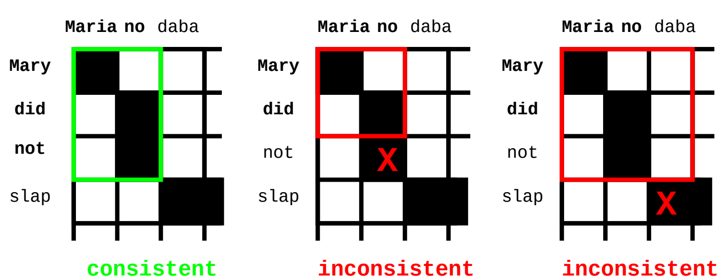
How to learn the phrase translation table?

- Start with the *word alignment*:



- Collect all phrase pairs that are **consistent** with the word alignment

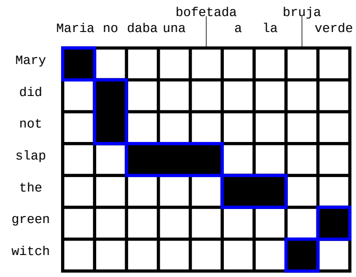
Consistent with word alignment



- **Consistent with the word alignment** :=
  phrase alignment has to *contain all alignment points* for all covered words

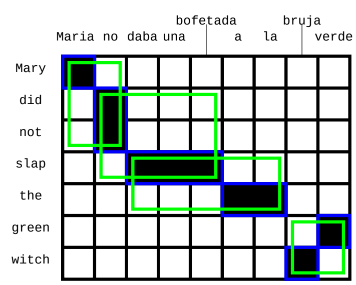
$$(\overline{e}, \overline{f}) \in BP \Leftrightarrow \quad \forall e_i \in \overline{e} : (e_i, f_j) \in A \rightarrow f_j \in \overline{f}$$
$$\text{AND} \quad \forall f_j \in \overline{f} : (e_i, f_j) \in A \rightarrow e_i \in \overline{e}$$

Word alignment induced phrases



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

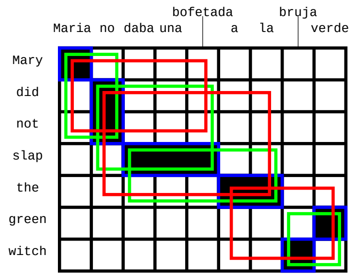
Word alignment induced phrases



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

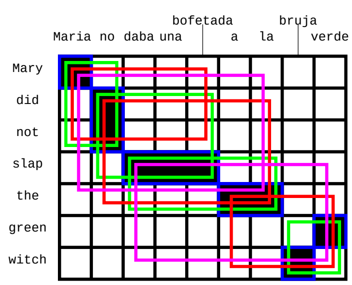
(Maria no, Mary did not), (no daba una bofetada, did not slap), (daba una bofetada a la, slap the), (bruja verde, green witch)

Word alignment induced phrases



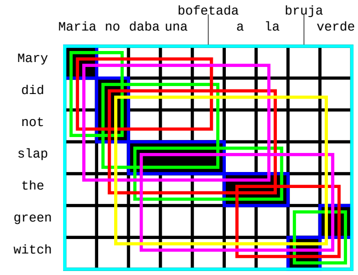
(Maria, Mary), (no, did not), (slap, daba una bofetada), (a la, the), (bruja, witch), (verde, green),
(Maria no, Mary did not), (no daba una bofetada, did not slap), (daba una bofetada a la, slap the),
(bruja verde, green witch), (Maria no daba una bofetada, Mary did not slap),
(no daba una bofetada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

Word alignment induced phrases



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

Word alignment induced phrases (5)



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

Probability distribution of phrase pairs

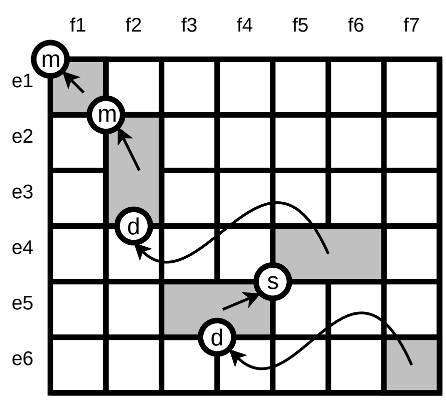
- We need a **probability distribution** $\phi(\overline{f}|\overline{e})$ over the collected phrase pairs

⇒ Possible *choices*
  - *relative frequency* of collected phrases: $\phi(\overline{f}|\overline{e}) = \frac{\text{count}(\overline{f},\overline{e})}{\sum_{\overline{f}} \text{count}(\overline{f},\overline{e})}$
  - or, conversely $\phi(\overline{e}|\overline{f})$
  - use *lexical translation probabilities*

Reordering

- *Monotone* translation
  - do not allow any reordering
  - $\rightarrow$ worse translations
- *Limiting* reordering (to movement over max. number of words) helps
- *Distance-based* reordering cost
  - moving a foreign phrase over n words: cost ω^n
- *Lexicalized* reordering model

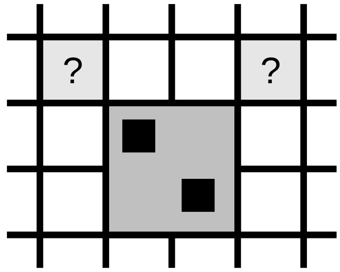
Lexicalized reordering models



[from Koehn et al., 2005, IWSLT]

- Three **orientation** types: **monotone**, **swap**, **discontinuous**
- Probability $p(swap|e, f)$ depends on foreign (and English) *phrase* involved

Learning lexicalized reordering models



[from Koehn et al., 2005, IWSLT]

- Orientation type is *learned during phrase extractions*
- *Alignment point* to the *top left* (monotone) or *top right* (swap)?
- For more, see [Tillmann, 2003] or [Koehn et al., 2005]

Syntax-Based Translation: The Good, The Bad, and How to Win Big

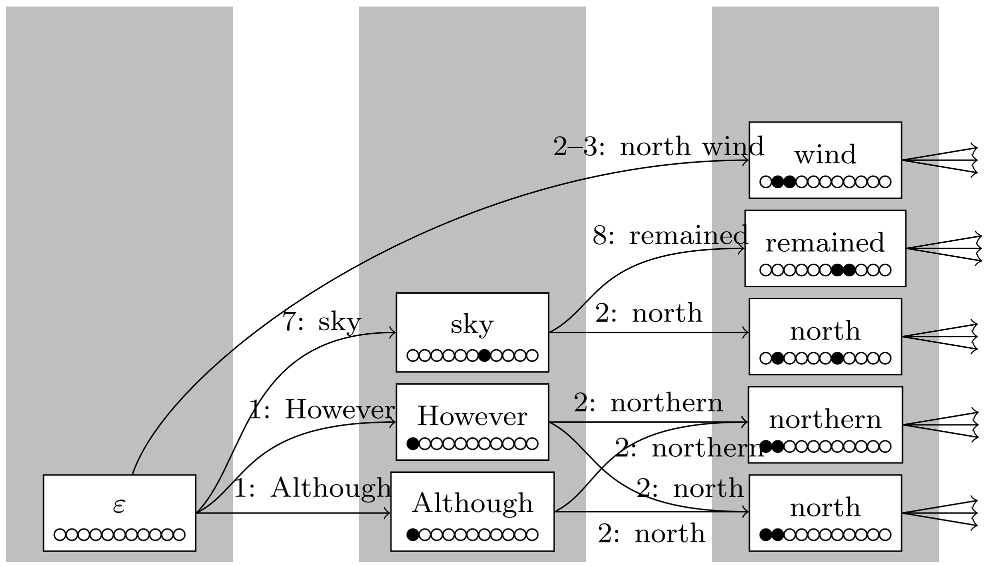
Adam Lopez

with thanks to Ondřej Bojar
(and apologies to Richard P. Gabriel)

- Why do we care about syntax-based MT?
- How does it work?
- What are the open problems?

Disclaimer
Fast-moving field, we only scratch the surface

虽然	北	风	呼啸	,	但	天空	依然	十分	清澈	。
Although	*north*	*wind*	*howls*	*,*	*but*	*sky*	*still*	*extremely*	*limpid*	*.*
1	2	3	4	5	6	7	8	9	10	11



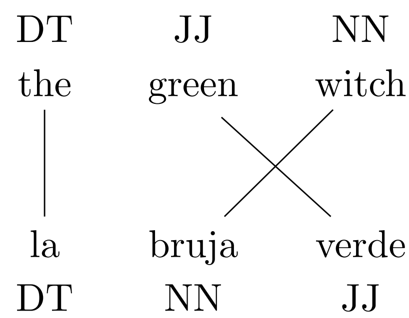
Phrase-based models are good, but not perfect

- computing all possible reorderings is NP-complete
- can't generalize
- can't model long-distance dependencies
- can't model grammaticality

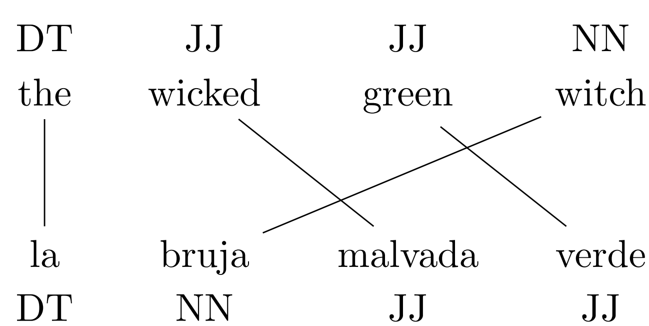
The Good

Syntax-based models aim to solve these problems

- polynomial complexity
- can generalize
- can model long-distance dependencies
- can model grammaticality



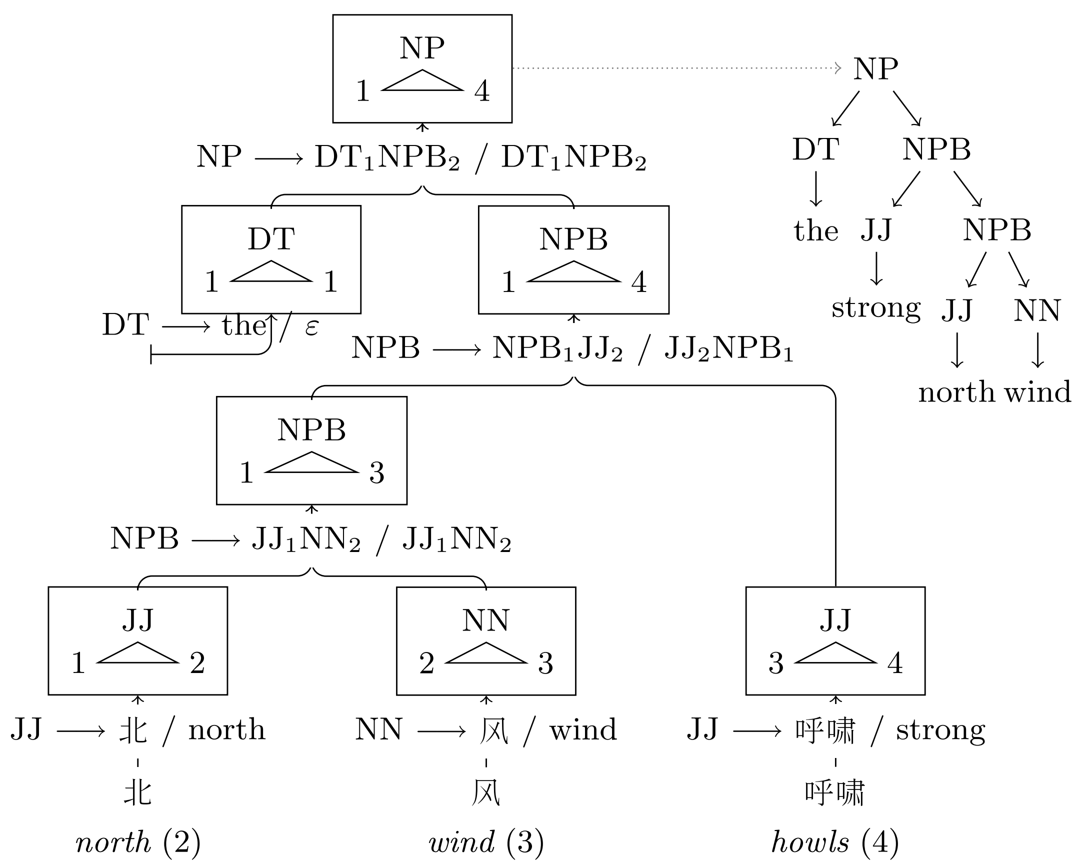
$$\mathrm{NP} \longrightarrow \mathrm{DT}_1\mathrm{JJ}_2\mathrm{NN}_3/\mathrm{DT}_1\mathrm{NN}_3\mathrm{JJ}_1$$



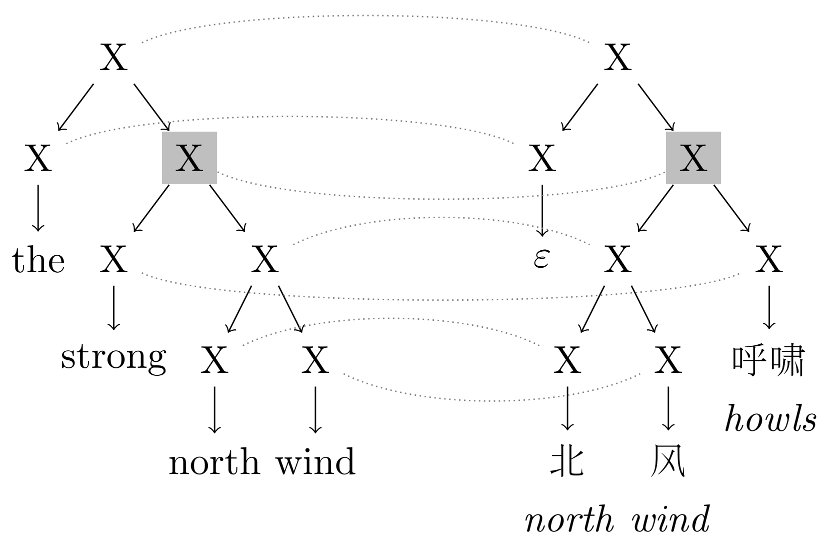
$$\mathrm{NP} \longrightarrow \mathrm{DT}_1\mathrm{JJ}_2\mathrm{JJ}_3\mathrm{NN}_4/\mathrm{DT}_1\mathrm{NN}_4\mathrm{JJ}_2\mathrm{JJ}_3$$

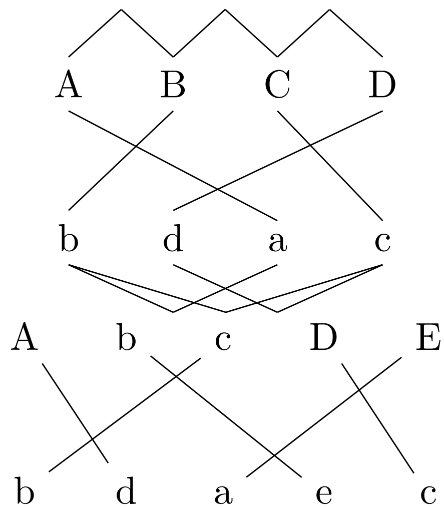
Problem Stack decoding doesn't apply

Idea Decoding is parsing



Problem	Phrase-based decoding with full reordering has exponential complexity.
Idea	Use binary-bracketing SCFG for polynomial complexity.



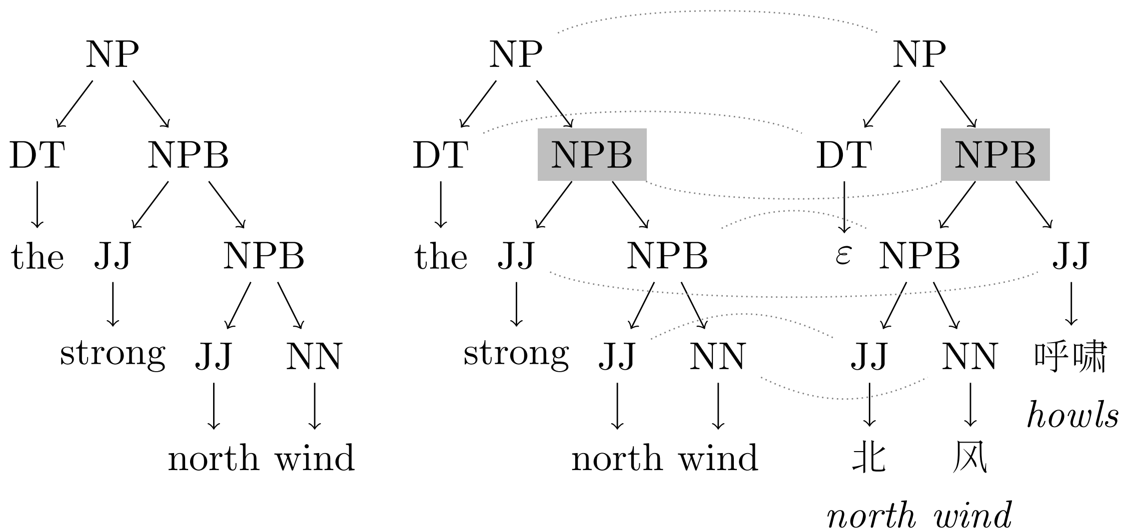


not possible with binary SCFG

not possible with 4-ary SCFG

Problem Phrase-based cannot model grammaticality.

Idea Constrain SCFG to target-side syntax.


NP
DT
NPB
the
JJ
NPB
strong
JJ
NN
north
wind
NP
DT
NPB
the
JJ
NPB
strong
JJ
NN
north
wind
NP
DT
NPB
ε
NPB
JJ
JJ
NN
呼啸
howls
北
风
north wind

The Bad

It doesn't really work.

- Bracketing grammar doesn't capture all alignments.
- Tree isomorphism at production level is too strict.

Where do we go next?

- More theory?
- More articulated models?

> Modeling translational equivalence using wieghted finite state transducers is like approximating a high-order polynomial with line segments... the relatively low expressive power of weighted finite state transducers limits the quality of SMT systems.
>
> –Burbank *et al.* 2005

> But language is hierarchical.
>
> –anonymous MT researcher

> I think phrases are a passing fad.
>
> –anonymous MT researcher

This type of difficulty has happened in other research areas.

See: "Lisp: Good News, Bad News, How to Win Big", presented at the EUROPAL conference by Richard P. Gabriel in 1989.

Lisp = syntax-based models
Unix and C++ = phrase-based models

Simplicity the design must be simple, both in implementation and interface. It is more important for the interface to be simple than the implementation.

Correctness the design must be correct in all observable aspects. Incorrectness is simply not allowed.

Consist … gn is … mplete … portant

The Right Thing

Completeness the design must cover as many important situations as is practical. All reasonably expected cases must be covered. Simplicity is not allowed to overly reduce completeness.

Worse is Better

Simplicity the design must be simple. Simplicity is the most important consideration in a design.

Correctness the design must be correct in all observable aspects. It is slightly better to be simple than correct.

Consistency the design must not be overly inconsistent. It is … at deal … ntroduce …

Completeness the design must cover as many important situations as is practical. Completeness can be sacrificed in favor of any other quality. In fact, completeness must sacrificed whenever implementation simplicity is jeopardized.

The good news is that in 1995 we will have a good operating system and programming language. The bad news is that they will be Unix and C++.

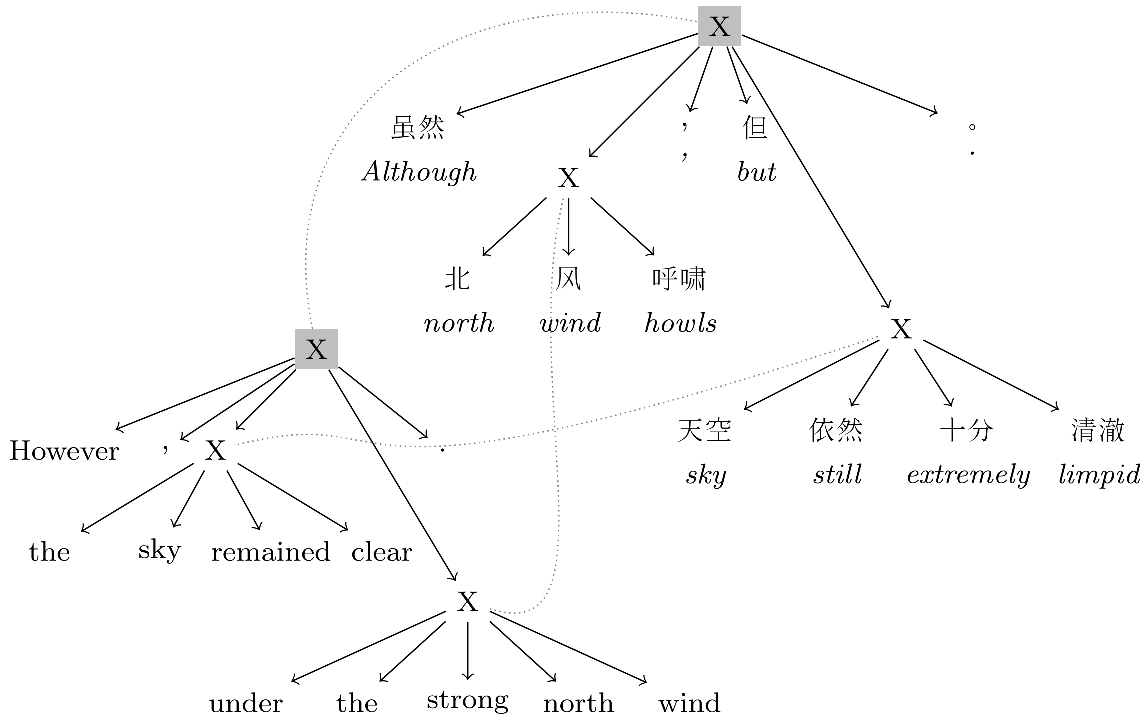
–Richard Gabriel

In 2018, will we have a good translation system based on phrases?

How to Win Big

Observation Phrase-based models good at local reordering.

Idea Use phrases to reorder phrases.



Observation Phrase-based models good, but not grammatical.

Idea Add syntax, but keep the phrases.

Current status

- Syntax-based models competitive with phrase-based
  - Slightly better for Chinese-English
  - Slightly worse for Arabic-English
  - Open question for European languages
  - Language models make a bigger difference
- Not as fast as advertised
  - With 5-gram language model – $O(n^{11})$
  - Easy tricks in phrase-based models not applicable
  - Work on clever search algorithms
- Parsing progress – 1997: 88.1%, 2007: 92.4%

Many, many more angles

- Different formal models with different properties
  - Dependency grammar
  - Synchronous tree substitution grammar
  - Synchronous tree adjoining grammar
- Parsing: source, target, or both?

See handout for some further reading

Additional Notes on Syntax-based Translation

Ondřej Bojar, Adam Lopez

1 Overview

The lecture that accompanies this handout only scratches the surface of a wide and deep field of study. Most researchers in syntax-based translation are motivated to solve one or more problems of phrase-based translation using more expressive models based on various notions of syntax, either formal or linguistic. However, added modeling power comes with added modeling challenges, and meeting these challenges is currently an area of much active research. There are many different approaches. One primary axis of classification of these approaches is the underlying syntactic formalism.

The lecture deals mainly with synchronous context free grammars (constituent trees). These are known in different guises as **syntax-directed translation** (Lewis and Stearns, 1968), **inversion transduction grammar** (Wu, 1995), **head transducers** (Alshawi et al., 2000), and a number of other names. A formalism that generalizes these is **multitext grammar** (Melamed, 2003). Chiang and Knight (2006) provides a good overview of SCFG and several related variants. Lopez (2008) briefly reviews some additional formalisms in the context of a wider survey on statistical machine translation. However, neither of these are complete references. In the remaining sections, we describe some important grammatical formalisms that are useful for European languages, which have application in translation. This text should be viewed as an advanced primer that gives pointers to more complete descriptions found in the literature.

2 Dependency vs. Constituency Trees

Syntactic structure of sentences can be represented using **constituency trees** or **dependency trees**.

Constituency trees indicate recursive "bracketing" of the sentence–sequences of words are grouped together to form constituents:

(1) John (loves Mary)

Dependency trees indicate which words depend on which. Nivre (2005) gives a good review of dependency-based formalisms and dependency parsing.

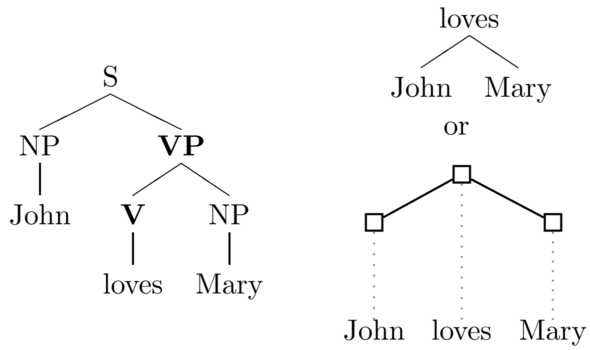


Figure 1: A constituency and a dependency tree. Non-terminals in bold mark heads. Following the trail of heads, we find the terminal node with the same label as the node in a dependency tree would have.

Figure 1 illustrates a constituency tree and a dependency tree. In constituency trees, each **non-terminal node** (labelled in capital letters) represents a constituent. There are no non-terminals in dependency trees. If we choose one of the sons in each constituent to be the **head** of the constituent, e.g. the VP to be the head of the S, we can convert the constituency tree to a dependency tree by "lifting" the terminals up along paths marked with heads.

An **unordered dependency tree** is a connected rooted directed acyclic graph in graph-theoretic sense. An unordered dependency tree does not capture any linear order of words, just pure dependencies. We cannot speak about projectivity (see below) of unordered dependency trees.

An **ordered dependency tree** is an unordered dependency tree with a specified linear order of the nodes. We can thus draw the nodes in the tree from left to right (and the drawing actually means something).

A **constituency tree** can be defined e.g. as a term, using this recursive definition: 1) a terminal is a term, 2) if $t_1, \ldots, t_n$ are terms and N is a non-terminal, then $N(t_1, \ldots, t_n)$ is a term. In the graph-theoretic view, a constituency tree is a tree with linearly ordered sons of each non-terminal.

2.1 Crossing Brackets, Non-Projectivity

Here is a simple example of a sentence with "crossing brackets":

(2) Mary, John loves.

Constituency trees cannot represent structures where a constituent was "moved" outside of its father's span (unless we use empty constituents, sometimes called "traces", i.e. constituents spanning no words, optionally co-indexed with the "moved" words). Because there are no non-terminals in dependency trees to represent the derivation history, some of the "crossing brackets" structures just disappear, see Figure 2.[1]

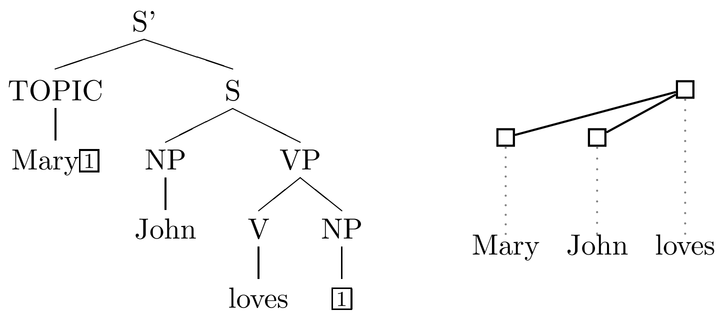


Figure 2: An example of a crossing-bracket yet projective structure.

There are however structures, such as the Dutch "cross-serial" dependencies where, even dependency trees become **non-projective**, i.e. there is a "gap" in the span of a subtree. Representing non-projectivity in dependency trees is easy and natural, see Figure 3.

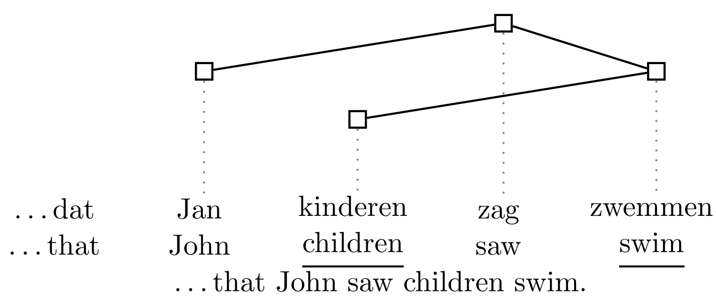


Figure 3: Dutch "cross-serial" dependencies, a non-projective tree with one gap caused by *saw* within the span of *swim*.

Non-projective structures can be relatively rare in English but amount to 23% of sentences in Czech, a Slavic language with relatively free word order (Debusmann and Kuhlmann, 2007).

2.2 Gap Degree and Well-Nestedness

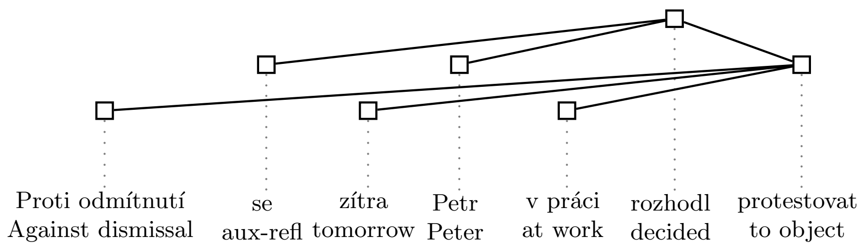
Holan et al. (1998) and Kuhlmann and Möhl (2007) define a measure of non-projectivity: **gap degree** is the number of gaps in a dependency structure. Gap-zero structures are projective structures.

[1] See the difference between a *D*-tree and a *DR*-tree as defined by Holan et al. (1998).

Kuhlmann and Möhl (2007) define another constraint on dependency structures: in **well-nested** structures, disjoint subtrees must not interleave.

Debusmann and Kuhlmann (2007) evaluated that in the Prague Dependency Treebank (Hajič et al., 2006), 99.5% of structures are well-nested and up to gap-1, despite the fact that Czech grammar in principle allows unbounded pumping of gap-degree. The construction is based on two verbs and intermixed modifiers where the dependency relations are disambiguated based on syntactic criteria (e.g. obligatory reflexive particle *se* or subcategorization for a particular preposition or case) and semantic criteria (e.g. verb in past tense cannot accept time modifier referring to future):

(3)



Peter decided to object against the dismissal at work tomorrow.

The non-projective dependencies are *se* and *Peter* depending on the main verb *decided* but appearing within the span of dependents of *to object*: *against dismissal*, *tomorrow*, *at work*. With the main verb itself, there are 3 gaps within the yield of *to object*.

3 Tree Grammars

Tree grammars are one type of finite formal means to define (infinite) sets of trees.

Tree-adjoining grammars (TAG, tag (), see also the review by Joshi et al. (1990)) start from a set of initial trees and use **tree substitution** and **tree adjunction** to derive a tree. The tree substitution operation attaches a treelet to a **frontier** (leaf non-terminal). The tree adjunction splits a tree in a non-terminal and stitches a treelet in between, see Figure 4. **Tree-substitution grammars** (TSG, Eisner (2003) or e.g. Bojar and Čmejrek (2007)) are like TAG but allow only tree substitution, no tree adjunction.

Figure 5 illustrates how a sentence is analyzed using a constituency-based TSG and a dependency-based TSG. The difference between constituency- and dependency-based TSG is the type of underlying trees. Non-terminal nodes in a dependency-based TSG can appear as leaves of unfinished trees only and have to be substituted by a tree later in the derivation.

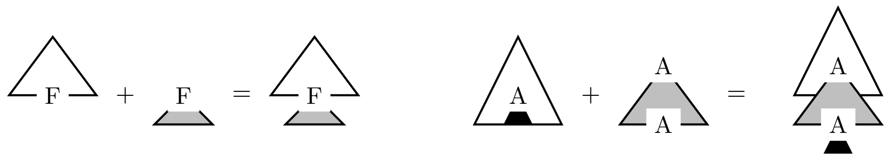


Figure 4: Tree substitution at frontier F and tree adjunction at internal node A.

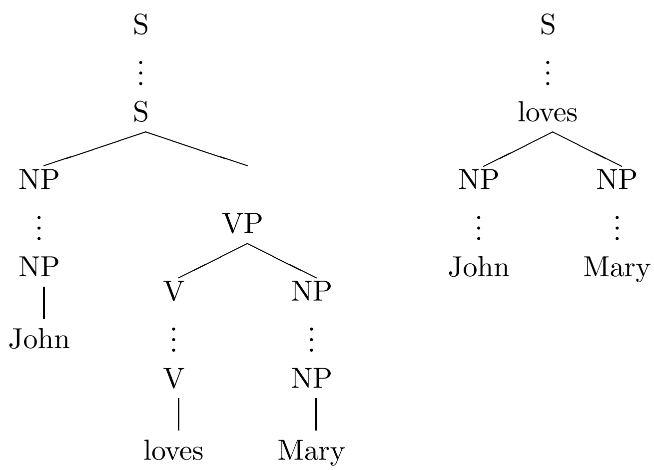


Figure 5: Derivation of a sentence using constituency-based and dependency-based tree substitutions. The substitution is indicated by "⋮".

3.1 Constituency vs. Dependency Tree Adjunction

TAG defines the adjunction operation for constituency trees only. The same definition cannot be casted to dependency-based TSG (dep-TSG) because there are no internal non-terminals to adjoin at. However, we can still think of the "linguistic adjunction" in dep-TSG. This operation adds adjuncts to a node. In terms of TSG, a little tree gets attached to an internal node instead at a frontier. dep-TSG adjunction thus allows to add siblings to an already existing node.

The trouble starts if we consider ordered dependency trees. Where is the new dependent placed with respect of the existing dependents? And is the newly attached subtree attached projectively, or can older nodes in the tree introduce gaps into it? (And where the gaps are allowed to be?) E.g. Quirk et al. (2005) use a probabilistic model to interleave old dependents and newly adjoint dependents but do not seem allow non-projective attachments.

3.2 Remarks on Generative Capacity

This is by no means a complete survey.

Gaifman (1965) shows that *projective* dependency structures are weakly equivalent to CFG. We have already illustrated how marking of heads is used to convert a constituency tree to a dependency tree in Figure 1.

Joshi et al. (1990) describe various formalisms for so-called **mildly context sensitive** (MCS) grammars. The term MCS refers to various grammars beyond CFG but still parsable in polynomial time. TAG is one of them and was motivated by the need to represent Dutch cross-serial dependencies (Figure 3). Naturally, TAG needs traces in its constituency trees.

Kuhlmann and Möhl (2007) shows that lexicalized TAG (LTAG) is equivalent to well-nested dependency structures with at most one gap. kuhlmann-mohl:2007:ACLMain (also define an infinite hierarchy of mildly context-sensitive dependency structures (i.e. parsable in polynomial time) of ever growing weak generative power.

Plátek (2001) defines a special type of formal automata to define a hierarchy of languages beyond CFG. Jurdziński et al. (2008) shows that already the class of languages accepted by a quite restricted from of the automaton contains NP-complete languages and is thus not much useful for efficient parsing.

3.3 Translation Direction

When designing an MT system, one should consider the properties of the source and target languages.

For instance, when translating from Czech to English, source-side non-projectivities have to be accounted for. Alternatively, a non-projective dependency parser such as (McDonald et al., 2005) can be used and the resulting dependency tree can be tranfered to the target language using e.g. STSG.

When translating from English to Czech, significant portion of non-projective structures can be disregarded because there exists a grammatically correct reordering that reduces the gap degree. For instance, the sentence in Example 3 could be translated from the English gloss as *Petr se rozhodl proti odmítnutí zítra v práci protestovat.*, rendering no gap at all. However, the position of the reflexive particle *se* is fairly rigid (the "second" position in the sentence) and constraints on topic-focus articulation often lead to a gap-1 structure. Forcing projective word order by e.g. CFG as Galley et al. (2006) do on the target side would lead to mildly disfluent output.

4 References

Hiyan Alshawi, Srinivas Bangalore, and Shona Douglas. 2000. Learning dependency translation models as collections of finite state head transducers. *Computational Linguistics*, 26(1):45–60, Mar.

Ondřej Bojar and Martin Čmejrek. 2007. Mathematical Model of Tree Transformations. Project Euromatrix - Deliverable 3.2, ÚFAL, Charles University, December.

David Chiang and Kevin Knight. 2006. An Introduction to Synchronous Grammars. Part of a tutorial given at ACL 2006, `http://www.isi.edu/˜chiang/papers/synchtut.pdf`.

Ralph Debusmann and Marco Kuhlmann. 2007. Dependency grammar: Classification and exploration. Project report (CHORUS, SFB 378).

Jason Eisner. 2003. Learning Non-Isomorphic Tree Mappings for Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL), Companion Volume*, pages 205–208, Sapporo, July.

Haim Gaifman. 1965. Dependency Systems and Phrase-Structure Systems. *Information and Control*, 8(3):304–337.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 961–968. Association for Computational Linguistics.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková Razímová. 2006. Prague Dependency Treebank 2.0. LDC2006T01, ISBN: 1-58563-370-4.

T. Holan, V. Kuboň, K. Oliva, and M. Plátek. 1998. Two Useful Measures of Word Order Complexity. In A. Polguere and S. Kahane, editors, *Proceedings of the Coling '98 Workshop: Processing of Dependency-Based Grammars*, Montreal. University of Montreal.

Aravind K. Joshi, K. Vijay Shanker, and David Weir. 1990. The Convergence of Mildly Context-Sensitive Grammar Formalisms. Technical Report MS-CIS-90-01, University of Pennsylvania Department of Computer and Information Science.

Tomasz Jurdziński, Friedrich Otto, František Mráz, and Martin Plátek. 2008. On the complexity of 2-monotone restarting automata. *Theor. Comp. Sys.*, 42(4):488–518.

Marco Kuhlmann and Mathias Möhl. 2007. Mildly context-sensitive dependency languages. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 160–167, Prague, Czech

Republic, June. Association for Computational Linguistics.

P. M. II Lewis and R. E. Stearns. 1968. Syntax-directed transductions. *Journal of the ACM*, 15:465–488.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys*, 40(3), Sep. In press. Preprint draft available at http://homepages.inf.ed.ac.uk/alopez/pdf/survey.pdf

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-Projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of HLT/EMNLP 2005*, October.

I. Dan Melamed. 2003. Multitext grammars and synchronous parsers. In *Proc. of HLT-NAACL*, pages 79–86, May.

Joakim Nivre. 2005. Dependency Grammar and Dependency Parsing. Technical Report MSI report 05133, Växjö University: School of Mathematics and Systems Engineering.

Martin Plátek. 2001. Two-way restarting automata and j-monotonicity. In *SOFSEM '01: Proceedings of the 28th Conference on Current Trends in Theory and Practice of Informatics Piestany*, pages 316–325, London, UK. Springer-Verlag.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 271–279. Association for Computational Linguistics.

Dekai Wu. 1995. Stochastic inversion transduction grammars, with application to segmentation, bracketing, and alignment of parallel corpora. In *Proc. of IJCAI*, pages 1328–1335, Aug.

ESSLLI Summer School 2008

Day 5: Factored Translation Models and Discriminative Training

Philipp Koehn, University of Edinburgh

Day 5

Factored Translation Models

- **Motivation**
- Example
- Model and Training
- Decoding
- Experiments
- Planned Work

Statistical machine translation today

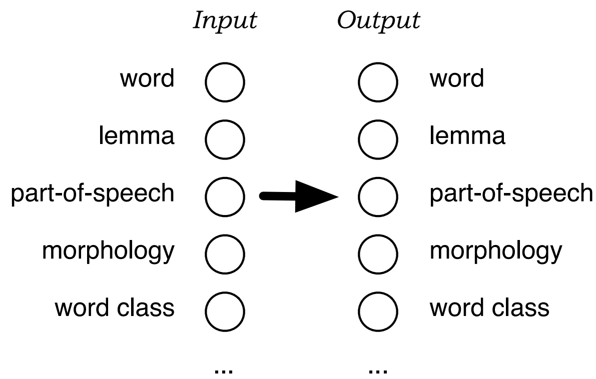
- Best performing methods based on **phrases**
  - short sequences of words
  - no use of explicit syntactic information
  - no use of morphological information
  - currently best performing method
- Progress in **syntax-based** translation
  - tree transfer models using syntactic annotation
  - still shallow representation of words and non-terminals
  - active research, improving performance

One motivation: morphology

- Models treat *car* and *cars* as completely different words
  - training occurrences of *car* have no effect on learning translation of *cars*
  - if we only see *car*, we do not know how to translate *cars*
  - rich morphology (German, Arabic, Finnish, Czech, ...) → many word forms
- Better approach
  - analyze surface word forms into **lemma** and **morphology**, e.g.: *car +plural*
  - translate lemma and morphology separately
  - generate target surface form

Factored translation models

- **Factored represention** of words



- Goals
  - **Generalization**, e.g. by translating lemmas, not surface forms
  - **Richer model**, e.g. using syntax for reordering, language modeling)

Related work

- **Back off** to representations with richer statistics (lemma, etc.)
[Nießen and Ney, 2001, Yang and Kirchhoff 2006, Talbot and Osborne 2006]
- Use of additional annotation in **pre-processing** (POS, syntax trees, etc.)
[Collins et al., 2005, Crego et al, 2006]
- Use of additional annotation in **re-ranking** (morphological features, POS, syntax trees, etc.)
[Och et al. 2004, Koehn and Knight, 2005]

→ we pursue an *integrated approach*

- Use of syntactic **tree structure**
[Wu 1997, Alshawi et al. 1998, Yamada and Knight 2001, Melamed 2004, Menezes and Quirk 2005, Chiang 2005, Galley et al. 2006]

→ may be *combined* with our approach

Factored Translation Models

- Motivation
- **Example**
- Model and Training
- Decoding
- Experiments
- Planned Work

Decomposing translation: example

- **Translate** lemma and syntactic information **separately**

$$\begin{bmatrix}\text{lemma}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{lemma}\end{bmatrix}$$

$$\begin{bmatrix}\text{part-of-speech}\\ \text{morphology}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{part-of-speech}\\ \text{morphology}\end{bmatrix}$$

Decomposing translation: example

- **Generate surface** form on target side

$$\begin{bmatrix} \text{surface} \end{bmatrix}$$
$$\Uparrow$$
$$\begin{bmatrix} \text{lemma} \\ \text{part-of-speech} \\ \text{morphology} \end{bmatrix}$$

Translation process: example

Input: *(Autos, Auto, NNS)*

1. Translation step: lemma ⇒ lemma
   (?, car, ?), (?, auto, ?)
2. Generation step: lemma ⇒ part-of-speech
   (?, car, NN), (?, car, NNS), (?, auto, NN), (?, auto, NNS)
3. Translation step: part-of-speech ⇒ part-of-speech
   (?, car, NN), (?, car, NNS), (?, auto, NNP), (?, auto, NNS)
4. Generation step: lemma,part-of-speech ⇒ surface
   (car, car, NN), (cars, car, NNS), (auto, auto, NN), (autos, auto, NNS)

Factored Translation Models

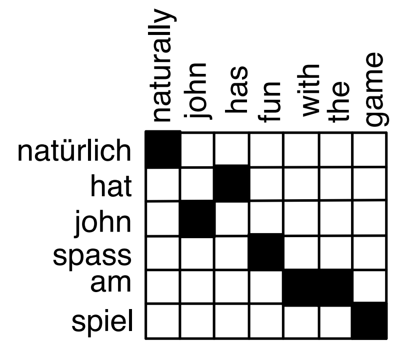
- Motivation
- Example
- **Model and Training**
- Decoding
- Experiments
- Planned Work

Model

- Extension of *phrase model*
- Mapping of foreign words into English words broken up into steps
  - **translation step**: maps foreign factors into English factors (on the phrasal level)
  - **generation step**: maps English factors into English factors (for each word)
- Each step is modeled by one or more *feature functions*
  - fits nicely into log-linear model
  - weight set by discriminative training method
- Order of mapping steps is chosen to optimize search

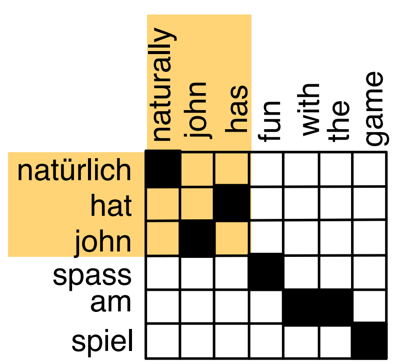
Phrase-based training

- Establish word alignment (GIZA++ and symmetrization)



Phrase-based training

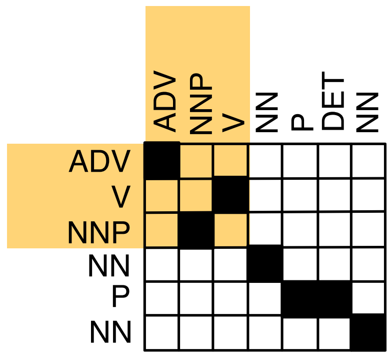
- Extract phrase



$\Rightarrow$ *natürlich hat john* — *naturally john has*

Factored training

- Annotate training with factors, extract phrase



⇒ *ADV V NNP* — *ADV NNP V*

Training of generation steps

- Generation steps map target factors to target factors
  - typically trained on target side of parallel corpus
  - may be trained on additional monolingual data
- Example: *The*/DET *man*/NN *sleeps*/VBZ
  - count collection
    - count(*the*,DET)++
    - count(*man*,NN)++
    - count(*sleeps*,VBZ)++
  - evidence for probability distributions (max. likelihood estimation)
    - p(DET|*the*), p(*the*|DET)
    - p(NN|*man*), p(*man*|NN)
    - p(VBZ|*sleeps*), p(*sleeps*|VBZ)

Factored Translation Models

- Motivation
- Example
- Model and Training
- **Decoding**
- Experiments
- Planned Work

Phrase-based translation

- Task: *translate this sentence* from German into English

er **geht** **ja** **nicht** **nach** **hause**

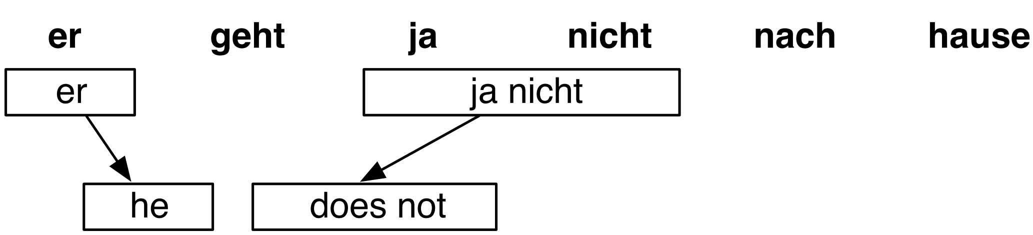
Translation step 1

- Task: translate this sentence from German into English

- *Pick* phrase in input, *translate*

Translation step 2

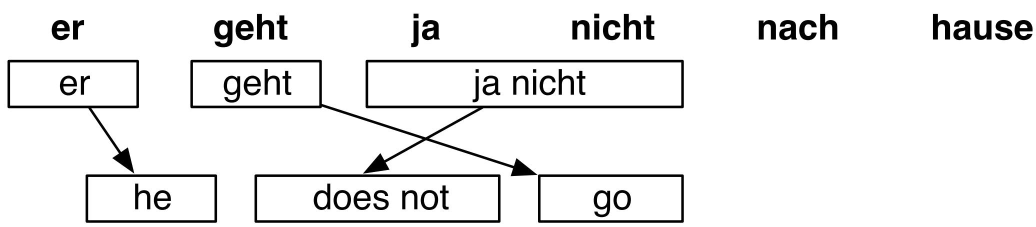
- Task: translate this sentence from German into English



- Pick phrase in input, translate
  - it is allowed to pick words *out of sequence* (**reordering**)
  - phrases may have multiple words: *many-to-many* translation

Translation step 3

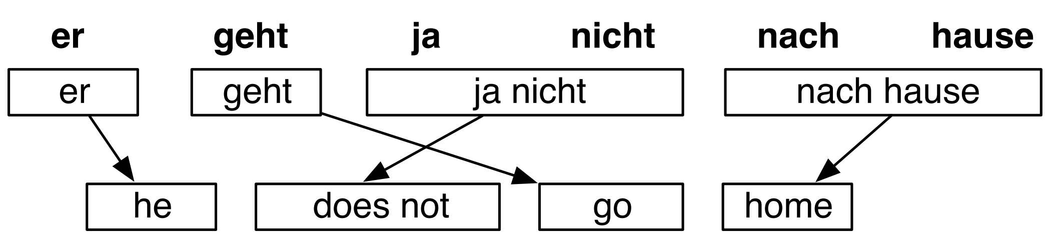
- Task: translate this sentence from German into English



- Pick phrase in input, translate

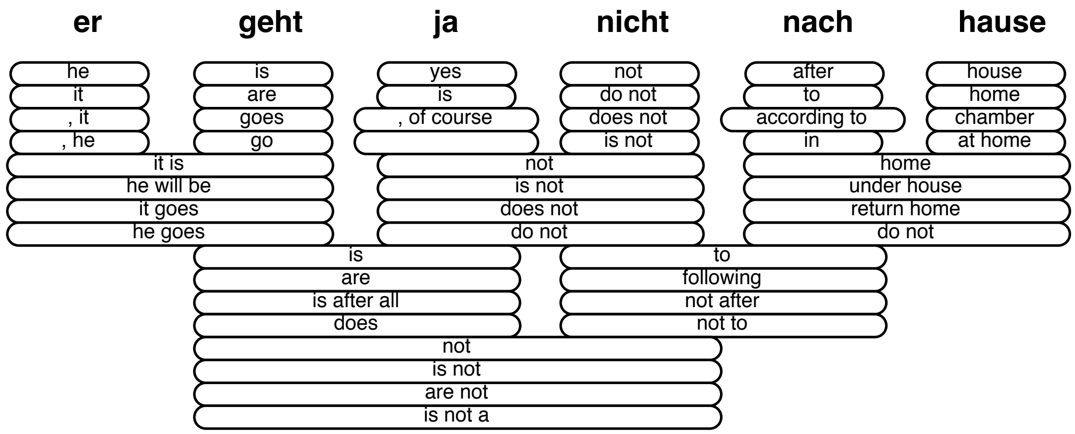
Translation step 4

- Task: translate this sentence from German into English



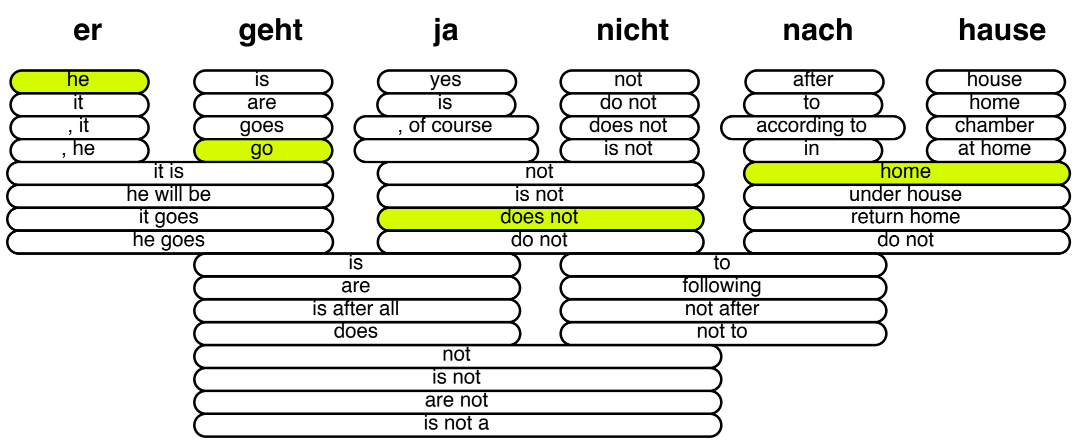
- Pick phrase in input, translate

Translation options



- *Many translation options* to choose from
  - in Europarl phrase table: *2727 matching phrase pairs* for this sentence
  - by pruning to the top 20 per phrase, *202 translation options* remain

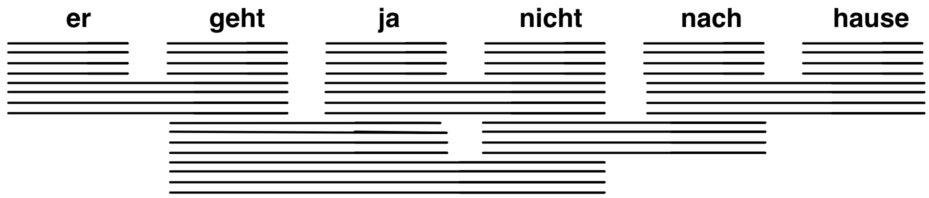
Translation options



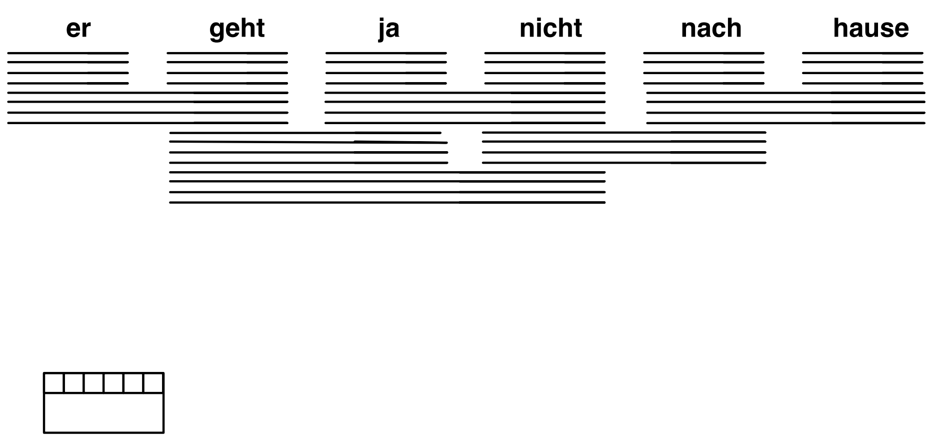
- The machine translation decoder does not know the right answer

→ *Search problem* solved by heuristic beam search

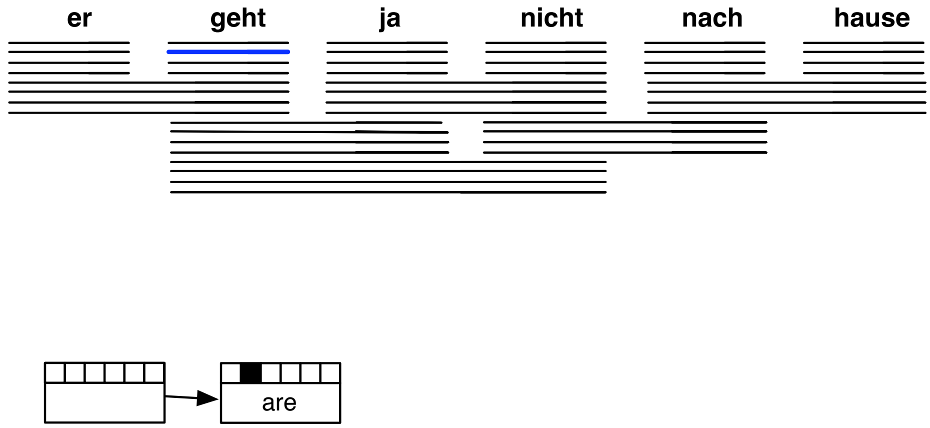
Decoding process: precompute translation options



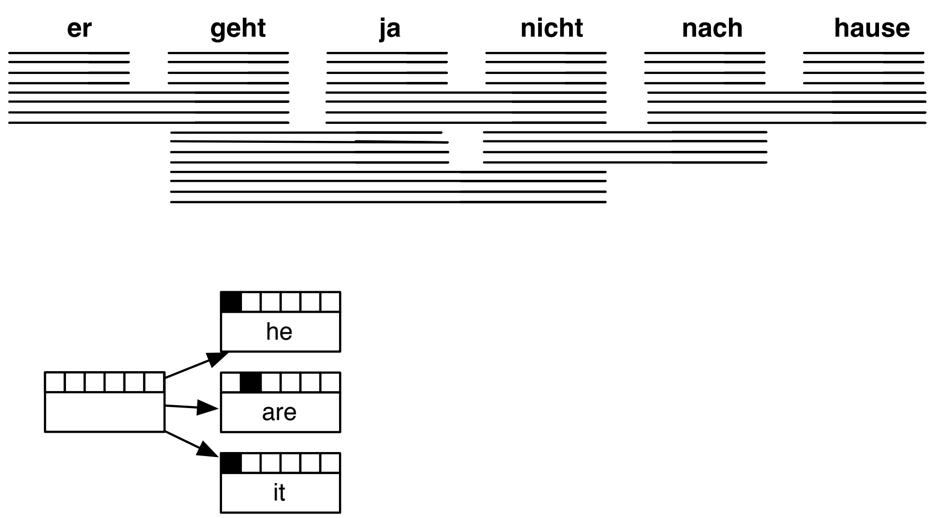
Decoding process: start with initial hypothesis



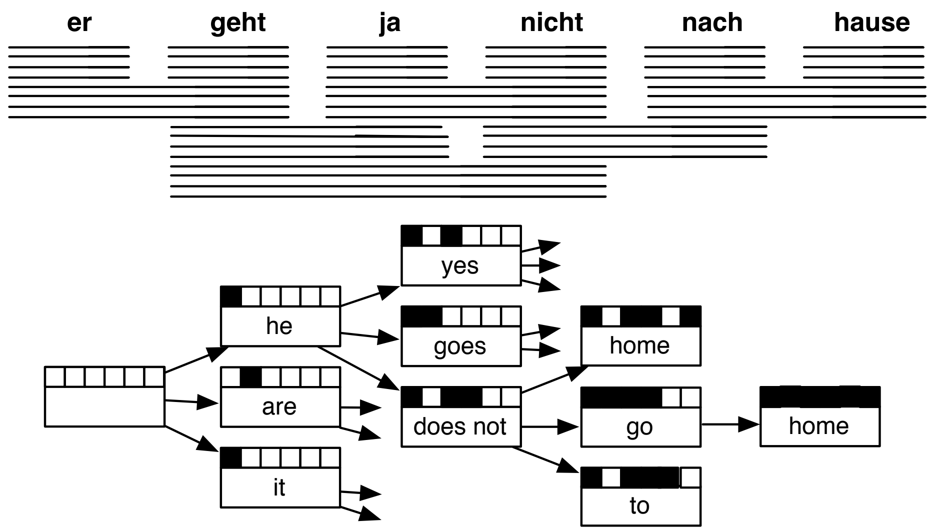
Decoding process: hypothesis expansion



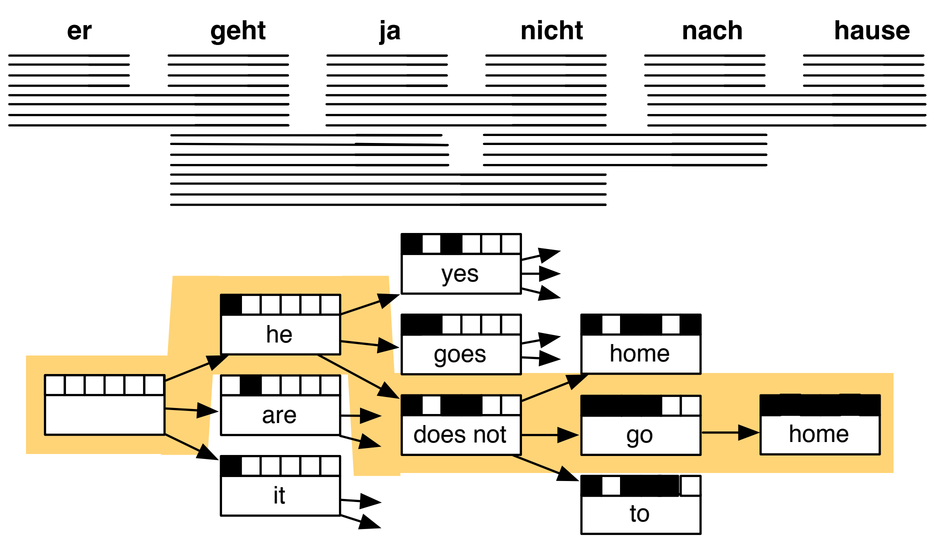
Decoding process: hypothesis expansion



Decoding process: hypothesis expansion



Decoding process: find best path



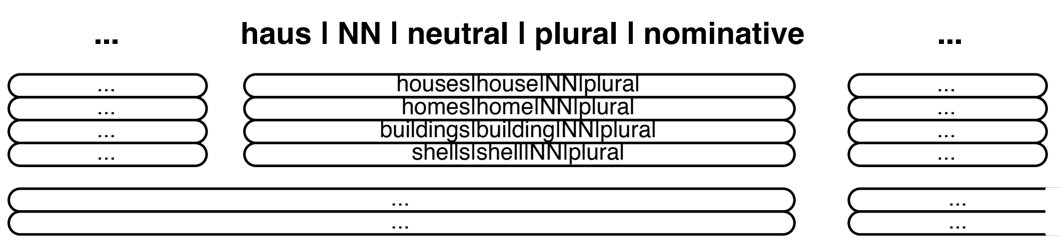
Factored model decoding

- Factored model decoding introduces *additional complexity*
- Hypothesis expansion not any more according to simple translation table, but by *executing a number of mapping steps*, e.g.:
  1. translating of *lemma* → *lemma*
  2. translating of *part-of-speech, morphology* → *part-of-speech, morphology*
  3. generation of *surface form*
- Example: *haus|NN|neutral|plural|nominative*
  → { *houses|house|NN|plural, homes|home|NN|plural,*
  buildings|building|NN|plural, shells|shell|NN|plural }
- Each time, a hypothesis is expanded, these mapping steps have to applied

Efficient factored model decoding

- Key insight: executing of mapping steps can be *pre-computed* and stored as translation options
  - apply mapping steps to all input phrases
  - store results as *translation options*

  → decoding algorithm *unchanged*



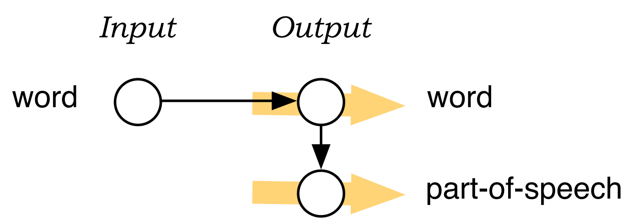
Efficient factored model decoding

- Problem: *Explosion* of translation options
  - originally limited to 20 per input phrase
  - even with simple model, now 1000s of mapping expansions possible
- Solution: *Additional pruning* of translation options
  - *keep only the best* expanded translation options
  - current default 50 per input phrase
  - decoding only about 2-3 times slower than with surface model

Factored Translation Models

- Motivation
- Example
- Model and Training
- Decoding
- **Experiments**
- Outlook

Adding linguistic markup to output



- Generation of POS tags on the target side
- Use of high order language models over POS (7-gram, 9-gram)
- Motivation: syntactic tags should enforce syntactic sentence structure model not strong enough to support major restructuring

Some experiments

- English–German, Europarl, 30 million word, test2006

Model	BLEU
best published result	18.15
baseline (surface)	18.04
surface + POS	18.15

- German–English, News Commentary data (WMT 2007), 1 million word

Model	BLEU
Baseline	18.19
With POS LM	19.05

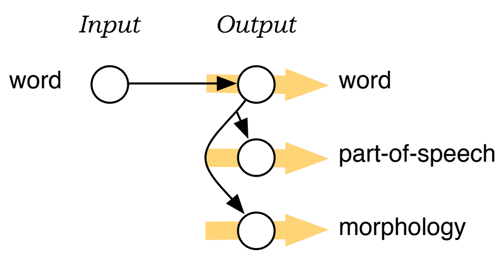
- Improvements under sparse data conditions
- Similar results with CCG supertags [Birch et al., 2007]

Sequence models over morphological tags

die	hellen	Sterne	erleuchten	das	schwarze	Himmel
(the)	(bright)	(stars)	(illuminate)	(the)	(black)	(sky)
fem	fem	fem	-	neutral	neutral	male
plural	plural	plural	plural	sgl.	sgl.	sgl
nom.	nom.	nom.	-	acc.	acc.	acc.

- Violation of noun phrase agreement in gender
  - *das schwarze* and *schwarze Himmel* are perfectly fine bigrams
  - but: *das schwarze Himmel* is not
- If relevant n-grams does not occur in the corpus, a lexical n-gram model would *fail to detect* this mistake
- Morphological sequence model: $p(\text{N-male}|\text{J-male}) > p(\text{N-male}|\text{J-neutral})$

Local agreement (esp. within noun phrases)



- High order language models over POS and morphology
- Motivation
  - *DET-sgl NOUN-sgl* good sequence
  - *DET-sgl NOUN-plural* bad sequence

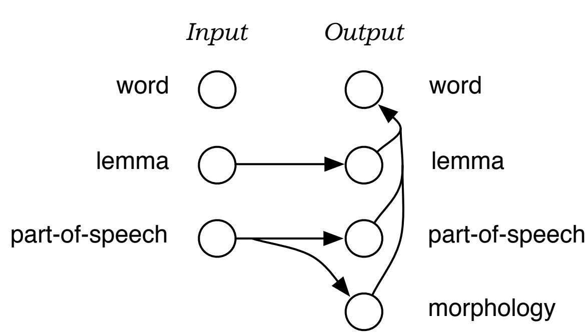
Agreement within noun phrases

- Experiment: 7-gram POS, morph LM in addition to 3-gram word LM
- Results

Method	Agreement errors in NP	devtest	test
baseline	15% in NP $\geq$ 3 words	18.22 BLEU	18.04 BLEU
factored model	4% in NP $\geq$ 3 words	18.25 BLEU	18.22 BLEU

- Example
  - baseline: *... zur zwischenstaatlichen methoden ...*
  - factored model: *... zu zwischenstaatlichen methoden ...*
- Example
  - baseline: *... das zweite wichtige änderung ...*
  - factored model: *... die zweite wichtige änderung ...*

Morphological generation model



- Our motivating example
- Translating lemma and morphological information more robust

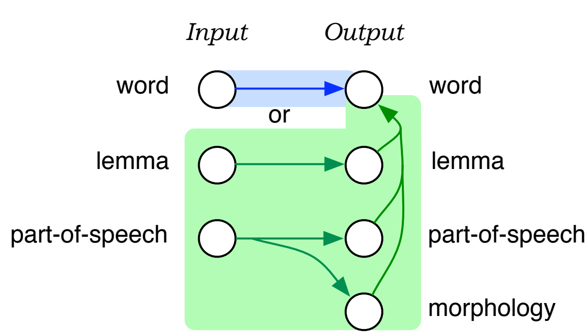
Initial results

- Results on 1 million word News Commentary corpus (German–English)

System	In-doman	Out-of-domain
Baseline	18.19	15.01
With POS LM	19.05	15.03
Morphgen model	14.38	11.65

- What went wrong?
  - why back-off to lemma, when we know how to translate surface forms?
  - → loss of information

Solution: alternative decoding paths



- Allow both surface form translation and morphgen model
  - prefer surface model for known words
  - morphgen model acts as back-off

Results

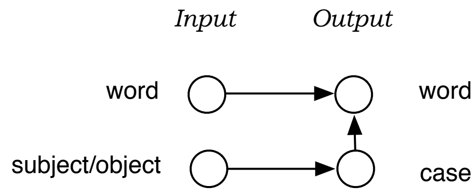
- Model now beats the baseline:

System	In-doman	Out-of-domain
Baseline	**18.19**	**15.01**
With POS LM	19.05	15.03
Morphgen model	14.38	11.65
Both model paths	**19.47**	**15.23**

Adding annotation to the source

- Source words may **lack sufficient information** to map phrases
  - English-German: what case for noun phrases?
  - Chinese-English: plural or singular
  - pronoun translation: what do they refer to?
- Idea: **add additional information** to the source that makes the required information available locally (where it is needed)
- see [Avramidis and Koehn, ACL 2008] for details

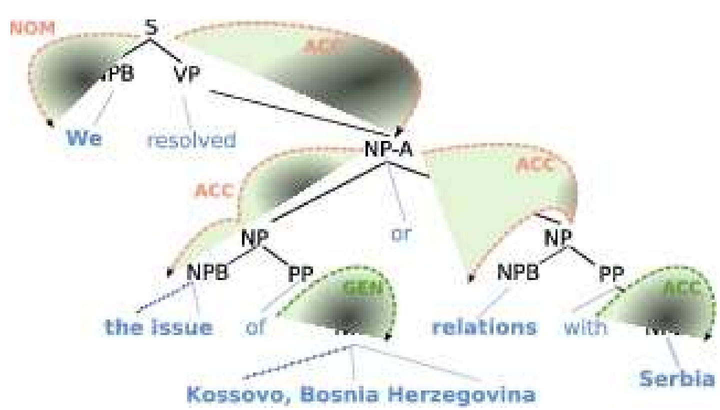
Case Information for English–Greek



- Detect in English, if noun phrase is subject/object (using parse tree)
- Map information into case morphology of Greek
- Use case morphology to generate correct word form

Obtaining Case Information

- Use syntactic parse of English input
  (method similar to semantic role labeling)



Results English-Greek

- Automatic BLEU scores

System	devtest	test07
baseline	18.13	18.05
enriched	18.21	18.20

- Improvement in verb inflection

System	Verb count	Errors	Missing
baseline	311	19.0%	7.4%
enriched	294	5.4%	2.7%

- Improvement in noun phrase inflection

System	NPs	Errors	Missing
baseline	247	8.1%	3.2%
enriched	239	5.0%	5.0%

- Also successfully applied to English-Czech

Factored Translation Models

- Motivation
- Example
- Model and Training
- Decoding
- Experiments
- **Planned Work**

Using POS in reordering

- **Reordering** is often due to syntactic reasons
  - French-English: *NN ADJ* $\rightarrow$ *ADJ NN*
  - Chinese-English: *NN1 F NN2* $\rightarrow$ *NN1 NN2*
  - Arabic-English: *VB NN* $\rightarrow$ *NN VB*
- Extension of lexicalized reordering model
  - already have model that learns *p(monotone|bleue)*
  - can be extended to *p(monotone|ADJ)*
- Gains in preliminary experiments

Shallow syntactic features

the	**paintings**	**of**	**the**	**old**	**man**	**are**	**beautiful**
-	*plural*	-	-	-	*singular*	*plural*	-
B-NP	*I-NP*	*B-PP*	*I-PP*	*I-PP*	*I-PP*	*V*	*B-ADJ*
SBJ	*SBJ*	*OBJ*	*OBJ*	*OBJ*	*OBJ*	*V*	*ADJ*

- Shallow syntactic tasks have been formulated as sequence labeling tasks
  - base noun phrase chunking
  - syntactic role labeling

Long range reordering

- **Long range** reordering
  - movement often not limited to local changes
  - German-English: *SBJ AUX OBJ V* → *SBJ AUX V OBJ*
- **Asynchronous** models
  - some factor mappings (POS, syntactic chunks) may have longer scope than others (words)
  - larger mappings form template for shorter mappings
  - computational problems with this

Discriminative Training

Overview

- Evolution from generative to discriminative models
  - IBM Models: purely generative
  - MERT: discriminative training of generative components
  - More features → better discriminative training needed
- Perceptron algorithm
- Problem: overfitting
- Problem: matching reference translation

The birth of SMT: generative models

- The definition of translation probability follows a **mathematical derivation**

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

- Occasionally, some **independence assumptions** are thrown in for instance IBM Model 1: word translations are independent of each other

$$p(\mathbf{e}|\mathbf{f}, a) = \frac{1}{Z} \prod_i p(e_i|f_{a(i)})$$

- Generative story leads to **straight-forward estimation**
  - maximum likelihood estimation of component probability distribution
  - **EM algorithm** for discovering hidden variables (alignment)

Log-linear models

- IBM Models provided mathematical justification for factoring **components** together

$$p_{LM} \times p_{TM} \times p_D$$

- These may be **weighted**

$$p_{LM}^{\lambda_{LM}} \times p_{TM}^{\lambda_{TM}} \times p_D^{\lambda_D}$$

- **Many components** p_i with weights λ_i

$$\prod_i p_i^{\lambda_i} = exp(\sum_i \lambda_i log(p_i))$$

$$log \prod_i p_i^{\lambda_i} = \sum_i \lambda_i log(p_i)$$

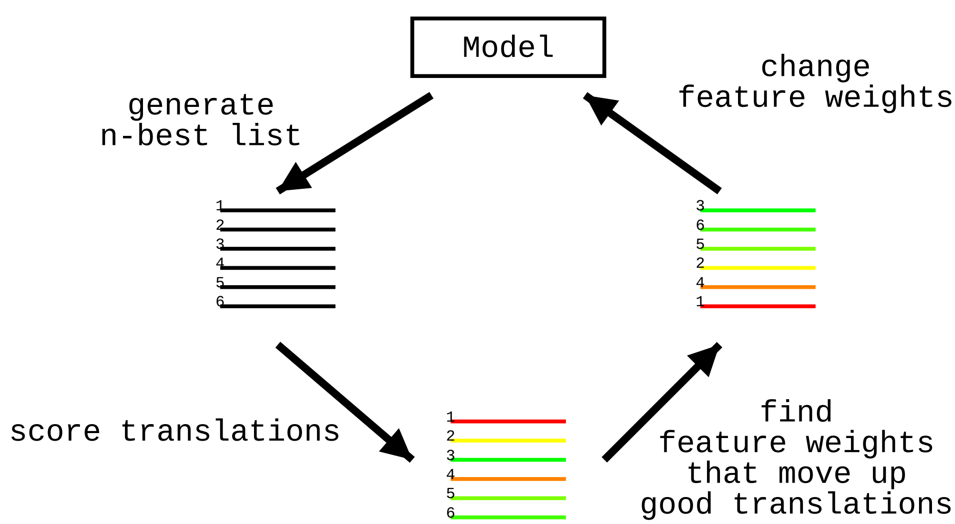
Knowledge sources

- Many different **knowledge sources** useful
  - language model
  - reordering (distortion) model
  - phrase translation model
  - word translation model
  - word count
  - phrase count
  - drop word feature
  - phrase pair frequency
  - additional language models
  - additional features

Set feature weights

- Contribution of components p_i determined by weight λ_i
- Methods
  - *manual setting* of weights: try a few, take best
  - *automate* this process
- Learn weights
  - set aside a **development corpus**
  - set the weights, so that **optimal translation performance** on this development corpus is achieved
  - requires *automatic scoring* method (e.g., BLEU)

Discriminative training



Discriminative vs. generative models

- Generative models
  - translation process is broken down to *steps*
  - each step is modeled by a *probability distribution*
  - each probability distribution is estimated from the data by *maximum likelihood*
- Discriminative models
  - model consist of a number of *features* (e.g. the language model score)
  - each feature has a *weight*, measuring its value for judging a translation as correct
  - feature weights are *optimized on development data*, so that the system output matches correct translations as close as possible

Discriminative training

- Training set (*development set*)
  - different from original training set
  - small (maybe 1000 sentences)
  - must be different from test set
- Current model *translates* this development set
  - *n-best list* of translations (n=100, 10000)
  - translations in n-best list can be *scored*
- Feature weights are *adjusted*
- N-Best list generation and feature weight adjustment repeated for a number of iterations

Learning task

- Task: *find weights*, so that feature vector of the correct translations *ranked first*

rank	translation	LM	TM	WP	SER
1	Mary not give slap witch green .	-17.2	-5.2	-7	1
2	Mary not slap the witch green .	-16.3	-5.7	-7	1
3	Mary not give slap of the green witch .	-18.1	-4.9	-9	1
4	Mary not give of green witch .	-16.5	-5.1	-8	1
5	Mary did not slap the witch green .	-20.1	-4.7	-8	1
6	Mary did not slap green witch .	-15.5	-3.2	-7	1
7	Mary not slap of the witch green .	-19.2	-5.3	-8	1
8	Mary did not give slap of witch green .	-23.2	-5.0	-9	1
9	Mary did not give slap of the green witch .	-21.8	-4.4	-10	1
10	Mary did slap the witch green .	-15.5	-6.9	-7	1
11	**Mary did not slap the green witch .**	**-17.4**	**-5.3**	**-8**	**0**
12	Mary did slap witch green .	-16.9	-6.9	-6	1
13	Mary did slap the green witch .	-14.3	-7.1	-7	1
14	Mary did not slap the of green witch .	-24.2	-5.3	-9	1
15	Mary did not give slap the witch green .	-25.2	-5.5	-9	1

rank translation — feature vector (LM, TM, WP)

Och's minimum error rate training (MERT)

- **Line search** for best feature weights

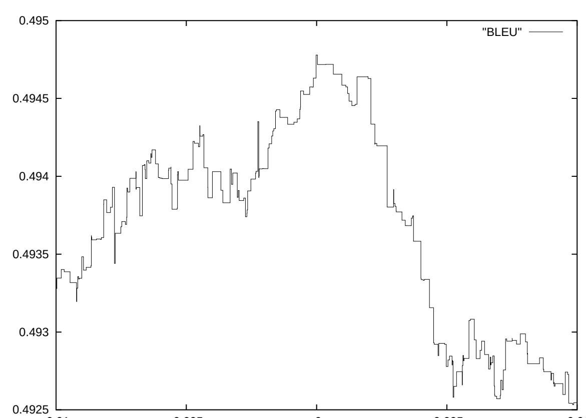
```
given:  sentences with n-best list of
translations
iterate n times
    randomize starting feature weights
        iterate until convergences
            for each feature
                find best feature weight
                update if different from current
return best feature weights found in any
iteration
```

Methods to adjust feature weights

- **Maximum entropy** [Och and Ney, ACL2002]
  - match *expectation* of feature values of model and data
- **Minimum error rate** training [Och, ACL2003]
  - try to *rank best translations first* in n-best list
  - can be adapted for various error metrics, even BLEU
- **Ordinal regression** [Shen et al., NAACL2004]
  - *separate* k worst from the k best translations

BLEU error surface

- Varying one parameter: a rugged line with many local optima



Unstable outcomes: weights vary

component	run 1	run 2	run 3	run 4	run 5	run 6
distance	0.059531	0.071025	0.069061	0.120828	0.120828	0.072891
lexdist 1	0.093565	0.044724	0.097312	0.108922	0.108922	0.062848
lexdist 2	0.021165	0.008882	0.008607	0.013950	0.013950	0.030890
lexdist 3	0.083298	0.049741	0.024822	-0.000598	-0.000598	0.023018
lexdist 4	0.051842	0.108107	0.090298	0.111243	0.111243	0.047508
lexdist 5	0.043290	0.047801	0.020211	0.028672	0.028672	0.050748
lexdist 6	0.083848	0.056161	0.103767	0.032869	0.032869	0.050240
lm 1	0.042750	0.056124	0.052090	0.049561	0.049561	0.059518
lm 2	0.019881	0.012075	0.022896	0.035769	0.035769	0.026414
lm 3	0.059497	0.054580	0.044363	0.048321	0.048321	0.056282
ttable 1	0.052111	0.045096	0.046655	0.054519	0.054519	0.046538
ttable 1	0.052888	0.036831	0.040820	0.058003	0.058003	0.066308
ttable 1	0.042151	0.066256	0.043265	0.047271	0.047271	0.052853
ttable 1	0.034067	0.031048	0.050794	0.037589	0.037589	0.031939
phrase-pen.	0.059151	0.062019	-0.037950	0.023414	0.023414	-0.069425
word-pen	-0.200963	-0.249531	-0.247089	-0.228469	-0.228469	-0.252579

Unstable outcomes: scores vary

- Even different scores with different runs (varying 0.40 on dev, 0.89 on test)

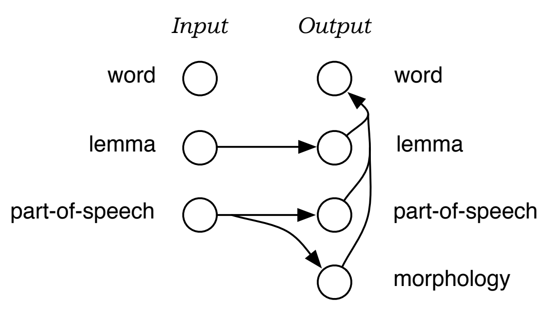
run	iterations	dev score	test score
1	8	50.16	51.99
2	9	50.26	51.78
3	8	50.13	51.59
4	12	50.10	51.20
5	10	50.16	51.43
6	11	50.02	51.66
7	10	50.25	51.10
8	11	50.21	51.32
9	10	50.42	51.79

More features: more components

- We would like to add **more components** to our model
  - multiple language models
  - domain adaptation features
  - various special handling features
  - using linguistic information

→ MERT becomes even **less reliable**
  - runs many more iterations
  - fails more frequently

More features: factored models



- Factored translation models break up phrase mapping into smaller steps
  - multiple translation tables
  - multiple generation tables
  - multiple language models and sequence models on factors

→ **Many more features**

Millions of features

- Why **mix** of discriminative training and generative models?
- Discriminative training of all components
  - phrase table [Liang et al., 2006]
  - language model [Roark et al, 2004]
  - additional features
- **Large-scale** discriminative training
  - millions of features
  - training of full training set, not just a small development corpus

Perceptron algorithm

- Translate each sentence
- If no match with reference translation: update features

```
set all lambda = 0
do until convergence
    for all foreign sentences f
        set e-best to best translation according to model
        set e-ref to reference translation
        if e-best != e-ref
            for all features feature-i
                lambda-i += feature-i(f,e-ref)
                          - feature-i(f,e-best)
```

Problem: overfitting

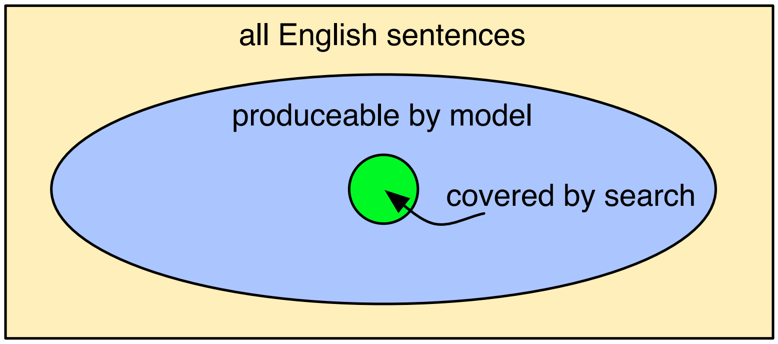
- Fundamental problem in machine learning
  - what works best for training data, may not work well in general
  - **rare, unrepresentative features** may get too much weight
- **Especially severe problem** in phrase-based models
  - **long phrase pairs** explain well *individual sentences*
  - ... but are less general, *suspect to noise*
  - EM training of phrase models [Marcu and Wong, 2002] has same problem

Solutions

- **Restrict to short phrases**, e.g., maximum 3 words (current approach)
  - limits the power of phrase-based models
  - ... but not very much [Koehn et al, 2003]
- **Jackknife**
  - collect phrase pairs from one part of corpus
  - optimize their feature weights on another part
- IBM direct model: **only one-to-many** phrases [Ittycheriah and Salim Roukos, 2007]

Problem: reference translation

- Reference translation may be anywhere in this box



- If produceable by model → we can compute feature scores
- If not → we can not

Some solutions

- **Skip sentences**, for which reference can not be produced
  – invalidates large amounts of training data
  – biases model to shorter sentences
- Declare candidate translations closest to reference as **surrogate**
  – closeness measured for instance by smoothed BLEU score
  – may be not a very good translation: odd feature values, training is severely distorted

Experiment

- Skipping sentences with unproduceable reference **hurts**

Handling of reference	BLEU
with skipping	25.81
w/o skipping	29.61

- When including all sentences: surrogate reference picked from 1000-best list using maximum *smoothed BLEU score* with respect to reference translation
- Czech-English task, **only binary features**
  - phrase table features
  - lexicalized reordering features
  - source and target phrase bigram
- See also [Liang et al., 2006] for similar approach

Better solution: early updating?

- At some point the reference translation **falls out** of the search space
  - for instance, due to *unknown words*:

Reference: `The group attended the meeting in` `Najaf ...`

System: `The group meeting was attended in` `UNKNOWN ...`

only update features involved in this part

- Early updating [Collins et al., 2005]:
  - stop search, when reference translation is not covered by model
  - only update **features involved in partial** reference / system output

Conclusions

- Currently have proof-of-concept implementation
- Future work: Overcome various technical challenges
  - reference translation may not be produceable
  - overfitting
  - mix of binary and real-valued features
  - scaling up
- More and more features are unavoidable, let's deal with them