MT Marathon

26th – 30th January 2009

Prague, Czech Republic

Lectures, Talks, Labs

	Morning Lecture	Research Talks	Afternoon Lab Session	Evening
	9.00 – 10.30	11.00 – 12.30	14.00 – 17.00	17.00 – ?
Monday	Introduction to MT and MT Evaluation (Adam Lopez)	Projects: Introduction Intro to Stat-XFER (Alon Lavie) Intro to TectoMT (Zdeněk Žabokrtský, Ondřej Bojar)	Manual judgement of MT quality	
Tuesday	Word Alignment (Barry Haddow)	PostCAT, apertium-cy, MBMT	Implementing IBM model 1	Projects: Update
Wednesday	Phrase-Based Models and Decoding (Chris Dyer)	Joshua, SAMT, MERT+	Installing and running Moses (Hieu Hoang and Josh Schroeder)	
Thursday	TectoMT: Processing Trees (Zdeněk Žabokrtský and others)	RIA, Sub-Tree Aligner	TectoMT hands-on experience: Installation and tutorial (Jana Kravalová)	Projects: Update
Friday	Richer Models and Optimization (Philipp Koehn)	Projects: Final Short Presentations	Using factored models and MERT in Moses (Hieu Hoang, Barry Haddow, Abhishek Arun)	

Accepted Contributions, Research Talks

The following contributions will be presented during late mornings:

1. apertium-cy: **F. M. Tyers, K. Donnelly**: apertium-cy - a collaboratively-developed free RBMT system for Welsh to English
2. Joshua: **Z. Li, C. Callison-Burch, W. Thornton, S. Khudanpur**: Joshua: an Open-source Decoder for Parsing-based Machine Translation
3. MBMT: **A. van den Bosch and P. Berck**: Memory-Based Machine Translation and Language Modeling
4. MERT+: **N. Bertoldi, B. Haddow, J.-B. Fouet**: Improved Minimum Error Rate Training in Moses
5. PostCAT: **J. Graça, K. Ganchev, B. Taskar**: PostCAT - Posterior Constrained Alignment Toolkit
6. RIA: **Y. Graham, J. van Genabith**: An Open Source Rule Induction Tool for Transfer-Based SMT
7. SAMT: **A. Venugopal, A. Zollmann**: Grammar based statistical MT on Hadoop. An end-to-end toolkit for large scale PSCFG based statistical machine translation
8. Sub-Tree Aligner: **V. Zhechev**: Unsupervised Generation of Parallel Treebanks through Sub-Tree Alignment
9. Z-MERT: **O. Zaidan**: Z-MERT: A Fully Configurable Open Source Tool for Minimum Error Rate Training of Machine Translation Systems (There is no presentation for this paper.)

Research talk presentations should be 20 to 25 minutes long with additional 5 minutes for a discussion.

Contents

Statistical Machine Translation

presentation: Adam Lopez
slides: 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

$$p(e|f)$$

Probabilities

- Find most probable English sentence given a foreign language sentence

$$p(e|f)$$

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

Probabilities

- Find most probable English sentence given a foreign language sentence

$$p(e|f)$$

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

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

Probabilities

- Find most probable English sentence given a foreign language sentence

$$p(e|f)$$

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

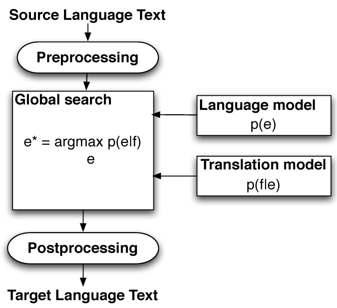
$$p(e|f) = \frac{p(e)p(f|e)}{p(f)}$$

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

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) =

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

Trigram language model

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

$$\begin{aligned}&.8 * p(w_3|w_1w_2) + \\ &.15 * p(w_3|w_2) + \\ &.049 * p(w_3) + \\ &.001\end{aligned}$$

- 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.

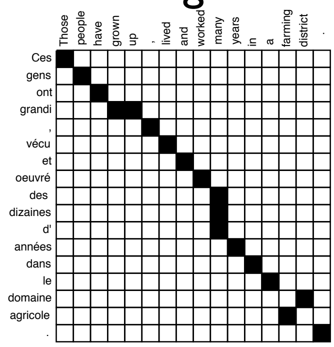
Translation model

- 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

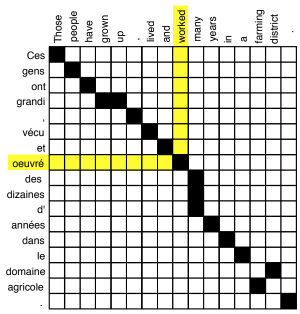
- So we can calculate translation probabilities by way of these alignment probabilities

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

- 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!

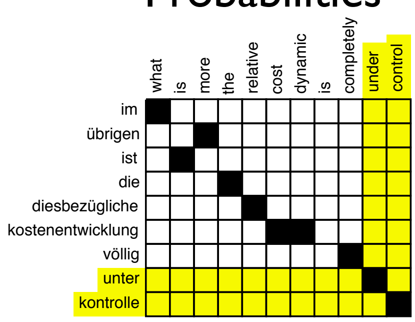
$$= \frac{count(f_j, e_i)}{count(e_i)}$$

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

Calculating $t(f_j|e_i)$

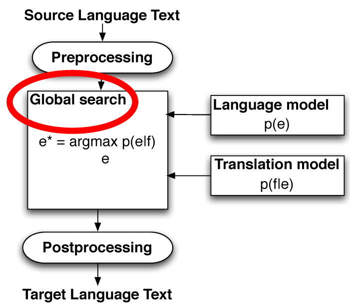
- Unfortunately we don't have word aligned data, so we can't do this directly.
- OK, so it's not quite as easy as I said.
- Tomorrow's lecture will describe how word alignments are obtained using Expectation Maximization.

Phrase Translation Probabilities



Phrase Translation Probabilities

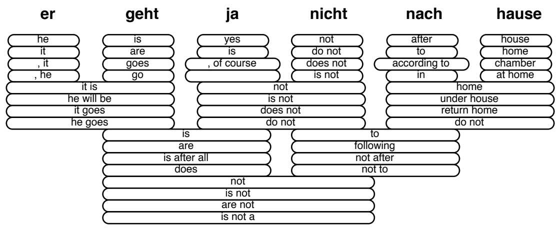
``Diagram Number 1"



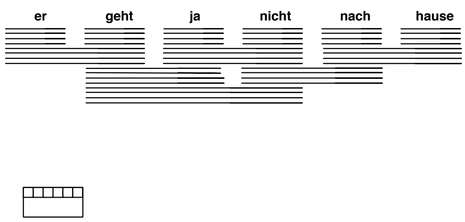
The Search Process AKA ``Decoding"

- Look up all translations of every source phrase
- 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

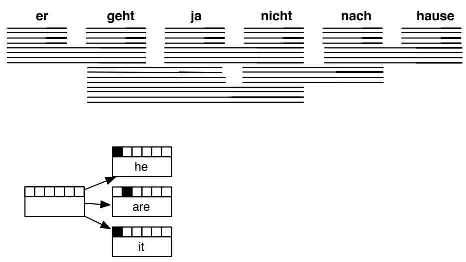
Translation Options



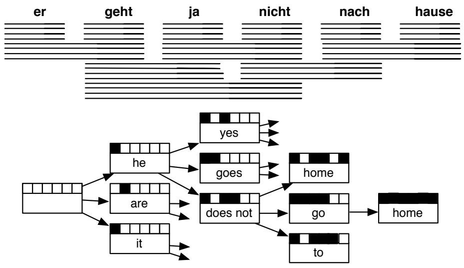
Search



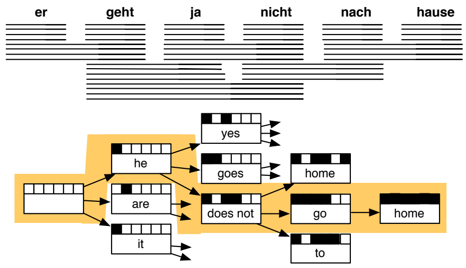
Search



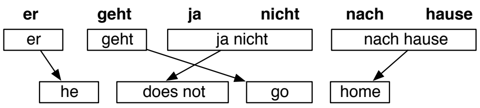
Search



Search



Best Translation



The Search Space

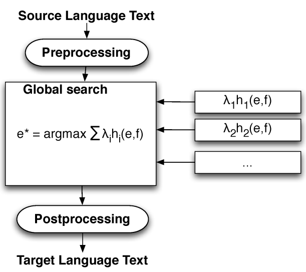
- In the end the item which covers all of the source words and which has the highest probability wins!
- That's our best translation

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

- And there was much rejoicing!

Alternative models

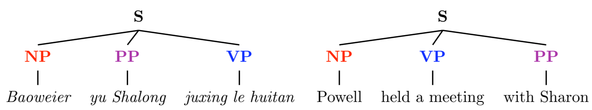
Linear models



Alternative models

Tree-based models

$$\begin{array}{rcll}
\mathbf{S} & \rightarrow & \mathbf{NP}^{(1)}\ \mathbf{PP}^{(2)}\ \mathbf{VP}^{(3)}, & \mathbf{NP}^{(1)}\ \mathbf{VP}^{(3)}\ \mathbf{PP}^{(2)} \\
\mathbf{NP} & \rightarrow & \textit{Baoweier}, & \text{Powell} \\
\mathbf{PP} & \rightarrow & \textit{yu Shalong}, & \text{with Sharon} \\
\mathbf{VP} & \rightarrow & \textit{juxing le huitan}, & \text{held a meeting}
\end{array}$$



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

More Information

- http://www.statmt.org - papers, tutorials, etc.
- Statistical Machine Translation. In *ACM Computing Surveys* 40(3), Aug 2008.

  At http://homepages.inf.ed.ac.uk/alopez

  BibTeX at http://github.com/alopez/smtbib

Evaluating Translation Quality

Presentation: Adam Lopez
Slides: 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

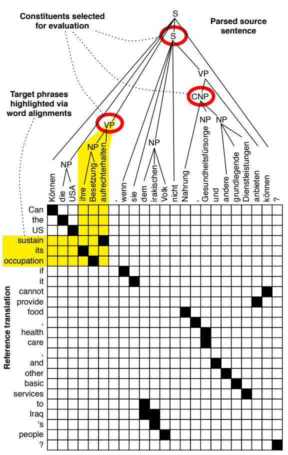
- This text contains how much of the information in the reference translation:
- 5) All
  4) Most
  3) Much
  2) Little
  1) None

Relative ranking

- An alternative to absolute scales
- Simply ask
  - Is A better than B?
  - Is B better than A?
  - Or are they indistinguishable?

Consistent-based evaluation

- Rather than ranking the translations of whole sentences, instead have people focus on smaller parts



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

ACL Workshop on SMT

- Translation between English, French, German, Spanish, Hungarian and Czech
- 30 different systems
- In-domain and out-of-domain test sets
- Scores produced by multiple automatic metrics
- Systems ranked by 100+ human judges

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

Homework Exercise

- Evaluation exercise for homework
- 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!

Stat-XFER: A General Framework for Search-based Syntax-driven MT

Alon Lavie
Language Technologies Institute
Carnegie Mellon University

Joint work with:
Greg Hanneman, Vamshi Ambati, Alok Parlikar, Edmund Huber, Jonathan Clark, Erik Peterson, Christian Monson, Abhaya Agarwal, Kathrin Probst, Ari Font Llitjos, Lori Levin, Jaime Carbonell, Bob Frederking, Stephan Vogel

Outline

- Context and Rationale
- CMU Statistical Transfer MT Framework
- Extracting Syntax-based MT Resources from Parallel-corpora
- Integrating Syntax-based and Phrase-based Resources
- Open Research Problems
- Conclusions

Rule-based vs. Statistical MT

- Traditional Rule-based MT:
  - Expressive and linguistically-rich formalisms capable of describing complex mappings between the two languages
  - Accurate “clean” resources
  - Everything constructed manually by experts
  - Main challenge: obtaining and maintaining broad coverage
- Phrase-based Statistical MT:
  - Learn word and phrase correspondences automatically from large volumes of parallel data
  - Search-based “decoding” framework:
    - Models propose many alternative translations
    - Effective search algorithms find the “best” translation
  - Main challenge: obtaining and maintaining high translation accuracy

Research Goals

- Long-term research agenda (since 2000) focused on developing a unified framework for MT that addresses the core fundamental weaknesses of previous approaches:
  - Representation – explore richer formalisms that can capture complex divergences between languages
  - Ability to handle morphologically complex languages
  - Methods for automatically acquiring MT resources from available data and combining them with manual resources
  - Ability to address both rich and poor resource scenarios
- Main research funding sources: NSF (AVENUE and LETRAS projects) and DARPA (GALE)

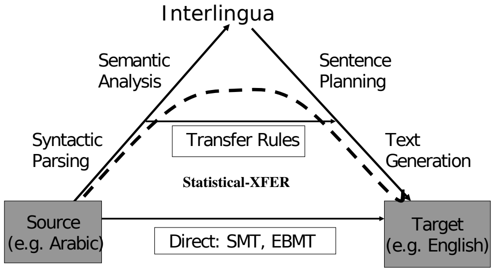
CMU Statistical Transfer (Stat-XFER) MT Approach

- Integrate the major strengths of rule-based and statistical MT within a common framework:
  - Linguistically rich formalism that can express complex and abstract compositional transfer rules
  - Rules can be written by human experts and also acquired automatically from data
  - Easy integration of morphological analyzers and generators
  - Word and syntactic-phrase correspondences can be automatically acquired from parallel text
  - Search-based decoding from statistical MT adapted to find the best translation within the search space: multi-feature scoring, beam-search, parameter optimization, etc.
  - Framework suitable for both resource-rich and resource-poor language scenarios

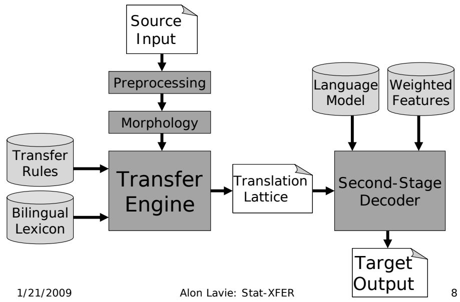
Stat-XFER Main Principles

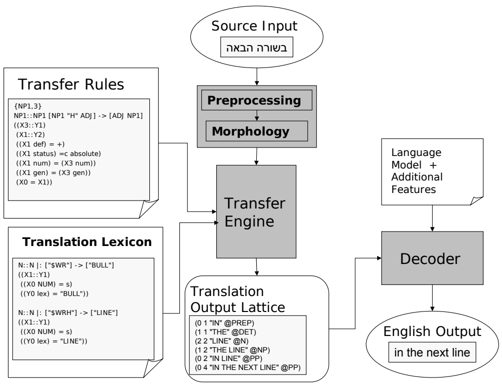
- Framework: Statistical search-based approach with syntactic translation transfer rules that can be acquired from data but also developed and extended by experts
- Automatic Word and Phrase translation lexicon acquisition from parallel data
- Transfer-rule Learning: apply ML-based methods to automatically acquire syntactic transfer rules for translation between the two languages
- Elicitation: use bilingual native informants to produce a small high-quality word-aligned bilingual corpus of translated phrases and sentences
- Rule Refinement: refine the acquired rules via a process of interaction with bilingual informants
- XFER + Decoder:
  - XFER engine produces a lattice of possible transferred structures at all levels
  - Decoder searches and selects the best scoring combination

Stat-XFER MT Approach

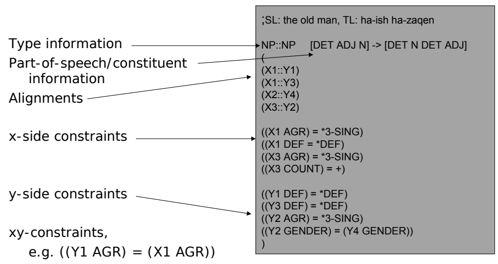


Stat-XFER Framework

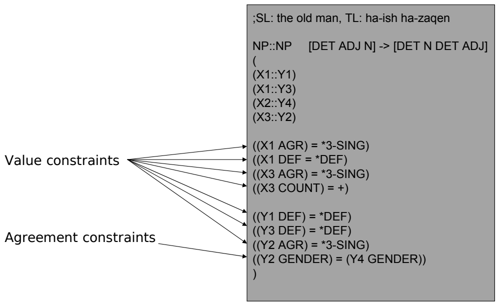




Transfer Rule Formalism



Transfer Rule Formalism



Translation Lexicon: Hebrew-to-English Examples (Semi-manually-developed)

```
PRO::PRO |: ["ANI"] -> ["I"]
(
(X1::Y1)
((X0 per) = 1)
((X0 num) = s)
((X0 case) = nom)
)

PRO::PRO |: ["ATH"] -> ["you"]
(
(X1::Y1)
((X0 per) = 2)
((X0 num) = s)
((X0 gen) = m)
((X0 case) = nom)
)
```

```
N::N |: ["$&H"] -> ["HOUR"]
(
(X1::Y1)
((X0 NUM) = s)
((Y0 NUM) = s)
((Y0 lex) = "HOUR")
)

N::N |: ["$&H"] -> ["hours"]
(
(X1::Y1)
((Y0 NUM) = p)
((X0 NUM) = p)
((Y0 lex) = "HOUR")
)
```

Translation Lexicon: French-to-English Examples (Automatically-acquired)

```
DET::DET |: [“le"] -> [“the"]
(
(X1::Y1)
)

Prep::Prep |:[“dans”] -> [“in”]
(
(X1::Y1)
)

N::N |: [“principes"] -> [“principles"]
(
(X1::Y1)
)

N::N |: [“respect"] -> [“accordance"]
(
(X1::Y1)
)
```

```
NP::NP |: [“le respect"] -> [“accordance"]
(
)

PP::PP |: [“dans le respect"] -> [“in accordance"]
(
)

PP::PP |: [“des principes"] -> [“with the principles"]
(
)
```


Hebrew-English Transfer Grammar Example Rules (Manually-developed)

```
{NP1,2}
;;SL: $MLH ADWMH
;;TL: A RED DRESS

NP1::NP1 [NP1 ADJ] -> [ADJ NP1]
(
(X2::Y1)
(X1::Y2)
((X1 def) = -)
((X1 status) =c absolute)
((X1 num) = (X2 num))
((X1 gen) = (X2 gen))
(X0 = X1)
)
```

```
{NP1,3}
;;SL: H $MLWT H ADWMWT
;;TL: THE RED DRESSES

NP1::NP1 [NP1 "H" ADJ] -> [ADJ NP1]
(
(X3::Y1)
(X1::Y2)
((X1 def) = +)
((X1 status) =c absolute)
((X1 num) = (X3 num))
((X1 gen) = (X3 gen))
(X0 = X1)
)
```


French-English Transfer Grammar Example Rules (Automatically-acquired)

```
{PP,24691}
;;SL: des principes
;;TL: with the principles

PP::PP [“des” N] -> [“with the” N]
(
(X1::Y1)
)
```

```
{PP,312}
;;SL: dans le respect des principes
;;TL: in accordance with the principles

PP::PP [Prep NP] -> [Prep NP]
(
(X1::Y1)
(X2::Y2)
)
```


The Transfer Engine

- Input: source-language input sentence, or source-language confusion network
- Output: lattice representing collection of translation fragments at all levels supported by transfer rules
- Basic Algorithm: “bottom-up” integrated “parsing-transfer-generation” chart-parser guided by the synchronous transfer rules
  - Start with translations of individual words and phrases from translation lexicon
  - Create translations of larger constituents by applying applicable transfer rules to previously created lattice entries
  - Beam-search controls the exponential combinatorics of the search-space, using multiple scoring features

The Transfer Engine

- Some Unique Features:
  - Works with either learned or manually-developed transfer grammars
  - Handles rules with or without unification constraints
  - Supports interfacing with servers for morphological analysis and generation
  - Can handle ambiguous source-word analyses and/or SL segmentations represented in the form of lattice structures

Hebrew Example
(From [Lavie et al., 2004])

- Input word: B$WRH

```
0    1    2    3    4
|--------B$WRH--------|
|-----B-----|$WR|--H--|
|--B--|-H--|--$WRH---|
```

Hebrew Example

(From [Lavie et al., 2004])

```
Y0: ((SPANSTART 0)
     (SPANEND 4)
     (LEX B$WRH)
     (POS N)
     (GEN F)
    (NUM S)
    (STATUS ABSOLUTE))

Y1: ((SPANSTART 0)
      (SPANEND 2)
      (LEX B)
      (POS PREP))

Y2: ((SPANSTART 1)
      (SPANEND 3)
      (LEX $WR)
      (POS N)
      (GEN M)
     (NUM S)
      (STATUS ABSOLUTE))

Y3: ((SPANSTART 3)
      (SPANEND 4)
      (LEX $LH)
      (POS POSS))

Y4: ((SPANSTART 0)
      (SPANEND 1)
      (LEX B)
      (POS PREP))

Y5: ((SPANSTART 1)
      (SPANEND 2)
     (LEX H)
      (POS DET))

Y6: ((SPANSTART 2)
      (SPANEND 4)
      (LEX $WRH)
      (POS N)
      (GEN F)
      (NUM S)
      (STATUS ABSOLUTE))

Y7: ((SPANSTART 0)
      (SPANEND 4)
      (LEX B$WRH)
      (POS LEX))
```

1/21/2009 Alon Lavie: Stat-XFER 19

XFER Output Lattice

```
(28 28 "AND" -5.6988 "W" "(CONJ,0 'AND')")
(29 29 "SINCE" -8.20817 "MAZ " "(ADVP,0 (ADV,5 'SINCE')) ")
(29 29 "SINCE THEN" -12.0165 "MAZ " "(ADVP,0 (ADV,6 'SINCE THEN')) ")
(29 29 "EVER SINCE" -12.5564 "MAZ " "(ADVP,0 (ADV,4 'EVER SINCE')) ")
(30 30 "WORKED" -10.9913 "&BD " "(VERB,0 (V,11 'WORKED')) ")
(30 30 "FUNCTIONED" -16.0023 "&BD " "(VERB,0 (V,10 'FUNCTIONED')) ")
(30 30 "WORSHIPPED" -17.3393 "&BD " "(VERB,0 (V,12 'WORSHIPPED')) ")
(30 30 "SERVED" -11.5161 "&BD " "(VERB,0 (V,14 'SERVED')) ")
(30 30 "SLAVE" -13.9523 "&BD " "(NP0,0 (N,34 'SLAVE')) ")
(30 30 "BONDSMAN" -18.0325 "&BD " "(NP0,0 (N,36 'BONDSMAN')) ")
(30 30 "A SLAVE" -16.8671 "&BD " "(NP,1  (LITERAL 'A') (NP2,0 (NP1,0 (NP0,0
          (N,34 'SLAVE')) ) ) ) ")
(30 30 "A BONDSMAN" -21.0649 "&BD " "(NP,1  (LITERAL 'A') (NP2,0 (NP1,0
          (NP0,0 (N,36 'BONDSMAN')) ) ) ) ")
```

1/21/2009 Alon Lavie: Stat-XFER 20

The Lattice Decoder

- Stack Decoder, similar to standard Statistical MT decoders
- Searches for best-scoring path of non-overlapping lattice arcs
- No reordering during decoding
- Scoring based on log-linear combination of scoring features, with weights trained using Minimum Error Rate Training (MERT)
- Scoring components:
  - Statistical Language Model
  - Bi-directional MLE phrase and rule scores
  - Lexical Probabilities
  - Fragmentation: how many arcs to cover the entire translation?
  - Length Penalty: how far from expected target length?

1/21/2009 Alon Lavie: Stat-XFER 21

XFER Lattice Decoder

```
0 0     ON THE FOURTH DAY THE LION ATE THE RABBIT TO A MORNING MEAL
Overall: -8.18323, Prob: -94.382, Rules: 0, Frag: 0.153846, Length: 0,
          Words: 13,13
235 < 0 8 -19.7602: B H IWM RBI&I  (PP,0 (PREP,3 'ON')(NP,2 (LITERAL 'THE')
          (NP2,0 (NP1,1 (ADJ,2 (QUANT,0 'FOURTH'))(NP1,0 (NP0,1 (N,6 'DAY')))))))>
918 < 8 14 -46.2973: H ARIH AKL AT H $PN  (S,2 (NP,2 (LITERAL 'THE') (NP2,0
          (NP1,0 (NP0,1 (N,17 'LION'))))))(VERB,0 (V,0 'ATE'))(NP,100
          (NP,2 (LITERAL 'THE') (NP2,0 (NP1,0 (NP0,1 (N,24 'RABBIT')))))))>
584 < 14 17 -30.6607: L ARWXH BWQR  (PP,0 (PREP,6 'TO')(NP,1 (LITERAL 'A')
          (NP2,0 (NP1,0 (NNP,3 (NP0,0 (N,32 'MORNING'))(NP0,0 (N,27 'MEAL')))))))>
```

1/21/2009 Alon Lavie: Stat-XFER 22

Stat-XFER MT Systems

- General Stat-XFER framework under development for past seven years
- Systems so far:
  - Chinese-to-English
  - French-to-English
  - Hebrew-to-English
  - Urdu-to-English
  - German-to-English
  - Hindi-to-English
  - Dutch-to-English
  - Turkish-to-English
  - Mapudungun-to-Spanish
- In progress or planned:
  - Arabic-to-English
  - Brazilian Portuguese-to-English
  - English-to-Arabic
  - Hebrew-to-Arabic
  - Czech-to-English

1/21/2009 Alon Lavie: Stat-XFER 23

Syntax-based MT Resource Acquisition in Resource-rich Scenarios

- Scenario: Significant amounts of parallel-text at sentence-level are available
  - Parallel sentences can be word-aligned and parsed (at least on one side, ideally on both sides)
- Goal: Acquire both broad-coverage translation lexicons and transfer rule grammars automatically from the data
- Syntax-based translation lexicons:
  - Broad-coverage constituent-level translation equivalents at all levels of granularity
  - Can serve as the elementary building blocks for transfer trees constructed at runtime using the transfer rules

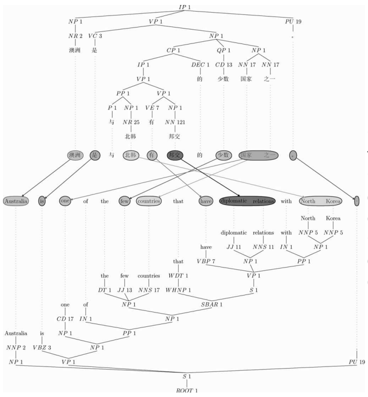
1/21/2009 Alon Lavie: Stat-XFER 24

Syntax-driven Resource Acquisition Process

- Automatic Process for Extracting Syntax-driven Rules and Lexicons from sentence-parallel data:
  1. Word-align the parallel corpus (GIZA++)
  2. Parse the sentences **independently** for both languages
  3. Tree-to-tree Constituent Alignment:
     a) Run our new **Constituent Aligner** over the parsed sentence pairs
     b) Enhance alignments with additional Constituent Projections
  4. Extract all **aligned constituents** from the parallel trees
  5. Extract all **derived synchronous transfer rules** from the constituent-aligned parallel trees
  6. Construct a **"data-base"** of all extracted parallel constituents and synchronous rules **with their frequencies** and model them statistically (assign them **relative-likelihood probabilities**)

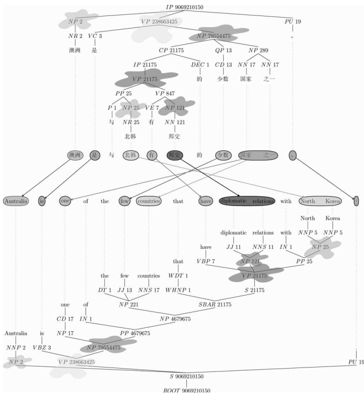
PFA Constituent Node Aligner

- Input: a bilingual pair of parsed and word-aligned sentences
- Goal: find all sub-sentential constituent alignments between the two trees which are translation equivalents of each other
- Equivalence Constraint: a pair of constituents <S,T> are considered translation equivalents if:
  - All words in yield of <S> are aligned only to words in yield of <T> (and vice-versa)
  - If <S> has a sub-constituent <S1> that is aligned to <T1>, then <T1> must be a sub-constituent of <T> (and vice-versa)
- Algorithm is a bottom-up process starting from word-level, marking nodes that satisfy the constraints



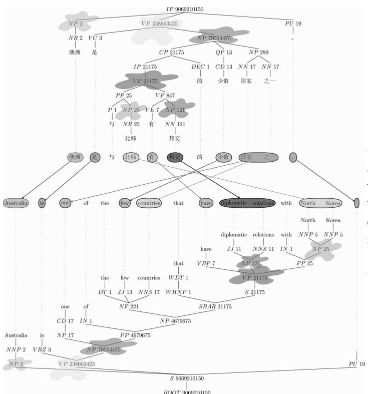
PFA Node Alignment Algorithm Example

- Words don't have to align one-to-one
- Constituent labels can be different in each language
- Tree Structures can be highly divergent



PFA Node Alignment Algorithm Example

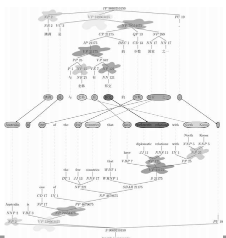
- Aligner uses a clever arithmetic manipulation to enforce equivalence constraints
- Resulting aligned nodes are highlighted in figure



PFA Node Alignment Algorithm Example

Extraction of Phrases:
- Get the **yields** of the aligned nodes and add them to a phrase table tagged with **syntactic categories** on both source and target sides

- Example:
NP # NP ::
澳洲 # Australia



PFA Node Alignment Algorithm Example

All Phrases from this tree pair:

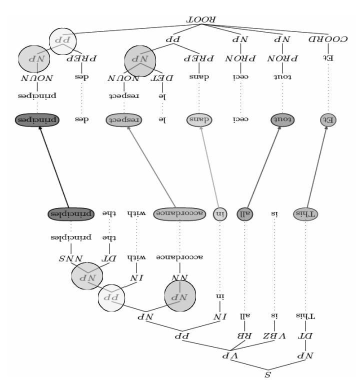
1.IP # S :: 澳洲 是 与 北韩 有 邦交 的 少数 国家 之一 。 # Australia is one of the few countries that have diplomatic relations with North Korea .
2.VP # VP :: 是 与 北韩 有 邦交 的 少数 国家 之一 # is one of the few countries that have diplomatic relations with North Korea
3.NP # NP :: 与 北韩 有 邦交 的 少数 国家 之一 # one of the few countries that have diplomatic relations with North Korea
4.VP # VP :: 与 北韩 有 邦交 # have diplomatic relations with North Korea
5.NP # NP :: 邦交 # diplomatic relations
6.NP # NP :: 北韩 # North Korea
7.NP # NP :: 澳洲 # Australia

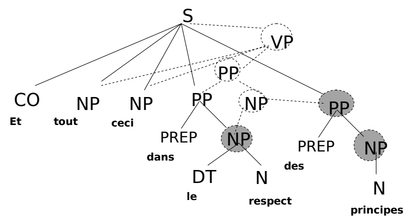
Recent Improvements

- The **Tree-to-Tree** (T2T) method is high precision but suffers from low recall
- Alternative: **Tree-to-String** (T2S) methods (i.e. [Galley et al., 2006]) use trees on ONE side and project the nodes based on word alignments
  - High recall, but lower precision
- Recent work by Vamshi Ambati [Ambati and Lavie, 2008]: combine both methods (**T2T***) by seeding with the T2T correspondences and then adding in additional consistent projected nodes from the T2S method
  - Can be viewed as restructuring target tree to be maximally isomorphic to source tree
  - Produces richer and more accurate syntactic phrase tables that improve translation quality (versus T2T and T2S)

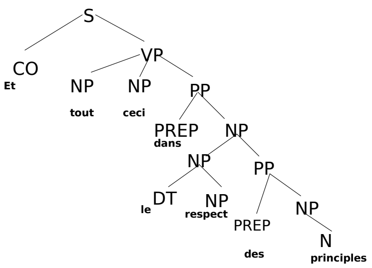
TnS vs TnT Comparison French-English

TYPE	Total	TnS	%	TnT	%	O%
ADJP	600104	412250	68.6	176677	29.4	90.7
ADVP	1010307	696106	68.9	106532	10.5	83.1
NP	11204763	8377739	74.7	4152363	37.1	93.8
VP	4650093	2918628	62.7	238659	5.1	67.9
PP	3772634	2766654	73.3	842308	22.3	89.4
S	2233075	1506832	67.4	248281	11.1	94.5
SBAR	912240	591755	64.8	42407	4.6	91.9
SBARQ	19935	9084	45.5	7576	38	99.6





- Add consistent projected nodes from source tree
- Tree Restructuring:
  - Drop links to a higher parent in the tree in favor of a lower parent
  - In case of a tie, prefer a node projected or aligned over an unaligned node



T*: Restructured target tree

Extracted Syntactic Phrases

English	French
The principles	Principes
With the principles	Principes
Accordance with the..	Respect des principes
Accordance	Respect
In accordance with the…	Dans le respect des principes
Is all in accordance with..	Tout ceci dans le respect…
This	et

TnS

English	French
The principles	Principes
With the principles	des Principes
Accordance	Respect

TnT

English	French
The principles	Principes
With the principles	des Principes
Accordance with the..	Respect des principes
Accordance	Respect
In accordance with the…	Dans le respect des principes
Is all in accordance with..	Tout ceci dans le respect…
This	et

TnT*

Comparative Results French-to-English

	Dev-Set	Test-Set	
System	BLEU	BLEU	METEOR
Xfer-TnS	26.57	27.02	57.68
Xfer-TnT	21.75	22.23	54.05
Xfer-TnT''	27.34	27.76	57.82
Xfer-Moses	29.54	30.18	58.13

- MT Experimental Setup
  - Dev Set: 600 sents, WMT 2006 data, 1 reference
  - Test Set: 2000 sents, WMT 2007 data, 1 reference
  - NO transfer rules, Stat-XFER monotonic decoder
  - SALM Language Model (430M words)

Combining Syntactic and Standard Phrase Tables

- Recent work by Greg Hanneman, Alok Parlikar and Vamshi Ambati
- Syntax-based phrase tables are still significantly lower in coverage than "standard" heuristic-based phrase extraction used in Statistical MT
- Can we combine the two approaches and obtain superior results?
- Experimenting with two main combination methods:
  - Direct Combination: Extract phrases using both approaches and then jointly score (assign MLE probabilities) them
  - Prioritized Combination: For source phrases that are syntactic – use the syntax-extracted method, for non-syntactic source phrases - take them from the "standard" extraction method
- Direct Combination appears to be slightly better so far
- Grammar builds upon syntactic phrases, decoder uses both

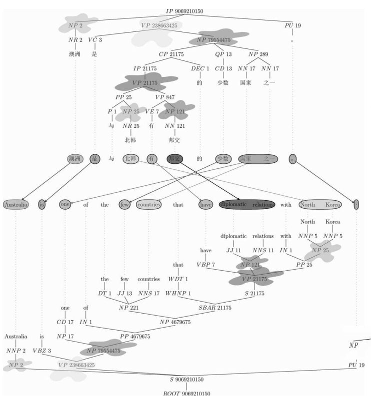
Recent Comparative Results French-to-English

Condition	BLEU	METEOR
Syntax Phrases Only	27.34	56.54
Non-syntax Phrases Only	30.18	58.35
Syntax Prioritized	29.61	58.00
Direct Combination	30.08	58.35

- MT Experimental Setup
  - Dev Set: 600 sents, WMT 2006 data, 1 reference
  - Test Set: 2000 sents, WMT 2007 data, 1 reference
  - NO transfer rules, Stat-XFER monotonic decoder
  - SALM Language Model (430M words)

Transfer Rule Learning

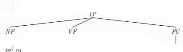
- Input: Constituent-aligned parallel trees
- Idea: Aligned nodes act as possible decomposition points of the parallel trees
  - The sub-trees of any aligned pair of nodes can be broken apart at any lower-level aligned nodes, creating an inventory of "treelet" correspondences
  - Synchronous "treelets" can be converted into synchronous rules
- Algorithm:
  - Find all possible treelet decompositions from the node aligned trees
  - "Flatten" the treelets into synchronous CFG rules

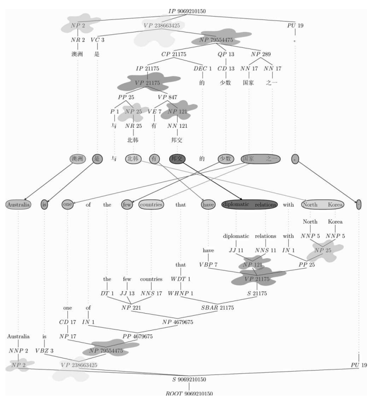


Rule Extraction Algorithm

Sub-Treelet extraction:

Extract Sub-tree segments including synchronous alignment information in the target tree. All the sub-trees and the super-tree are extracted.





Rule Extraction Algorithm

Flat Rule Creation:

Each of the treelets pairs is flattened to create a Rule in the 'Stat-XFER Formalism' –

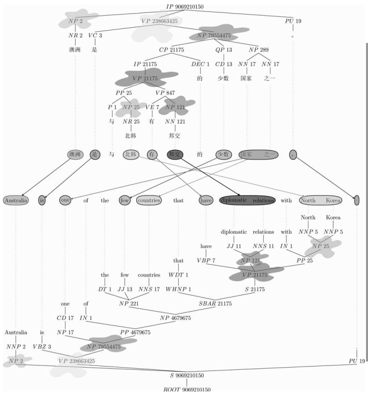
Four major parts to the rule:

1. Type of the rule: Source and Target side type information

2. Constituent sequence of the synchronous flat rule

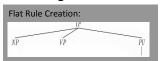
3. Alignment information of the constituents

4. Constraints in the rule (Currently not extracted)



Rule Extraction Algorithm

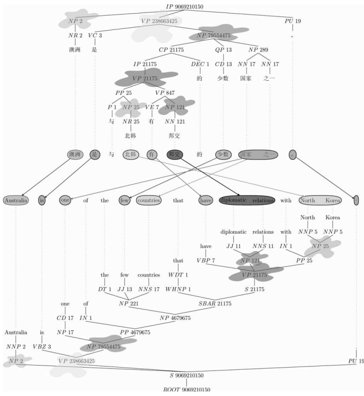
Flat Rule Creation:



Sample rule:

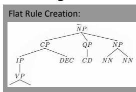
```
IP::S [ NP VP .] -> [NP VP .]
(
;; Alignments
(X1::Y1)
(X2::Y2)
;;Constraints
)
```

43



Rule Extraction Algorithm

Flat Rule Creation:

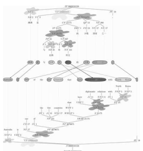


Sample rule:

```
NP::NP [VP 北 CD 有 邦交 ] -> [one
    of the CD countries that VP]
(
;; Alignments
(X1::Y7)
(X3::Y4)
)
```

Note:
1. Any one-to-one aligned words are elevated to Part-Of-Speech in flat rule.
2. Any non-aligned words on either source or target side remain lexicalized

44



Rule Extraction Algorithm

All rules extracted:

```
NP::NP [VP 北 CD 有 邦交 ] -> [one of the CD countries that VP]
(
(*score* 0.5)
;; Alignments
(X1::Y7)
(X3::Y4)
)

IP::S [ NP VP ] -> [NP VP ]
(
(*score* 0.5)
;; Alignments
(X1::Y1)
(X2::Y2)
)

NP::NP [ “北韩”] -> [“North” “Korea”]
(
;Many to one alignment is a phrase
)
```

All rules extracted:

```
VP::VP [VC NP] -> [VBZ NP]
(
(*score* 0.5)
;; Alignments
(X1::Y1)
(X2::Y2)
)

VP::VP [VC NP] -> [VBZ NP]
(
(*score* 0.5)
;; Alignments
(X1::Y1)
(X2::Y2)
)

NP::NP [NR] -> [NNP]
(
(*score* 0.5)
;; Alignments
(X1::Y1)
(X2::Y2)
)
VP::VP [北 NP VE NP] -> [ VBP NP with NP]
(
(*score* 0.5)
;; Alignments
(X2::Y4)
(X3::Y1)
(X4::Y2)
)
```

45

French-English System

- Large-scale broad-coverage system, developed for research experimentation
- Participated in WMT-08 and WMT-09 Evaluations
- Latest version integrates our most up-to-date processing methods:
  – French and English parsing using Berkeley Parser
  – Moses phrase tables combined with syntactic phrase tables using syntax-prioritized method
  – Very small grammar (26 rules) selected from large extracted rule set

1/21/2009 Alon Lavie: Stat-XFER 46

French-English System Data Resources

- Europarl corpus v. 4:
  – European parliamentary proceedings
  – 1.43 million sentences (36 MW)
- News Commentary corpus:
  – Editorials, columns
  – 0.06 million sentences (1 MW)
- Giga-FrEn corpus, pre-release version:
  – Crawled Canadian, European websites in various domains
  – 8.60 million sentences (191 MW)
- TOTAL:
  – about 10M sentence pairs
  – 9.57M sentence pairs after cleaning and filtering

1/21/2009 Alon Lavie: Stat-XFER 47

French-English System Phrase Tables

- After complete phrase pair extraction, filtering and collapsing:
  – 424 million standard SMT phrases
  – 27 million syntactic phrases
- Combined in a syntax-prioritized combination

1/21/2009 Alon Lavie: Stat-XFER 48

French-English System Example Grammar Rules

```
{NP,5256912}
NP::NP [N "de" N ] -> [N N ]
(
        (*sgtrule* 0.736382560)
        (*tgsrule* 0.292253105)
;       (*freq* 232772)
        (X3::Y1)
        (X1::Y2)
)
```

```
{NP,5782420}
NP::NP [N ADJ ] -> [ADJ N ]
(
        (*sgtrule* 0.726698577)
        (*tgsrule* 0.628385699)
;       (*freq* 1279387)
        (X2::Y1)
        (X1::Y2)
)
```

```
{VP,2042518}
VP::VP ["ne" V "pas" VP ] -> [V "not" VP ]
(
        (*sgtrule* 0.97076900)
        (*tgsrule* 0.55735608)
;       (*freq* 45332)
        (X2::Y1)
        (X4::Y3)
)
```

English-French System Translation Example

```
SrcSent 1
L' extrême droite européenne est caractérisée par son racisme et son
utilisation de la question de l' immigration en tant que divergence politique .

1 0     The extreme right in Europe is characterised by its racism and use of
the immigration as a political difference .

Overall: -1105.41, Prob: -94.7024, Rules: -9.9594, RuleSGT: -14.8736,
RuleTGS: -9.9594, TransSGT: -75.2431, TransTGS: -34.7368, Frag: -0.20412,
Length: -0.0398972, Words: 24,20
SGT -1.61107 TGS -0.745873

( 0 3 "The extreme right" -188.191 "L' extrême droite"
        "(PHRS,14515871 'The extreme right')")
( 3 5 "in Europe is" -187.021 "européenne est"
        "(PHRS,113195218 'in Europe is')")
( 5 6 "characterised" -125.731 "caractérisée" "(VS,331391 'characterised')")
( 6 8 "by its" -118.707 "par son" "(PHRS,62116997 'by its')")
( 8 9 "racism" -101.507 "racisme" "(NS,300037 'racism')")
( 9 12 "and use" -176.864 "et son utilisation" "(PHRS,112444704 'and use')")
( 12 17 "of the" -192.468 "de la question de l'" "(PHRS,150075588 'of the')")
( 17 21 "immigration as a" -205.369 "immigration en tant que"
         "(PHRS,316257845 'immigration as a')")
( 21 23 "political difference" -209.152 "divergence politique "
         "(NP,5782420 (ADJS,29478 'political') (NS,158428 'difference') ) ")
( 23 24 "." -39.2412 "." "(PUNCTS,3074 '.')")
```

Current and Future Research Directions

- Automatic Transfer Rule Learning:
  - Under different scenarios:
    - From large volumes of automatically word-aligned "wild" parallel data, with parse trees on one or both sides
    - From manually word-aligned elicitation corpus
    - In the absence of morphology or POS annotated lexica
  - Compositionality and generalization
    - Granularity of constituent labels – what works best for MT?
    - Lexicalization of grammars
  - Identifying "good" rules from "bad" rules
  - Effective models for rule scoring for
    - Decoding: using scores at runtime
    - Pruning the large collections of learned rules
  - Learning Unification Constraints

Current and Future Research Directions

- Advanced Methods for Extracting and Combining Phrase Tables from Parallel Data:
  - Leveraging from both syntactic and non-syntactic extraction methods
  - Can we "syntactify" the non-syntactic phrases or apply grammar rules on them?
- Syntax-aware Word Alignment:
  - Current word alignments are naïve and unaware of syntactic information
  - Can we remove incorrect word alignments to improve the syntax-based phrase extraction?
  - Develop new syntax-aware word alignment methods

Current and Future Research Directions

- Syntax-based LMs:
  - Our syntax-based MT approach performs parsing and translation as integrated processes
  - Our translations come out with syntax trees attached to them
  - Add syntax-based LM features that can discriminate between good and bad trees, on both target and source sides!

Current and Future Research Directions

- Algorithms for XFER and Decoding
  - Integration and optimization of multiple features into search-based XFER parser
  - Complexity and efficiency improvements
  - Non-monotonicity issues (LM scores, unification constraints) and their consequences on search

Current and Future Research Directions

- Building Elicitation Corpora:
  - Feature Detection
  - Corpus Navigation
- Automatic Rule Refinement
- Translation for highly polysynthetic languages such as Mapudungun and Iñupiaq

Conclusions

- Stat-XFER is a promising general MT framework, suitable to a variety of MT scenarios and languages
- Provides a complete solution for building end-to-end MT systems from parallel data, akin to phrase-based SMT systems (training, tuning, runtime system)
- No open-source publically available toolkits, but extensive collaboration activities with other groups
- Complex but highly interesting set of open research issues

Questions?

Czech-to-English Translation:

MT Marathon 2009 Session Preview

Jonathan Clark
Greg Hanneman

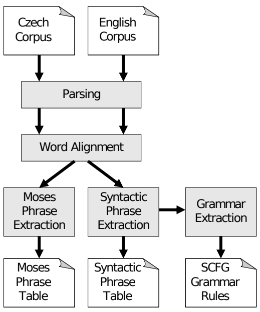
Language Technologies Institute
Carnegie Mellon University
26 January 2009

Outline

- Stat-XFER processing pipeline
- Processed Czech–English resources
- Possible workshop tasks
  - Syntactic phrase table combination methods
  - Synchronous grammar development
    - Selection of grammar rules
    - Exploration of label granularity
    - Development of manual grammars
  - Integration of morphological analysis

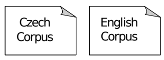
2

Stat-XFER Data Processing



3

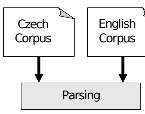
Stat-XFER Data Processing



- Corpus:
  - Project Syndicate news data: portion of CzEng corpus (84,141 sentences)

4

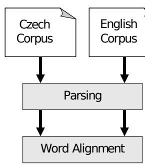
Stat-XFER Data Processing



- Parsing:
  - Czech dependency parses by TectoMT; converted to projective c-structure
  - English c-structure parses by Stanford parser

5

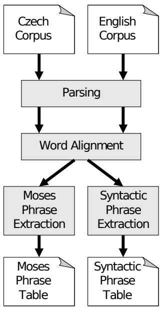
Stat-XFER Data Processing



- Word alignment:
  - GIZA++ grow-diag-final alignment done in advance on tokenized corpus
  - Alignments computed on full CzEng corpus of 8 million sentences

6

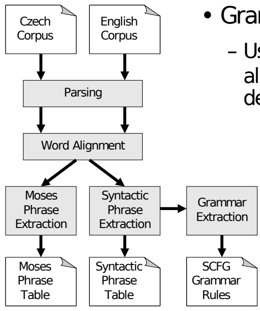
Stat-XFER Data Processing



- Phrase extraction:
  - Syntactic extraction by PFA node alignment algorithm, t2ts mode
  - Non-syntactic extraction with Moses package

7

Stat-XFER Data Processing



- Grammar extraction:
  - Using syntactic node alignments as tree decomposition points

8

Final Result

- Two phrase tables, with counts:

1	NNS	NNS	rozumem	brains
3	NN	NN	rozumem	reason
4	NN	NN	rozumem	sense
1	NP	NP	rozumem	reason
1	NN	NN	rozumností	wisdom
1	JJ	JJ	rozumnou	sensible
1	ADJP	ADJP	rozumnou měrou jisté	reasonably certain
1	NP	NP	rozumnou politiku	sensible policy

1	PHR	PHR	rozumem	brains
3	PHR	PHR	rozumem	reason
4	PHR	PHR	rozumem	sense
2	PHR	PHR	rozumem .	sense .
1	PHR	PHR	rozumem , a že	brains ; and that
1	PHR	PHR	rozumem , pokud	sense if
1	PHR	PHR	rozumem , pokud ne	sense if not

9

Final Result

- Three suffix-array language models
  - Target side of Project Syndicate corpus
  - … + more monolingual English data
  - … + target side of public CzEng corpus
- WMT tuning, development, and test sets
- = Baseline Stat-XFER system ready to analyse and expand

10

Outline

- Stat-XFER processing pipeline
- Processed Czech–English resources
- Possible workshop tasks
  - Syntactic phrase table combination methods
  - Synchronous grammar development
    - Selection of grammar rules
    - Exploration of label granularity
    - Development of manual grammars
  - Integration of morphological analysis

11

Phrase Table Combination

- Combination of non-syntactic and syntactic phrase pairs
  - Direct combination and syntax prioritization

12

Synchronous Grammars:
Rule Selection

- Rule learning yields huge grammars
- Decoding with millions of abstract rules is intractable
- Open Question: How do we select the best grammar rules with regard to translation quality and decoding speed?

13

Synchronous Grammars:
Label Granularity

- Rule learning assigns non-terminal and POS labels from input parse trees
- Input labels are believed appropriate…
  - For a given single language
  - According to a particular theory of grammar
- Open Question: How do we expand or collapse these labels so that they are appropriate for translating a particular language pair?

14

Synchronous Grammars:
Czech Example

- Subject moves in English translation
- Verbs in past tense cannot be associated with modifiers in present tense

Proti odmítnutí se zitra Petr
against dismissal AUX-REFL tomorrow Peter

v práci rozhodl protestovat
of work decided to protest

"Peter decided to protest against the dismissal of work tomorrow."

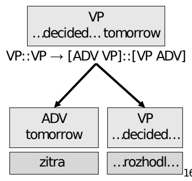
* Example from Bojar and Lopez, "Tree-based Translation," MT Marathon Presentation 2008

15

Synchronous Grammars:
Manual Grammar Writing

- Stat-XFER supports LFG-style unification
- Feature structures for unification can also be provided by the morphology server

```
VP::VP : [ADV VP] -> [VP ADV]
(
   (X1::Y2)
   (X2::Y1)
   (*tgsrule* 0.2)
   (*sgtrule* 0.6)
   ((X0 tense) = (X1 tense))
   ((X0 tense) = (X2 tense))
)
```



16

Czech Morphology: Example

- Czech words include clitics and inflectional morphology, marking meanings such as gender and number

nerozumím
ne+rozum+ím
NEG+understand+1SG
"I do not understand"

17

Czech Morphology in Stat-XFER

- Stat-XFER allows external morphology server to segment and annotate words at runtime
- Ambiguous word segmentations can be encoded as a lattice
- Must segment all training data, then rebuild phrase table & language model

18

(Your Idea Here)

- Any ideas about applying the statistical transfer framework to Czech–English translation are welcome!

19

TectoMT for Plaintext Freaks

Ondřej Bojar
bojar@ufal.mff.cuni.cz
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University, Prague

Outline

- Motivation: Large-scale rich NLP.
- Achievements: CzEng and Czech monolingual corpus parsed.
- HowTo: Which bits of TectoMT you need.
  – Caveats: Mind your NFS.
- Debugging someone else's code.
- Applications: Suggestions for the MT Marathon week.

Motivation

TectoMT is great:

- Bindings to many tools (taggers, parsers, aligners, . . .).
- Bindings *between* the tools.
- Easy to build pipelines.
- Easy to hack at various layers of NLP.

TectoMT was horrible:

- Rather verbose XML file format.
- Rather funny startup: init environment, then bash aliases to launch "Perl wrapped in btred" ⇒ pain to parallelize.
- Inevitable to debug someone else's code!

Achievements

Sun Grid Engine on 40 4-CPU computers.

We were able to annotate big Czech monolingual corpus:

Total sentences	**51.6 mil.**
Sentences with a t-tree	51.1 mil.
a-nodes, i.e. tokens	0.86 mld. (Gword)
t-nodes	0.60 mld. (G)
files	$>$ 1 mil.
disk space in tree format (.tmt.gz)	72GB
disk space in tab-delimited rich export (.txt.gz)	17GB

Data sources: Czech National Corpus 73%, Web Collection 17%, WMT09 Monolingual Training Data 10%

We also parsed and aligned CzEng (Bojar et al., 2008a), an extended version of 7 million Czech-English parallel sentences.

HowTo: Plaintext to TMT

TectoMT's file format is called TMT:

- XML, an application of PML (Pajas and Štěpánek, 2005).

⇒The first step needed is to wrap plaintext with XML tags.

```
...
<LM id='news-dev2009a-00-s8'>
  <english_source_sentence> Government crisis coming , says Gallup...
  <czech_source_sentence> Gallup vidí vládní krizi</czech_source_s...
</LM>
...
```

E.g. `tools/format_convertors/czeng07_to_tmt/czeng07_to_tmt.pl`.

- Avoid $>$ 50 to 100 sentences in a file.
- Avoid $>$ 1000 files in a directory.

⇒ Clever convertors create nested directory structure.

HowTo: Scenarios on Grid

1. Create filelist: `find dir -name '*.tmt.gz' > filelist`
2. Submit parallel execution of a TectoMT scenario:

```
tools/cluster_utils/qrunblocks \
  filelist \
  "Miscel::SuicideIfMemFull Miscel::SuicideIfDiskFull Block1 Block2 ..." \
  --jobs 20 --attempts 200 \
  --finished-contains "SCzechT"
```

- Suicides protect your environment.
- `--attempts` restart your jobs after suicides or random deaths.
- `--finished-contains` skips files that seem to contain the desired bit.
- Jobs run independently in the background.
- Independent log files (contain stdout).

HowTo: Escape the Devillish XML

Avoid parsing XML yourself, make use of TectoMT API for reading.

1. Implement a simple block to print information to stdout.
2. Submit parallel printing, e.g.:

```
tools/cluster_utils/qrunblocks \
  filelist \
  "Print::Factored" \
  --jobs 20 --no-save \
  --join \
  > joined_output
```

- `--no-save` avoids saving TMT files,
- `--join` waits for all the jobs to succeed and joins their stdouts preserving file order.

Caveats: NFS is the Bottleneck

`qrunblocks` simply splits the filelist and submits the jobs.
⇒ too many jobs accessing the same NFS server cause delays.

Current workarounds:

- Reduce the number of jobs.
- Spread your files to many NFS servers, e.g.:
  `/net/cluster/COMPUTER/tmp/` for various computers
  ⇒ inefficient processing of non-local files.

Ultimate solution:

- Know which files are local to a node.
- Submit jobs only to nodes with unfinished files.
- Jobs themselves figure out which (local) files need to be processed.

Debugging Someone Else's Code

- Your particular data may crash some of the TectoMT blocks.
- Debugging with huge datasets is slow or impossible.
- Need to send a small bug report if unable to fix the bug yourself.

1. Find one of the problematic files (e.g. study `qrunblocks` logs).
2. Apply auto-diagnose:
   ```
   $TMT_ROOT/tools/tests/auto_diagnose.pl --cleanup \
     file.tmt.gz targetdir 'block1 block2'
   ```
3. Run the test as instructed:
   `./targetdir/test.sh`
   Or simply send the targetdir to the assumed author.

Auto-diagnose finds the first crashing sentence, the first crashing block from the scenario, and construct a TMT file with just the sentence. The `test.sh` is just the command line to run the minimized test.

Suggested Applications

NLP hacking:

- Remove useless case markings, insert fake articles and preps:
  English $\xrightarrow{\text{Perl}}$ Czenglish $\xrightarrow{\text{ISI ReWrite}}$ English (Cuřín, 2006)
- Move verbs to the end of the clause:
  English $\xrightarrow{\text{TectoMT}}$ Hinglish $\xrightarrow{\text{Moses}}$ Hindi (Bojar et al., 2008b)
  We needed ˜230 lines of code, SVO→SOV alone is 12 lines.
- Truecasing based on names as marked by a lemmatizer/NER.

Feature fishing: Rich features for your favourite MT:

- Highlight non-local information, e.g. subject-verb agreement:
  Cat... talked → *... talked+sg* vs. *Cats... talked* → *... talked+pl*

More details in Thursday and Friday lectures.

Summary

- TectoMT *can* be used on large data.
- Debugging is just a regular nightmare, not worse.

Suggested workflow for your TectoMT Project at Marathon:

1. Get a brilliant idea, find friends.
2. Adapt `tools/format_convertors` to load your input.
3. Setup your annotation scenario.
   - Add your own blocks for NLP hacking.
4. Use `qrunblocks` to annotate huge data.
5. Export to plaintext.
6. Train/apply/test your favourite MT system.

References

Ondřej Bojar, Miroslav Janíček, Zdeněk Žabokrtský, Pavel Češka, and Peter Beňa. 2008a. CzEng 0.7: Parallel Corpus with Community-Supplied Translations. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May. ELRA.

Ondřej Bojar, Pavel Straňák, and Daniel Zeman. 2008b. English-hindi translation in 21 days. In *Proceedings of the 6th International Conference On Natural Language Processing (ICON-2008) NLP Tools Contest*, Pune, India. NLP Association of India.

Jan Cuřín. 2006. *Statistical Methods in Czech-English Machine Translation*. Ph.D. thesis, ÚFAL, MFF UK, Prague, Czech Republic.

Petr Pajas and Jan Štěpánek. 2005. A Generic XML-Based Format for Structured Linguistic Annotation and Its Application to Prague DependencyTreebank 2.0. Technical Report TR-2005-29, ÚFAL MFF UK, Prague, Czech Rep.

Winter School

Day 2: Word-based models and the EM algorithm

MT Marathon

27 Jan 2009

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

1	2	3	4
das	Haus	ist	klein
the	house	is	small
1	2	3	4

- Word *positions* are numbered 1–4

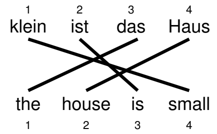
Alignment function

- 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 $a : i \to j$
- Example

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

Reordering

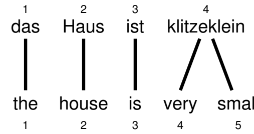
- Words may be **reordered** during translation



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

One-to-many translation

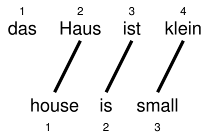
- A source word may translate into **multiple** target words



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

Dropping words

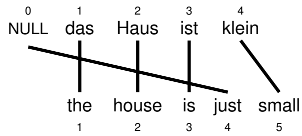
- Words may be **dropped** when translated
  - The German article *das* is dropped



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

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



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

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 $\mathbf{f} = (f_1, ..., f_{l_f})$ of length l_f
  - to an English sentence $\mathbf{e} = (e_1, ..., e_{l_e})$ of length l_e
  - with an alignment of each English word e_j to a foreign word f_i according to the alignment function $a : j \to i$

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

  - parameter ϵ is a *normalization constant*

Example

das

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

Haus

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

ist

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

klein

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

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

12

Learning lexical translation models

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

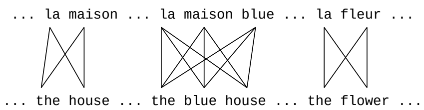
13

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

14

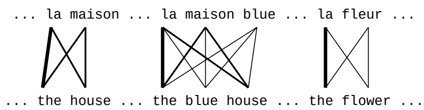
EM algorithm



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

15

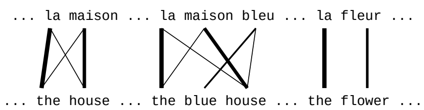
EM algorithm



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

16

EM algorithm



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

17

EM algorithm

- Convergence
- Inherent hidden structure revealed by EM

EM algorithm

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

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

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

- Parameter estimation from 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: estimate translation probabilities from weighted counts

IBM Model 1 and EM

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

la–the, maison–house	la–house, maison–the	la–the, maison–the	la–house, maison–house
$p(\mathbf{e}, a|\mathbf{f}) = 0.56$	$p(\mathbf{e}, a|\mathbf{f}) = 0.035$	$p(\mathbf{e}, a|\mathbf{f}) = 0.08$	$p(\mathbf{e}, a|\mathbf{f}) = 0.005$
$p(a|\mathbf{e}, \mathbf{f}) = 0.824$	$p(a|\mathbf{e}, \mathbf{f}) = 0.052$	$p(a|\mathbf{e}, \mathbf{f}) = 0.118$	$p(a|\mathbf{e}, \mathbf{f}) = 0.007$

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

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

IBM Model 1 and EM: Expectation Step

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

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

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

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

IBM Model 1 and EM: Expectation Step

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

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

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

- Note the trick in the last line
  - removes the need for an *exponential* number of products
  - → this makes IBM Model 1 estimation **tractable**

The trick

(case $l_e = l_f = 2$)

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

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

IBM Model 1 and EM: Expectation Step

- Combine what we have:

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

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

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

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:

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

- Using the expression on the previous slide, and noting that only alignments which link e and f are relevant, we obtain:

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

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

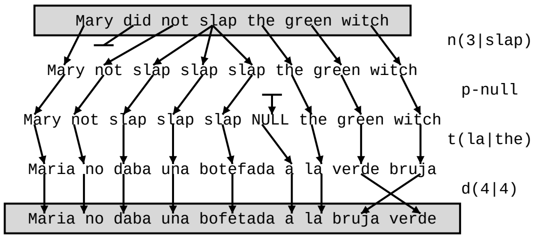
```
initialize t(e|f) uniformly
do until convergence
  set count(e|f) to 0 for all e,f
  set total(f) to 0 for all f
  for all sentence pairs (e_s,f_s)
    for all words e in e_s
      total_s(e) = 0
      for all words f in f_s
        total_s(e) += t(e|f)
    for all words e in e_s
      for all words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

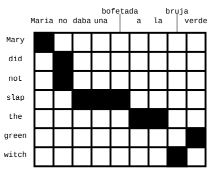
- Only IBM Model 1 has *global maximum*
  - training of a higher IBM model builds on previous model
- Compuationally biggest change in Model 3
  - trick to simplify estimation does not work anymore
  - → *exhaustive* count collection becomes computationally too expensive
  - **sampling** over high probability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

IBM Model 4



Word alignment

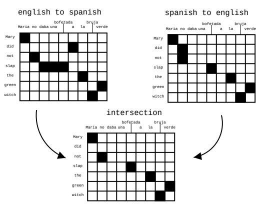
- IBM Models are nowadays mainly used for word alignment
- Other word alignment models proposed e.g. HMM
- Shared task at NAACL 2003 and ACL 2005 workshops



Word alignment with IBM models

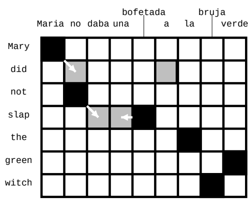
- IBM Models create a *many-to-one* mapping
  - words are aligned using an **alignment function**
  - a function may return the same value for different input (one-to-many mapping)
  - a function can not return multiple values for one input (*no many-to-one* mapping)
- But we need *many-to-many* mappings

Symmetrizing word alignments



- *Intersection* of GIZA++ bidirectional alignments

Symmetrizing word alignments



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

Growing heuristic

```
GROW-DIAG-FINAL-AND(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-AND(e2f); FINAL-AND(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 or f-new not aligned ) and
                 ( e-new, f-new ) in union( e2f, f2e ) )
              add alignment point ( e-new, f-new )
FINAL-AND(a):
  for english word e-new = 0 ... en
    for foreign word f-new = 0 ... fn
      if ( ( e-new not aligned and 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
  - e.g. Liang, Taskar and Klein, NAACL 2006
- Discriminative training methods
  - supervised learning based on labeled data
  - semi-supervised learning with limited labeled data
  - e.g. Blunsom and Cohn, ACL 2006
- Better generative models
  - e.g. Fraser and Marcu, EMNLP 2007

Winter School

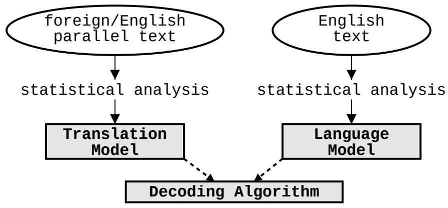
Day 3: Decoding / Phrase-based models

MT Marathon

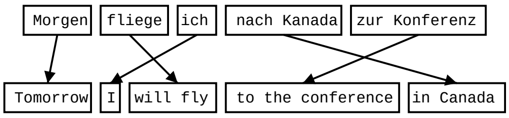
28 January 2009

Statistical Machine Translation

- Components: Translation model, language model, decoder



Phrase-Based Translation



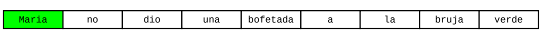
- Foreign input is segmented in phrases
  - any sequence of words, not necessarily linguistically motivated
- Each phrase is translated into English
- Phrases are reordered

Phrase Translation Table

- Phrase Translations for "den Vorschlag":

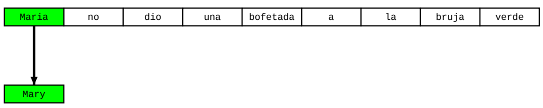
English	ϕ**(e|f)**	**English**	ϕ**(e|f)**
the proposal	0.6227	the suggestions	0.0114
's proposal	0.1068	the proposed	0.0114
a proposal	0.0341	the motion	0.0091
the idea	0.0250	the idea of	0.0091
this proposal	0.0227	the proposal ,	0.0068
proposal	0.0205	its proposal	0.0068
of the proposal	0.0159	it	0.0068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

Decoding Process



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

Decoding Process



- 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

Maria	no	dio una bofetada	a	la	bruja	verde

Mary	did not	slap

- Many to one translation

Decoding Process

Maria	no	dio una bofetada	a la	bruja	verde

Mary	did not	slap	the

- *Many to one* translation

Decoding Process

Maria	no	dio una bofetada	a la	bruja	verde

Mary	did not	slap	the	green

- *Reordering*

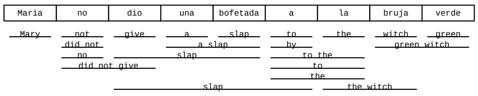
Decoding Process

Maria	no	dio una bofetada	a la	bruja	verde

Mary	did not	slap	the	green	witch

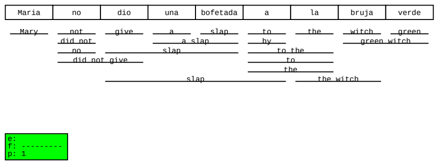
- Translation *finished*

Translation Options



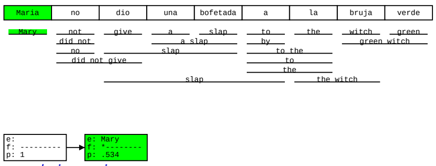
- Look up *possible phrase translations*
  - many different ways to *segment* words into phrases
  - many different ways to *translate* each phrase

Hypothesis Expansion



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

Hypothesis Expansion

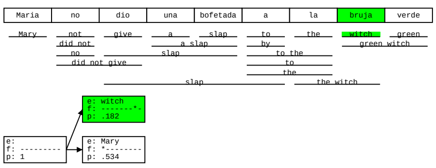


- Pick *translation option*
- Create *hypothesis*
  - e: add English phrase `Mary`
  - f: first foreign word covered
  - p: probability 0.534

A Quick Word on Probabilities

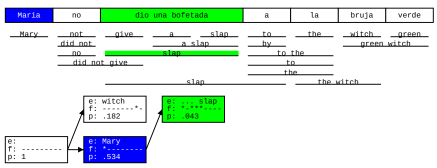
- Not going into detail here, but...
- *Translation Model*
  - phrase translation probability p(Mary|Maria)
  - reordering costs
  - phrase/word count costs
  - ...
- *Language Model*
  - uses trigrams:
  - p(Mary did not) =
    p(Mary|START) $\times p$(did|Mary,START) $\times$ p(not|Mary did)

Hypothesis Expansion



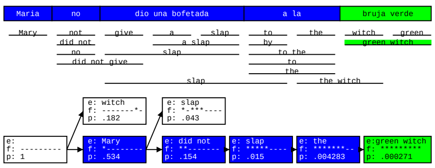
- Add another *hypothesis*

Hypothesis Expansion



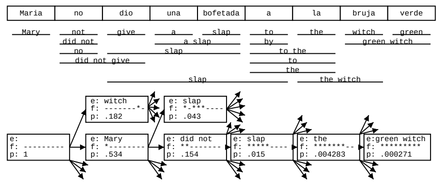
- Further *hypothesis expansion*

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

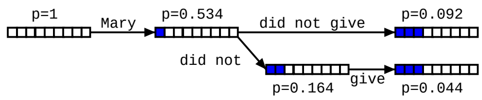
Explosion of Search Space

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

⇒ Decoding is NP-complete [Knight, 1999]

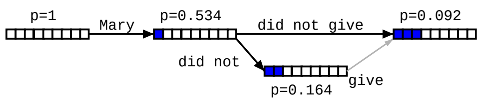
⇒ Need to *reduce search space*
  - risk free: hypothesis **recombination**
  - risky: **histogram/threshold pruning**

Hypothesis Recombination



- Different paths to the *same* partial translation

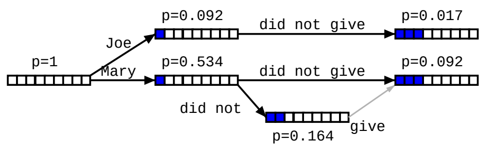
Hypothesis Recombination



- Different paths to the same partial translation

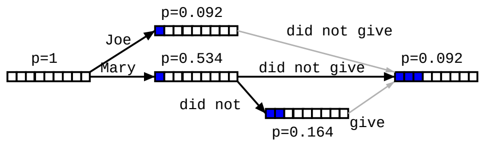
⇒ *Combine paths*
  - *drop weaker* path
  - keep pointer from weaker path (for lattice generation)

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*

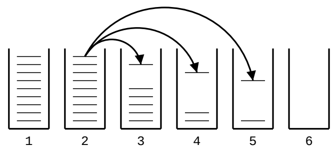
Pruning

- 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., α = 0.001)

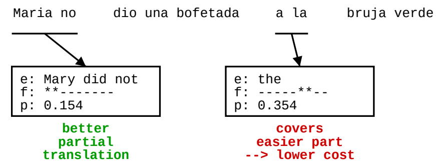
Hypothesis Stacks



- 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

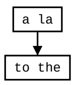
- Comparing hypotheses with *same number of foreign words* covered



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

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

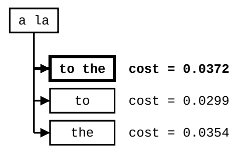
Future Cost Estimation



- *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

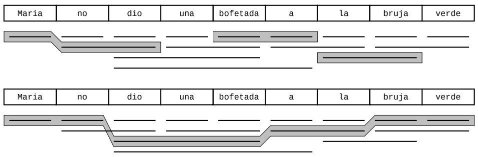
  → LM * TM = p(to) * p(the|to) * p(to the|a la)

Future Cost Estimation: Step 2



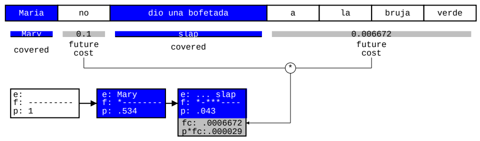
- Step 2: find *cheapest cost* among translation options

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

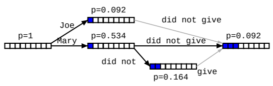
A* search

- 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

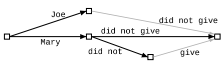
Limits on Reordering

- 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**:

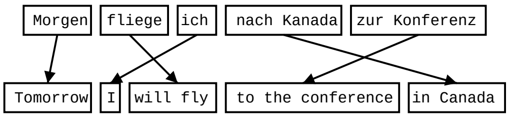
```
Translation ||| Reordering LM TM WordPenalty ||| Score
this is a small house ||| 0 -27.0908 -1.83258 -5 ||| -28.9234
this is a little house ||| 0 -28.1791 -1.83258 -5 ||| -30.0117
it is a small house ||| 0 -27.108 -3.21888 -5 ||| -30.3268
it is a little house ||| 0 -28.1963 -3.21888 -5 ||| -31.4152
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

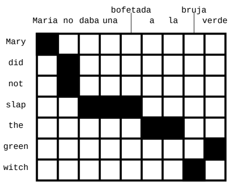
Phrase translation table

- Phrase translations for *den Vorschlag*

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

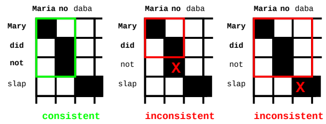
How to learn the phrase translation table?

- Start with the *word alignment*:



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

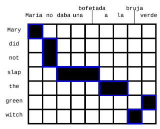
Consistent with word alignment



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

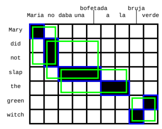
$$(\overline{e},\overline{f}) \in BP \Leftrightarrow \quad \forall e_i \in \overline{e} : (e_i, f_j) \in A \rightarrow f_j \in \overline{f}$$
$$\text{AND} \quad \forall f_j \in \overline{f} : (e_i, f_j) \in A \rightarrow e_i \in \overline{e}$$

Word alignment induced phrases



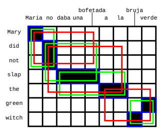
(Maria, Mary), (no, did not), (slap, daba una bofetada), (a la, the), (bruja, witch), (verde, green)

Word alignment induced phrases



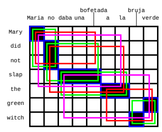
(Maria, Mary), (no, did not), (slap, daba una bofetada), (a la, the), (bruja, witch), (verde, green),
(Maria no, Mary did not), (no daba una bofetada, did not slap), (daba una bofetada a la, slap the),
(bruja verde, green witch)

Word alignment induced phrases



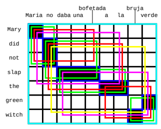
(Maria, Mary), (no, did not), (slap, daba una bofetada), (a la, the), (bruja, witch), (verde, green),
(Maria no, Mary did not), (no daba una bofetada, did not slap), (daba una bofetada a la, slap the),
(bruja verde, green witch), (Maria no daba una bofetada, Mary did not slap),
(no daba una bofetada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

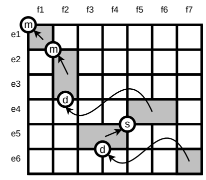
⇒ Possible *choices*
  - *relative frequency* of collected phrases: $\phi(\overline{f}|\overline{e}) = \frac{\text{count}(\overline{f},\overline{e})}{\sum_{\overline{f}} \text{count}(\overline{f},\overline{e})}$
  - or, conversely $\phi(\overline{e}|\overline{f})$
  - use *lexical translation probabilities*

Reordering

- *Monotone* translation
  - do not allow any reordering

  → 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

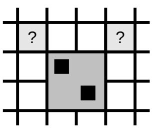
Lexicalized reordering models



[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]

ÚFAL

TectoMT

Software framework for developing
MT systems (and other NLP applications)

Zdeněk Žabokrtský
ÚFAL MFF UK

1/36

Outline

- Part I – Introduction
  - What is TectoMT
  - Motivation
- Part II - TectoMT System Architecture
  - Data structures
  - Processing units: blocks, scenarios, applications
- Part III – Applications implemented in TectoMT

2/36

What is TectoMT

- TectoMT is …
  - a highly modular extendable NLP software system
  - composed of numerous (mostly previously existing) NLP tools integrated into a uniform infrastructure
  - aimed at (not limited to) developing MT system
- TectoMT is not …
  - a specific method of MT (even if some approaches can profit from its existence more than others)
  - an end-user application (even if releasing of single-purpose stand-alone applications is possible and technically supported)

3/36

Motivation for creating TectoMT

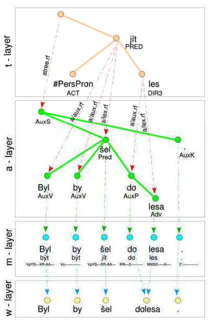
- First, technical reasons:
  - Want to make use of more than two NLP tools in your experiment? Be ready for endless data conversions, need for other people's source code tweaking, incompatibility of source code and model versions…
  - Unified software infrastructure might help us.
- Second, our long-term MT plan:
  - We believe that tectogrammar (deep syntax) as implemented in Prague Dependency Treebank might help to (1) **reduce data sparseness**, and (2) find and **employ structural similarities** revealed by tectogrammar even between typologically different languages.

4/36

Prague Dependency Treebank 2.0

- three layers of annotation:
- tectogrammatical layer
  - deep-syntactic dependency tree
- analytical layer
  - surface-syntactic dependency tree
  - 1 word (or punct.) ~ 1 node
- morphological layer
  - sequence of tokens with their lemmas and morphological tags

[Ex: *He would have gone into forest*]



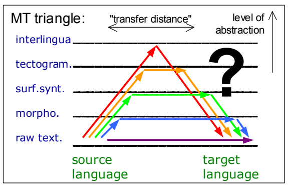
Tectogrammar in a nutshell

- tectogrammatical layer of language representation
  - introduced by Petr Sgall in 1960's, implemented in PDT 2.0
- key features:
  - each sentence represented as a **deep-syntactic dependency tree**
  - **functional words** (such as aux.verb, prepositions, subordinating conjunctions) accompanying an autosemantic word "**collapse**" with it into a single t-node, labeled with the autosemantic t-lemma
  - "added" nodes (e.g. because of pro-dropped subjects)
  - semantically indispensable syntactic and morphological knowledge represented as attributes of nodes
  - **economy**: no nonterminals, less nodes than words in the original sentence, decreased morphological redundancy (categories imposed by agreement disappear), etc.

6/36

MT triangle in terms of PDT

- Key question: what is the optimal level of abstraction?

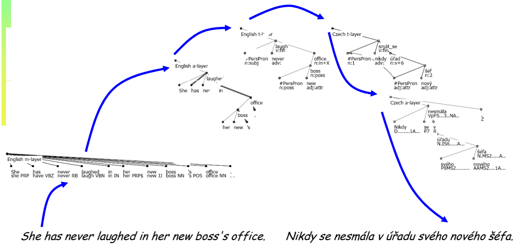


- Obvious trade-off: ease of transfer vs. additional analysis and synthesis costs (system complexity, errors...)

7/36

MT triangle *in vivo*

- Illustration: analysis-transfer-synthesis in TectoMT



8/36

How could tecto help?

- Vague assumption:
  - tectogrammatics **abstracts from several language-specific characteristics** (e.g. makes no difference between meanings expressed by isolated words, inflection or agglutination)
  - ...therefore **languages look more similar** at the tecto-layer
  - ...therefore the **transfer phase should be easier** (compared to the operation on raw sequences of word forms)
- Yes, but how exactly could it help?

9/36

How could tecto help? (cont.)

- n-gram view:
  - manifestations of lexemes are mixed with manifestations of language means expressing the relations between the lexemes and of other grammar rules
    - inflectional endings, agglutinative affixes, functional words, word order, punctuation orthographic rules ...
    - *It will be delivered to Mr. Green's assistants at the nearest meeting.*
  - → **training data sparsity**
- tectogrammar view:
  - clear separation of meaningful "signs" from "signs" which are only imposed by grammar (e.g. imposed by agreement)
  - clear separation of lexical, syntactical and morphological meaning components
  - → modularization of the translation task → **potential for a better structuring of statistical models** → more effective exploatation of the (limited) training data

10/36

Tecto transfer factorization

- Three transfer "channels" can be separated:
  - translation of lexicalization
    - E.g. 'koupit' goes to 'buy'
  - translation of syntactization
    - e.g. relative clause goes to attributive adjective
  - Translation of morphological meanings
    - e.g. singular goes to singular
- The channels are relatively loosely coupled (esp. the third one) which could be used for smoothing.

11/36

Tecto transfer factorization (cont.)

- Example: three ways to express future tense in Czech
  - (1) aux.verb: ***budu*** *... chodit – I will walk ...*
  - (2) prefix: ***po****letím – I will fly ...*
  - (3) ending: *uva****řím*** *– I will boil ...*
- nontrivial tense translation from the n-gram view
- but once we work with tecto analyses, we can translate the future tense just to future tense, separately from translating the lemma
  - similarly, plural goes mostly to plural, comparative to comparative, etc.

12/36

Tecto transfer factorization (cont.)

- we introduce the notion of **formemes** - morphosyntactic language means expressing the dependency relation
- example values:
  - **n:v+6** (in Czech) = semantic noun which is on the surface expressed in the form of prepositional group in locative with preposition "v"
  - **v:that+fin/a** (in English) = semantic verb expressed in active voice as a head of subordinating clause introduced with the sub.conjunction "that"
  - **v:rc** (in Czech and English) = head of relative clause
  - **n:sb** (in English) = noun in subject position
  - **n:1** (in Czech) = noun in nominative case
  - **adj:attr** (in Czech and English) = adjective in attributive position
- formemes allow us to introduce a **separate syntactization factor** and to train it using a parsed parallel corpus
- trained estimates of $P(F_{cz}|P_{en})$:

v:to+inf	v:inf	0.4817
v:to+inf	v:aby+fin	0.0950
v:to+inf	n:k+3	0.0702
v:to+inf	v:že+fin	0.0621
n:for+X	n:pro+4	0.2234
n:for+X	n:2	0.1669
n:for+X	n:4	0.0788
n:for+X	n:za+4	0.0775

13/36

Using tree context

- Hypothesis: translation choices are conditioned rather by governing/dependent words than by linear predecessors/followers
- syntactic dependency and linear adjacency often coincide, but long distance dependencies occur too
- long distance dependencies are notoriously difficult to handle by n-gram models

14/36

Using tree context (cont.)

- Example 1:
  - *The **grass** around your house should be **cut** soon.*
  - google trans.: ***Trávu** kolem vašeho domu by se měl **snížit** v nejbližší době.*
  - incorrect morphological choice with the subject; verb form is crucial for the correct choice, but it is too far
  - incorrect lexical choice of the verb; subject's lexical occupation could help, but it is too far
- Example 2
  - *Zítra se v kostele Svaté Trojice budou **brát** Marie a Honza.*
  - google trans: *Tomorrow is the Holy Trinity church will **take** Mary and John.*
  - Incorrect lexical choice: presence of the "se" clitic at the clause-second position is crucial, but it is too far

15/36

How could tecto help - summary

- Tectogrammar offers a natural **transfer factorization** into three relatively independent channels
- Tectogrammar offers **local tree context** (instead of only local linear context)

16/36

Hybrid MT with TectoMT

- other option: to combine translation based on tecto-transfer with a conventional phrase-based translation system X
- TectoMT can **provide** X with **additional hypotheses**
- TectoMT can be used for **decomposing input sentences** into smaller, relatively independently translatable chunks (e.g. finite clauses or even individual constituents)
- TectoMT can steel the lexical choices from X's output and **resynthetize** the sentence (or its parts) according to grammar rules, e.g. in order to correct agreement
- New features for reranking X's output hypotheses can be extracted from their syntactic analyses by TectoMT (e.g. by penalizing presence of abnormal tree configurations)

17/36

Hybrid MT with TectoMT (cont.)

- TectoMT can be used for making the source and target languages more similar even from the n-gram view:
  - Adding artificial tokens (e.g. inserting _det_ when translating to a language with determiners)
  - Joining tokens (e.g. of John -> of_John, when translating into a language using genitive ending instead of a functional word)
  - Regular grammar-based word order changes: e.g. shifting *ago* in front of the noun group (as it was a preposition) when translating from English to German

18/36

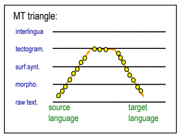
Part II:

TectoMT System Architecture

19/36

Design decisions

- **Linux + Perl**
- set of well-defined, **linguistically relevant layers** of language representation
- **neutral** w.r.t. chosen methodology ("rules vs. statistics")
- accent on modularity: translation **scenario** as a sequence of translation **blocks** (modules corresponding to individual NLP subtasks)
  - reusability
  - substitutability



20/36

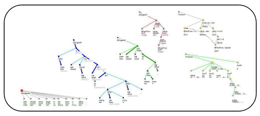
Design decisions (cont.)

- reuse of Prague Dependency Treebank technology (tools, XML-based format)
- in-house **object-oriented architecture** as the backbone
  - all tools communicate via standardized OO Perl interface
  - avoiding the former practice of tools communicating via files in specialized formats
- easy **incorporation of external tools**
  - previously existing parsers, taggers, lemmatizers etc.
  - just provide them with a Perl "wrapper" with the prescribed interface

21/36

Hierarchy of data-structure units

- **document**
  - the smallest independently storable unit (~ xml file)
  - represents a text as a sequence of bundles, each representing one sentence (or sentence tuples in the case of parallel documents)
- **bundle**
  - set of tree representations of a given sentence
- **tree**
  - representation of a sentence on a given layer of linguistic description
- **node**
- **attribute**
  - document's, node's, or bundle's attrname-value pair



22/36

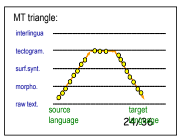
Layers of sentence description

- in each bundle, there can be at most one tree for each "layer"
- set of possible layers = {S,T} x {English,Czech,...} x {M,P,A,T,N}
  - S - source, T-target
  - M - morphological analysis
  - P - phrase-structure tree
  - A - analytical tree
  - T - tectogrammatical tree
  - N - instances of named entities
- Example: SEnglishA - tectogrammatical analysis of an English sentence on the source-language side

23/36

Hierarchy of processing units

- **block**
  - the smallest individually executable unit
  - with well-defined input and output
  - block parametrization possible (e.g. model size choice)
- **scenario**
  - sequence of blocks, applied one after another on given documents
- **application**
  - typically 3 steps:
    - 1. conversion from the input format
    - 2. applying the scenario on the data
    - 3. conversion into the output format



24/36

Blocks

- technically, Perl classes derived from TectoMT::Block
- either method process_bundle (if sentences are processed independently) or method process_document must be defined
- more than 200 blocks in TectoMT now, for various purposes:
  - blocks for analysis/transfer/synthesis, e.g.

    ```
    SEnglishW_to_SEnglishM::Lemmatize_mtree
    SEnglishP_to_SEnglishA::Mark_heads
    TCzechT_to_TCzechA::Vocalize_prepositions
    ```
  - blocks for alignment, evaluation, feature extraction, etc.
- some of them only implement simple rules, some of them call complex probabilistic tools
- English-Czech tecto-based translation currently composes of roughly 80 blocks

25/36

Tools integrated as blocks

- to integrate a stand-alone NLP tool into TectoMT means to create a block that encapsulates the functionality of the tool behind the standardized block interface
- already integrated tools:
  - taggers
    - Hajič's tagger, Raab&Spoustová Morče tagger, Rathnaparkhi MXPOST tagger, Brants's TnT tager, Schmid's Tree tagger, Coburn's Lingua::EN::Tagger
  - parsers
    - Collins' phrase structure parser, McDonalds dependency parser, ZŽ's dependency parser
  - named-entity recognizer
    - Stanford Named Entity Recognizer, Kravalová's SVM-based NE recognizer
  - several other
    - Klimeš's semantic role labeller, ZŽ's C5-based afun labeller, Ptáček's C5-based Czech preposition vocalizer, ...

26/36

Other TectoMT components

- "core" - Perl libraries forming the core of TectoMT infrastructure, esp. for memory representation of (and interface to) to the data structures
- numerous file-format convertors (e.g. from PDT, Penn treebank, Czeng corpus, WMT shared task data etc. to our xml format)
- TectoMT-customized Pajas' tree editor TrEd
- tools for parallelized processing (Bojar)
- data, esp. trained models for the individual tools, morphological dictionaries, probabilistic translation dictionaries...
- tools for testing (regular daily tests), documentation...

27/36

TectoMT directory structure

- everything under one directory tree specified in system variable TMT_ROOT

- **versioned part** (in a svn repo)
  - install/
  - libs/{core,blocks,packaged,other}/
  - tools/
  - applications/
  - doc/
  - personal/
  - tools/
  - training/
  - release_building/
  - evaluation/
- **shared part** (unversioned)
  - share/installed_tools/
  - share/installed_libs/
  - share/data/{models, resources...}
  - share/tred/

28/36

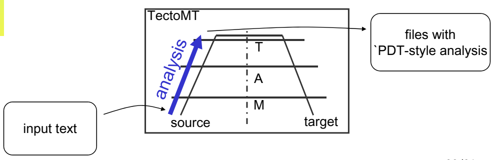
Part III:

Applications implemented in TectoMT

29/36

PDT-style layered analysis

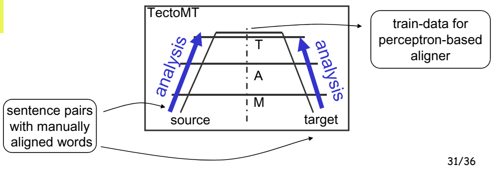
- analyze a given Czech or English text up to morphological, analytical and tectogrammatical layer
- used currently e.g. in experiments with intonation generation or information extraction



30/36

Training tecto-aligner

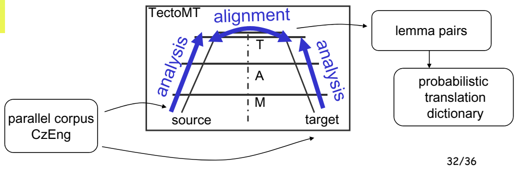
- data for training a perceptron-based aligner of tectogrammatical nodes, using manually sentence pairs aligned at the word layer
- the resulting aligner was used for aligning CzEng (parsed Czech-English parallel corpus, around 60MW)



31/36

Transl. dictionary extraction

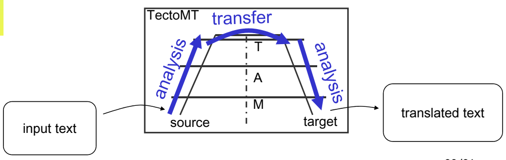
- using the lemma pairs from the aligned t-nodes from a huge parallel corpus, we build a probabilistic translation dictionary



32/36

Translation with tecto-transfer

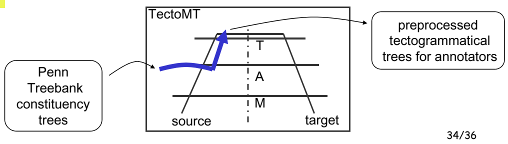
- analysis-transfer-synthesis translation from English to Czech and vice versa
- employed probabilistic dictionary from the previous slide



33/36

Preproc. data for PEDT

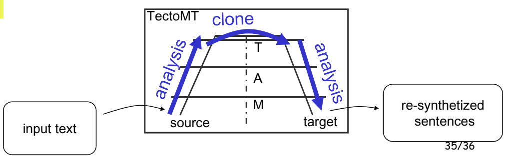
- Prague English Dependency Treebank
  - PDT-style annotation project at UFAL
  - currently 12000 English tectogrammatically analyzed sentences, since 2006, now 6 annotators, http://ufal.mff.cuni.cz/pedt
  - saving annotators' work by automatizing a part of the analysis in TectoMT



34/36

Sentence re-synthesis

- analysis-clone-synthesis scenario for
  - **postprocessing of other MT system's output** (to make it more grammatical)
  - **speech reconstruction** - postprocessing of STT's output (to make it more grammatical)
  - (useful also finding bugs anywhere along the scenario)
- very preliminary stage



35/36

Final remarks

- Our implementation of tectogrammar-based MT is still premature and does not reach state-of-the-art quality (WMT Shared Task 2009)
- However, having the TectoMT infrastructure and sharing its components already saves our work in several research directions.

36/36

Analysis and alignment of parallel data in TectoMT

David Mareček
marecek@ufal.mff.cuni.cz

MT Marathon 2009, January 26 - 30, Prague

Task and motivation

INPUT: set of English-Czech parallel sentences
OUTPUT: set of aligned tectogrammatical trees (+ lower layers)

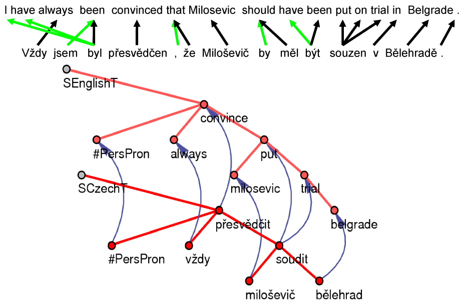
Advantage of tectogrammatical alignment over word alignment:

- Functional words (e.g. articles, prepositions, auxiliary verb 'be', modal verbs …), that are often problematic to align (they can have different functions in different languages), don't have their own node in the tectogrammatical layer – we needn't align them.
- The tree structure may help.

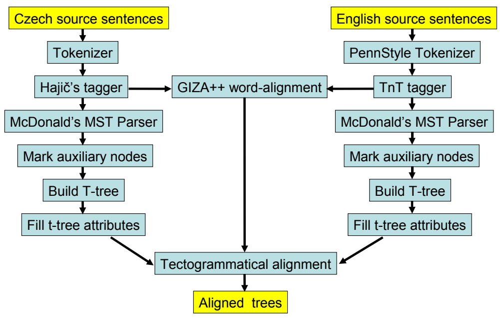
Usage:

- Extracting probabilistic translation dictionary from tectogramatically aligned parallel corpora.

Tecto-alignment x word-alignment



Schema



T-Aligner

- Greedy algorithm based on features
- A score is assigned to each possible connection (pair of Czech and English node)

$$score(en, cs) = \sum w_i \cdot f_i(en, cs)$$

- The weights ***w*** of the features ***f*** were obtained by perceptron learning
- Examples of features:
  – translation probability between tectogrammatical lemmas
  – similar position of nodes in the tree
  – similarities in other attributes
  – child/parent nodes similarities
- In each step, the algorithm finds the pair with the highest score.
- If both the nodes are free and the score is higher than a threshold, we connect them. (only on-to-one connections are allowed)

Alignment evaluation

- 2500 parallel sentences (E-Books, newspaper articles, EU-laws) were manually aligned on the word level, each by two annotators.
- The acquired word-alignment was then transferred to the tectogrammatical layer through the **lex.rf** references
  - **lex.rf** – attribute of a tectogrammatical node, refers to the analytical node from which it acquired its lexical meaning.

Aligner	F-measure
Our T-aligner	88.5 %
GIZA++ word-alignment transferred to t-trees	85.7 %
Our T-aligner using also GIZA++ word-alignment	91.0 %
(Inter-annotator agreement)	*94.8 %*

Tectogrammatical alignment results

References

David Mareček, Zdeněk Žabokrtský, Václav Novák: *Automatic Alignment of Czech and English Deep Syntactic Dependency Trees.* In Proceedings of EAMT08, Hamburg, Germany, 2008

Franz Josef Och, Hermann Ney: *A Systematic Comparison of Various Statistical Alignment Models*. Computational Linguistics 29(1), p.19-51, 2003

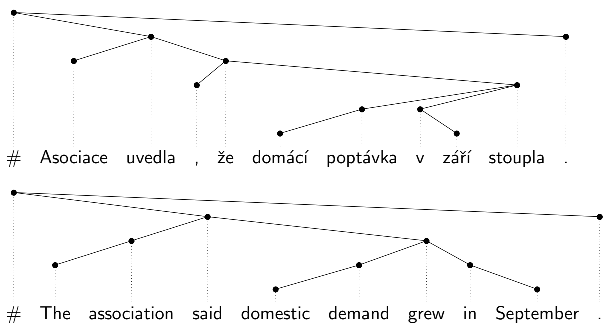
Bad News, NLP Hacking and Feature Fishing

Ondřej Bojar
bojar@ufal.mff.cuni.cz
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University, Prague

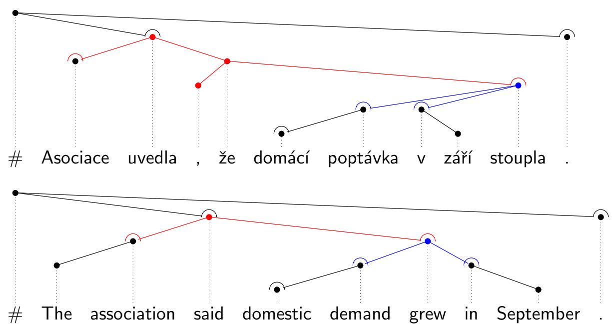
Outline

- Bad news: Syntax-based transfer is hard.
- NLP hacking:
  - Hinglish.
  - Source valency information.
- Proper feature fishing (near future experiments):
  - Phrase table marking, not filtering.
  - Source context features.

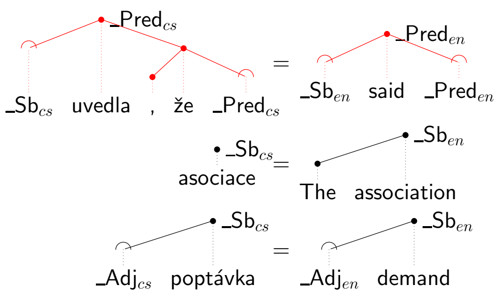
Idea: 1: Observe a Pair of Trees. . .



2: . . . Decompose into Treelets. . .



3: . . . Collect Dictionary of Treelets



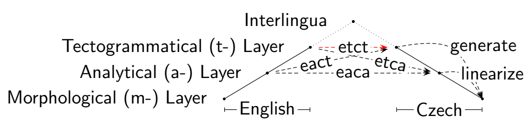
Synchronous Tree Substitution Grammar, e.g. Čmejrek (2006).
More details in Bojar and Čmejrek (2007).

Moses-like Decoding STSG

Given an input dependency tree:

- decompose it into known treelets,
- replace treelets by their treelet translations,
- join output treelets and produce output final tree; linearize or generate plaintext.

Applicable at or across layers:

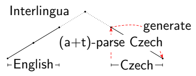


In Reality, t-nodes are not Atomic!

t-nodes have ~25 attributes: t-lemma, functor, gender, person, tense, iterativeness, dispositional modality, . . .

Upper Bound on MT Quality via t-layer:

- Analyse Czech sentences to t-layer.
- Optionally ignore some node attributes.
- Generate Czech surface.
- Evaluate BLEU against input Czech sentences.



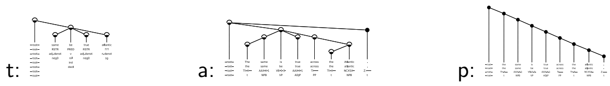
	BLEU
Full automatic t-layer, no attributes ignored	36.6±1.2
Ignore sentence mood (assume indicative)	36.6±1.2
Ignore verbal fine-grained info (resultativeness, . . .)	36.6±1.2
Ignore verbal tense, aspect, . . .	24.9±1.1
Ignore all grammatemes	5.3±0.5

⇒ Node attributes obviously very important.

BLEU Scores for STSG Transfer

- Identical decoder, only the structure + node labels differ.

Layers \ Language Models	no LM	with LM
epcp, atomic nodes	8.65±0.55	10.90±0.63
eaca, atomic nodes	6.59±0.52	8.75±0.61
etct, generated attrs, fixed structure	5.31±0.53	5.61±0.50
etct, atomic nodes, all attributes	1.61±0.33	2.56±0.35
etct, atomic nodes, just t-lemmas	0.67±0.19	-

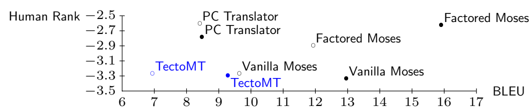


Why Is the t-layer So Poor?

- **Cumulation of Errors**:
  - e.g. 93% tagging * 85% parsing * 93% tagging * 92% parsing = 67%
  - We were using ancient tools: (Ratnaparkhi, 1996), (Collins, 1996), . . .
- **Data Loss** due to incompatible structures:
  - Any error in either of the parses and/or the word-alignment prevents treelet pair extraction.
- **Data Sparseness** when attributes or treelet structure atomic:
  - E.g. different case requires a new treelet pair.
  - There is no adjunction in STSG, new modifier needs a new treelet pair.
- **Combinatorial Explosion** when generating attributes dynamically:
  - Target treelets are first fully built, before combination is attempted.
  - Abundance of t-node attribute combinations
    - ⇒ e.g. lexically different translation options pushed off the stack
    - ⇒ n-bestlist varies in unimportant attributes.

Don't Dump Deep Syntax Yet

WMT08 Results	In-domain •		Out-of-domain ○	
	BLEU	Rank	BLEU	Rank
Factored Moses	15.91	-2.62	11.93	-2.89
PC Translator	8.48	-2.78	8.41	-2.60
TectoMT	9.28	-3.29	6.94	-3.26
Vanilla Moses	12.96	-3.33	9.64	-3.26
etct	4.98	-	3.36	-



- TectoMT ranked comparably to vanilla Moses (BLEU is wrong anyway).
- TectoMT great for preparing rich data.

NLP Hacking vs. Feature Fishing

NLP Hacking:

= Hardcoded behaviour based on some (rich/deep) feature.
- Well motivated but not well built into general search.
- Usually equivalent to deterministic modification of the source language.

Feature Fishing:

= Search properly considers additional features.
- Each feature softly steers the search.
- Data (training/optimization) decide which feature is important.
- The research goal is to have a few most informative features.

Feature Fishing ∼ Discriminative Training; also tomorrow.

NLP Hacking: Hinglish

Bojar et al. (2008) use TectoMT for rule-based reordering:

1. Parse English using MST parser (McDonald et al., 2005),
2. Move finite verbs to the end of the clause,
3. Transform prepositions to postpositions.

Hinglish→Hindi translation using Moses:

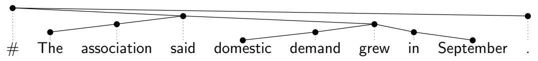
- Baselines: Distance-based or lexicalized reordering,
- Improved: (Rule-base Reord. and) Suffix LM with + Optional

	EILMT	TIDES
Baseline Moses, Distance Reordering	18.88±2.05	10.06±0.76
Baseline Moses, Reordering Using en+hi Forms	19.77±2.03	**10.95±0.75**
Suffix LM+Reord	20.09±2.18	10.18±0.74
Rule-based Reordering + Suffix LM+Reord	**21.01±2.18**	10.29±0.69

Join TectoMT tutorial lab session for SVO→SOV in 12 lines of Perl.

NLP Hacking: Valency Information

Bring non-local information closer based on dependency edges:



To produce "verbose tokens":

the|said assoc.|said said|- domestic|grew demand|grew grew|said in|grew September|in

Remember to back-off with regular tokens:
the assoc. said domestic demand grew in September
Details and further explanation: "Alternative decoding paths" in Friday lecture.

- Should help lexical choice under verbs (verb revealed).
- Should help case choice under prepositions.

en→cs preliminary BLEU scores	80k	2.2M sents.
Baseline	9.77±0.69	**14.57±0.83**
With source valency	**9.98±0.67**	14.52±0.85

Fishing: Phrase Table Marking

- Hard constraints always hurt. Also e.g. Ambati and Lavie (2008).
- Instead of dropping phrase/treelet table entries, *mark* them with an additional score/feature.
- MERT (see Friday class) will decide how much should the marked entries be penalized.

```
in europa ||| in europe ||| 0.829007 0.207955 0.801493 0.492402 2.718 1
europas ||| in europe ||| 0.0251019 0.066211 0.0342506 0.0079563 2.718 1
in europa , ||| in europe ||| 0.011371 0.207955 0.207843 0.492402 2.718 0
```

E.g. mark phrases in phrase table:

- confirmed by a printed/on-line dictionary,
- consistent with surface syntax,
- consistent with deep syntax and t-alignment

Currently me and Václav Novák, happy to join others.

Fishing: Source-Context Features

Some scores phrase translations could be computed on-line:

1. Create translation options for a span as usual.
2. Feed them to an external scorer.
3. Obtain an additional score for each translation option.

Such "dynamic scores" can condition on source sentence context:

- syntactic structure,
- detailed attributes (e.g. case), *without causing data sparseness.*

Consider "John loves Mary":
- Translation options for Mary: $\text{Marie}_{\text{nom}}$ $\text{Marii}_{\text{acc,dat}}$, ...
- Given "Mary" is object, "$\text{Marii}_{\text{acc,dat}}$" should be promoted.
- Better than relying on the presence of 2-word phrase "loves Mary" in the phrase table.

Me and Kamil Kos are looking for collaborators.
The "backdoor" from Moses to arbitrary external scorer implemented, we need to train the scorer.
Inspired by Carpuat and Wu (2007) and Trevor Cohn (pers.comm.).

Summary

- Syntax as a hard constraint is bad.
  - More so, if your tagger+parser+... are not perfect.
- Rich annotation is dangerous when not treated carefully.
  Occam's razor: think twice before adding an attribute.
  - Avoid data sparseness, always provide a back-off.
  - Avoid complex models, they are hard to tune (set parameters).

TectoMT is great for rich annotation and NLP hacking.

Feature fishing for Moses proposed:

- Marking phrases compatible/confirmed by an additional source.
- Dynamic source-context features.

References

Vamshi Ambati and Alon Lavie. 2008. Improving Syntax-Driven Translation Models by Re-structuring Divergent and Nonisomorphic Parse Tree Structures. In *Proc. of AMTA*, pages 235–244.

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

Ondřej Bojar, Pavel Straňák, and Daniel Zeman. 2008. English-hindi translation in 21 days. In *Proceedings of the 6th International Conference On Natural Language Processing (ICON-2008) NLP Tools Contest*, Pune, India. NLP Association of India.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic.

Martin Čmejrek. 2006. *Using Dependency Tree Structure for Czech-English Machine Translation.* Ph.D. thesis, ÚFAL, MFF UK, Prague, Czech Republic.

Michael Collins. 1996. A New Statistical Parser Based on Bigram Lexical Dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191.

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.

Adwait Ratnaparkhi. 1996. A Maximum Entropy Part-Of-Speech Tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*, University of Pennsylvania, May.

TectoMT Tutorial

Jana Kravalová

Welcome at TectoMT Tutorial. This tutorial should take about 3 hours.

What is TectoMT

TectoMT is a highly modular NLP (Natural Language Processing) software system implemented in Perl programming language under Linux. It is primarily aimed at Machine Translation, making use of the ideas and technology created during the Prague Dependency Treebank project. At the same time, it is also hoped to facilitate and significantly accelerate development of software solutions of many other NLP tasks, especially due to re-usability of the numerous integrated processing modules (called blocks), which are equipped with uniform object-oriented interfaces.

Prerequisities

In this tutorial, we assume

- Your system is Linux
- Your shell is bash
- You have basic experience with bash and can read basic Perl

Installation and setup

- Checkout SVN repository. If you are running this installation in computer lab in Prague, you have to checkout the repository into directory `/BIG` (because bigger disk quota applies here):

```
cd ~/BIG
svn --username mtm co https://svn.ms.mff.cuni.cz/svn/tectomt_devel/trunk tectomt
```

- In `tectomt/install/` run `./install.sh`:

```
cd tectomt/install
./install.sh
```

- In your `.bashrc` file, add line (or source the specified file every time before experimenting with TectoMT):

```
source ~/BIG/tectomt/config/init_devel_environ.sh
```

- In your `.bash_profile` file, add line

```
source .bashrc
```

TectoMT Architecture

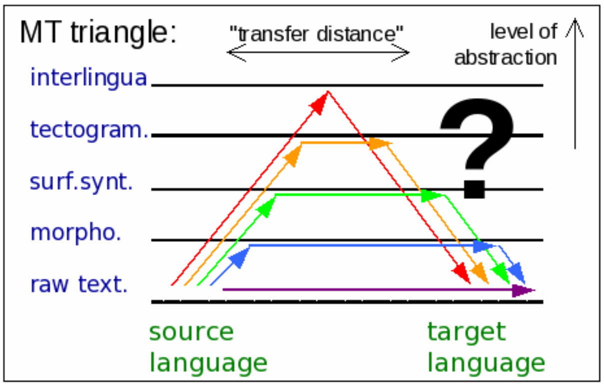
Blocks, scenarios and applications

In TectoMT, there is the following hierarchy of processing units (software components that process data):

- The basic units are blocks. They serve for some very limited, well defined, and often linguistically interpretable tasks (e.g., tokenization, tagging, parsing). Technically, blocks are Perl classes inherited from `TectoMT::Block`, each saved in a separate file. The blocks repository is in `libs/blocks/`.
- To solve a more complex task, selected blocks can be chained into a block sequence, called also a scenario. Technically, scenarios are instances of `TectoMT::Scenario` class, but in some situations (e.g. on the command line) it is sufficient to specify the scenario simply by listing block names separated by spaces.
- The highest unit is called application. Applications correspond to end-to-end tasks, be they real end-user applications (such as machine translation), or 'only' NLP-related experiments. Technically, applications are often implemented as `Makefiles`, which only glue the components existing in TectoMT. Some demo applications can be found in `applications`.

This tutorial itself has its blocks in `libs/blocks/Tutorial` and the application in `applications/tutorial`.

Layers of Linguistic Structures



The notion of 'layer' has a combinatorial nature in TectoMT. It corresponds not only to the layer of language description as used e.g. in the Prague Dependency Treebank, but it is also specific for a given language (e.g., possible values of morphological tags are typically different for different languages) and even for how the data on the given layer were created (whether by analysis from the lower layer or by synthesis/transfer).

Thus, the set of TectoMT layers is a Cartesian product {S,T} x {English,Czech,...} x {W,M,P,A,T}, in which:

- {S,T} distinguishes whether the data was created by analysis or transfer/synthesis (mnemonics: S and T correspond to (S)ource and (T)arget in MT perspective).
- {English,Czech...} represents the language in question
- {W,M,P,A,T...} represents the layer of description in terms of PDT 2.0 (W – word layer, M – morphological layer, A – analytical layer, T – tectogrammatical layer) or extensions (P – phrase-structure layer).

Blocks in block repository `libs/blocks` are located in directories indicating their purpose in machine translation.

Example: A block adding Czech morphological tags (pos, case, gender, etc.) can be found in `libs/blocks/SCzechW_to_SCzechM/Simple_tagger.pm`.

There are also other directories for other purpose blocks, for example blocks which only print out some information go to `libs/Print`. Our tutorial blocks are in `libs/blocks/Tutorial/`.

First application

Once you have TectoMT installed on your machine, you can find this tutorial in `applications/tutorial/`. After you `cd` into this directory, you can see our plain text sample data in `sample.txt`.

Most applications are defined in `Makefiles`, which describe sequence of blocks to be applied on our data. In our particular `Makefile`, four blocks are going to be applied on our sample text: sentence segmentation, tokenization, tagging and lemmatization. Since we have our input text in plain text format, the file is going to be converted into `tmt` format beforehand (the `in` target in the `Makefile`).

We can run the application:

```
make all
```

Our plain text data `sample.txt` have been transformed into `tmt`, an internal TectoMT format, and saved into `sample.tmt`. Then, all four blocks have been loaded and our data has been processed. We can now examine `sample.tmt` with a text editor (vi, emacs, etc).

- One physical `tmt` file corresponds to one document.
- A document consists of a sequence of bundles (`<bundle>`), mirroring a sequence of natural language sentences originating from the text. So, for one sentence we have one `<bundle>`.
- Each bundle contains tree shaped sentence representations on various linguistic layers. In our example `sample.tmt` we have morphological tree (`SEnglishM`) in each bundle. Later on, also an analytical layer (`SEnglishA`) will appear in each bundle as we proceed with our analysis.
- Trees are formed by nodes and edges. Attributes can be attached only to nodes. Edge's attributes must be stored as the lower node's attributes. Tree's attributes must be stored as attributes of the root node.

Changing the scenario

We'll now add a syntax analysis (dependency parsing) to our scenario by adding three more blocks. Instead of

```
analyze:
	brunblocks -S -o \
		SEnglishW_to_SEnglishM::Sentence_segmentation_simple \
		SEnglishW_to_SEnglishM::Penn_style_tokenization \
		SEnglishW_to_SEnglishM::TagMxPost \
		SEnglishW_to_SEnglishM::Lemmatize_mtree \
	-- sample.tmt
```

we'll have:

```
analyze:
	brunblocks -S -o \
		SEnglishW_to_SEnglishM::Sentence_segmentation_simple \
		SEnglishW_to_SEnglishM::Penn_style_tokenization \
		SEnglishW_to_SEnglishM::TagMxPost \
		SEnglishW_to_SEnglishM::Lemmatize_mtree \
		SEnglishM_to_SEnglishA::McD_parser_local \
		SEnglishM_to_SEnglishA::Fix_McD_Tree \
		SEnglishM_to_SEnglishA::Fill_afun_after_McD \
	-- sample.tmt
```

Note: `Makefiles` use tabulators to mark command lines. Make sure your lines start with a tabulator (or two tabulators) and not, for example, with 4 spaces.

After running

```
make all
```

we can examine our `sample.tmt` again. Really, an analytical layer `SEnglishA` describing a dependency tree with analytical functions (`<afun>`) has been added to each bundle.

Blocks can also be parametrized. For syntax parser, we might want to use a smaller but faster model. To achieve this, replace the line

```
SEnglishM_to_SEnglishA::McD_parser_local \
```

with

```
SEnglishM_to_SEnglishA::McD_parser_local TMT_PARAM_MCD_EN_MODEL=conll_mcd_order2_0.1.model \
```

You can view the trees in `sample.tmt` with TrEd by typing

```
tmttred sample.tmt
```

Try to click on some nodes to see their parameters (tag, lemma, form, analytical function etc).

Note: For more information about tree editor TrEd, see TrEd User's Manual.

If you are not familiar with `Makefile` syntax, another way of running a scenario in TectoMT is using `.scen` file (see `applications/tutorial.scen`). This file lists the blocks to be run – one block on a single line.

```
eval \${TMT_ROOT}/tools/format_convertors/plaintext_to_tmt/plaintext_to_tmt.pl English sample.txt
brunblocks -S -o --scen tutorial.scen -- sample.tmt
```

Finally, yet another way is to use a simple `bash` script (see `applications/tutorial/run_all.sh`):

```
./run_all.sh
```

Adding a new block

The linguistic structures in TectoMT are represented using the following object-oriented interface/types:

- document – `TectoMT::Document`
- bundle – `TectoMT::Bundle`
- node – `TectoMT::Node`

You can get TectoMT automatically execute your block code on each document or bundle by defining the main block entry point:

- `sub process_document` – run this procedure on each document
- `sub process_bundle` – run this procedure on each bundle (sentence)

Each block must have exactly one entry point.

We'll now examine an example of a new block in file `libs/blocks/Tutorial/Print_node_info.pm`.

This block illustrates some of the most common methods for accessing objects:

- `my @bundles = $document->get_bundles()` – an array of bundles contained in the document
- `my $root_node = $bundle->get_tree($layer_name)` – the root node of the tree of the given type in the given bundle
- `my @children = $node->get_children()` – array of the node's children
- `my @descendants = $node->get_descendants()` – array of the node's children and their children and children of their children ...
- `my $parent = $node->get_parent()` – parent node of the given node, or undef for root
- `my $root_node = $node->get_root()` – the root node of the tree into which the node belongs

Attributes of documents, bundles or nodes can be accessed by attribute getters and setters, for example:

- `$node->get_attr($attr_name)`
- `$node->set_attr($attr_name, $attr_value)`

Some interesting attributes on morphologic layer are `form`, `lemma` and `tag`. Some interesting attributes on analytical layer are `afun` (analytical function) and `ord` (surface word order). To reach `form`, `lemma` or `tag` from analytical layer, that is, when calling this attribute on an `a-node`, you use `$a_node->get_attr('m/form')` and the same way for `lemma` and `tag`. The easiest way to see the node attributes is to click on the node in TrEd:

```
tmttred sample.tmt
```

Our tutorial block `Print_node_info.pm` is ready to use. You only need to add this block to our scenario, e.g. as a new `Makefile` target:

```
print_info:
	brunblocks -S -o Tutorial::Print_node_info -- sample.tmt
```

We can observe our new block behaviour:

```
make print_info
```

Try to change the block so that it prints out the information only for verbs. (You need to look at an attribute `tag` at the `m` level). The tagset used is Penn Treebank Tagset.

Advanced block: finite clauses

Motivation

It is assumed that finite clauses can be translated independently, which would reduce combinatorial complexity or make parallel translation possible. We could even use hybrid translation – each finite clause could be translated by the most self-confident translation system. In this task, we are going to split the sentence into finite clauses.

Task

A block which, given an analytical tree (`SEnglishA`), fills each `a-node` with boolean attribute `is_clause_head` which is set to `1` if the `a-node` corresponds to a finite verb, and to `0` otherwise.

Instructions

There is a block template with hints in `libs/blocks/Tutorial/Mark_heads.pm`. You should edit the block so that the output of this block is the same a-tree, in addition with attribute `is_clause_head` attached to each `a-node`. There is also a printing block `libs/blocks/Print_finite_clauses.pm` which will print out the `a-nodes` grouped by clauses:

```
finite_clauses:
	brunblocks -S -o \
		Tutorial::Mark_heads \
		Tutorial::Print_finite_clauses \
	-- sample.tmt
```

You are going to need these methods:

- `my $root = $bundle->get_tree('tree_name')`

- `my $attr = $node->get_attr('attr_name')`
- `$node->set_attr('attr_name',$attr_value)`
- `my @eff_children = $node->get_eff_children()`

Note: `get_children()` returns topological node children in a tree, while `get_eff_children()` returns node children in a linguistic sense. Mostly, these do not differ. If interested, see Figure 1 in btred tutorial.

Hint: Finite clauses in English usually require grammatical subject to be present.

Advanced version

The output of our block might still be incorrect in special cases – we don't solve coordination (see the second sentence in sample.txt) and subordinate conjunctions.

Your turn: more tasks

SVO to SOV

Motivation: During translation from an SVO based language (e.g. English) to an SOV based language (e.g. Korean), we might need to change the word order from SVO to SOV.

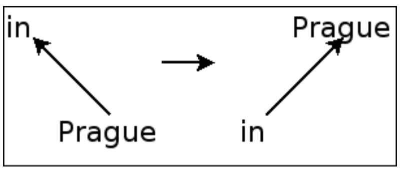
Task: Change the word order from SVO to SOV.

Instructions:

- You can use block template in `libs/blocks/BlockTemplate.pm`.
- To find an object of a verb, look for objects among effective children of a verb (`$child->get_attr('afun') eq 'Obj'`). That implies working on analytical layer.
- For debugging, a method returning surface word order of a node is useful: `$node->get_attr('ord')`. It can be used to print out nodes sorted by attribute `ord`.
- Once you have the node `$object` and the node `$verb`, use the method `$object->shift_before_node($verb)`. This method takes the whole subtree under the node `$object` and recalculates the attributes `ord` (surface word order) so that all the nodes in the subtree under `$object` have a smaller `ord` than `$verb`. That is, the method rearranges the surface word order from VO to OV.

Advanced version: This solution shifts object (or more objects) of a verb just in front of that verb node. So f.e.: *Mr. Brown has urged MPs.* changes to: *Mr. Brown has MPs urged.* You can try to change this solution, so the final sentence would be: *Mr. Brown MPs has urged.* You may need a method `$node->shift_after_subtree($root_of_that_subtree)`. Subjects should have attribute `'afun' eq 'Sb'`.

Prepositions



Motivation: In dependency approach the question "where to hang prepositions" arises. In the praguian style (PDT), prepositions are heads of the subtree and the noun/pronoun is dependent on the preposition. However, another ordering might be preferable: The noun/pronoun might be the head of subtree, while the preposition would take the role of a modifier.

Task: The task is to rehang all prepositions as indicated at the picture. You may assume that prepositions have at most 1 child.

Instructions:
You are going to need these new methods:

- `my @children = $node->get_children()`
- `my $parent = $node->get_parent()`
- `$node->set_parent($parent)`

Hint:

- On analytical layer, you can use this test to recognize prepositions: `$node->get_attr('afun') eq 'AuxP'`
- To see the results, you can again use TrEd (`tmttred sample.tmt`)

Advanced version: What happens in case of multiword prepositions? For example, `because of`, `instead of`. Can you handle it?

Winter School

Day 5: Discriminative Training and Factored Translation Models

MT Marathon

30 January 2009

1

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)

2

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

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

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

3

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

4

Set feature weights

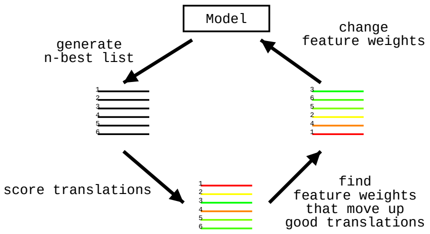
- Contribution of components p_i determined by weight λ_i
- Methods
  - *manual setting* of weights: try a few, take best
  - *automate* this process
- Learn weights
  - set aside a **development corpus**
  - set the weights, so that **optimal translation performance** on this development corpus is achieved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

5

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

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

Learning task

- Task: *find weights*, so that feature vector of best translations *ranked first*
- Input: *Er geht ja nicht nach Hause*, Ref: *He does not go home*

Translation	Feature values						Error
it is not under house	-32.22	-9.93	-19.00	-5.08	-8.22	-5	0.8
he is not under house	-34.50	-7.40	-16.33	-5.01	-8.15	-5	0.6
it is not a home	-28.49	-12.74	-19.29	-3.74	-8.42	-5	0.6
it is not to go home	-32.53	-10.34	-20.87	-4.38	-13.11	-6	0.8
it is not for house	-31.75	-17.25	-20.43	-4.90	-6.90	-5	0.8
he is not to go home	-35.79	-10.95	-18.20	-4.85	-13.04	-6	0.6
he does not home	-32.64	-11.84	-16.98	-3.67	-8.76	-4	**0.2**
it is not packing	-32.26	-10.63	-17.65	-5.08	-9.89	-4	0.8
he is not packing	-34.55	-8.10	-14.98	-5.01	-9.82	-4	0.6
he is not for home	-36.70	-13.52	-17.09	-6.22	-7.82	-5	0.4

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
```

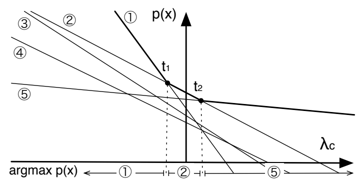

Find Best Feature Weight

- Core task:
  - find optimal value for one parameter weight λ
  - ... while leaving all other weights constant
- Score of translation i for a sentence $\mathbf{f}$:

$$p(\mathbf{e}_i|\mathbf{f}) = \lambda a_i + b_i$$

- Recall that:
  - we deal with 100s of translations $\mathbf{e}_i$ per sentence $\mathbf{f}$
  - we deal with 100s or 1000s of sentences $\mathbf{f}$
  - we are trying to find the value λ so that over all sentences, the error score is optimized

Translations for one Sentence



- each translation is a line $p(\mathbf{e}_i|\mathbf{f}) = \lambda a_i + b_i$
- the model-best translation for a given λ (x-axis), is highest line at that point
- there are one a few *threshold points* t_j where the model-best line changes

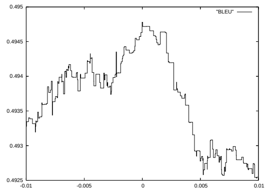
Finding the Optimal Value for λ

- Real-valued λ can have infinite number of values
- But only on threshold points, one of the model-best translation changes

⇒ Algorithm:
- find the threshold points
- for each interval between threshold points
  - find best translations
  - compute error-score
- pick interval with best error-score

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)

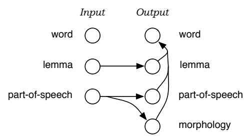
run	iterations	dev score	test score
1	8	50.16	51.99
2	9	50.26	51.78
3	8	50.13	51.59
4	12	50.10	51.20
5	10	50.16	51.43
6	11	50.02	51.66
7	10	50.25	51.10
8	11	50.21	51.32
9	10	50.42	51.79

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

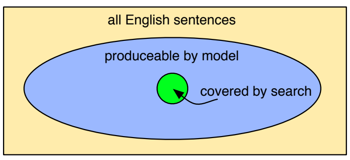
- Fundamental problem in machine learning
  - what works best for training data, may not work well in general
  - **rare, unrepresentative features** may get too much weight
- **Especially severe problem** in phrase-based models
  - **long phrase pairs** explain well *individual sentences*
  - ... but 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

Factored Translation Models

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

Statistical machine translation today

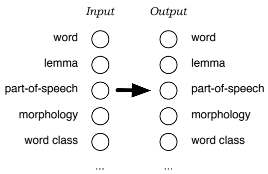
- Best performing methods based on **phrases**
  - short sequences of words
  - no use of explicit syntactic information
  - no use of morphological information
  - currently best performing method
- Progress in **syntax-based** translation
  - tree transfer models using syntactic annotation
  - still 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

Decomposing translation: example

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

$$\left[\text{lemma}\right] \Rightarrow \left[\text{lemma}\right]$$

$$\left[\begin{array}{c}\text{part-of-speech}\\ \text{morphology}\end{array}\right] \Rightarrow \left[\begin{array}{c}\text{part-of-speech}\\ \text{morphology}\end{array}\right]$$

Decomposing translation: example

- **Generate surface** form on target side

$$\left[\text{surface}\right]$$
$$\Uparrow$$
$$\left[\begin{array}{c}\text{lemma}\\ \text{part-of-speech}\\ \text{morphology}\end{array}\right]$$

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

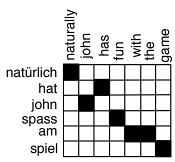
- Motivation
- Example
- **Model and Training**
- Decoding
- Experiments

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

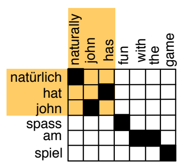
Phrase-based training

- Establish word alignment (GIZA++ and symmetrization)



Phrase-based training

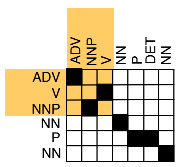
- Extract phrase



⇒ *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)

42

Factored Translation Models

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

43

Phrase-based translation

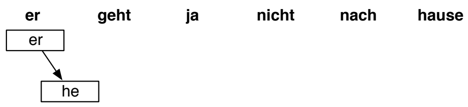
- Task: *translate this sentence* from German into English

er geht ja nicht nach hause

44

Translation step 1

- Task: translate this sentence from German into English

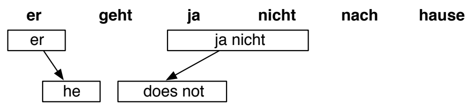


- *Pick* phrase in input, *translate*

45

Translation step 2

- Task: translate this sentence from German into English

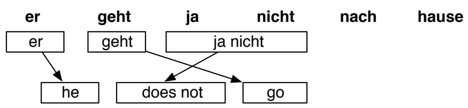


- Pick phrase in input, translate
  - it is allowed to pick words *out of sequence* (**reordering**)
  - phrases may have multiple words: *many-to-many* translation

46

Translation step 3

- Task: translate this sentence from German into English

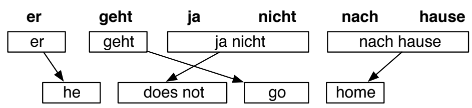


- Pick phrase in input, translate

47

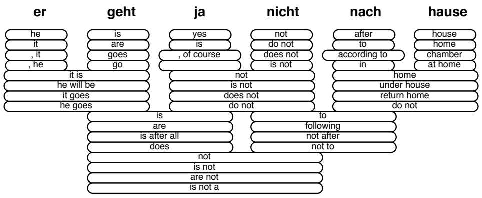
Translation step 4

- Task: translate this sentence from German into English



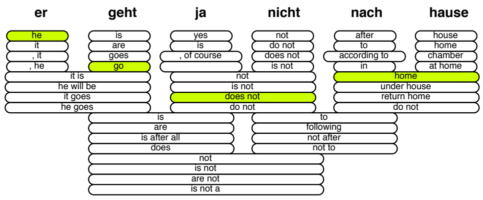
- Pick phrase in input, translate

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

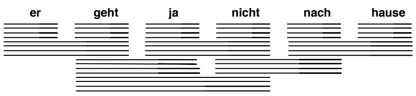
Translation options



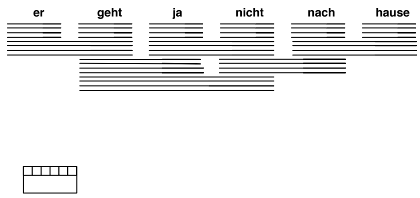
- The machine translation decoder does not know the right answer

→ *Search problem* solved by heuristic beam search

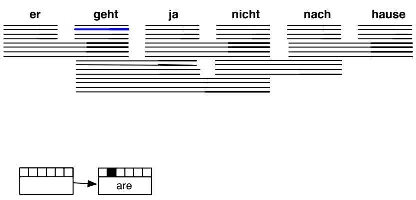
Decoding process: precompute translation options



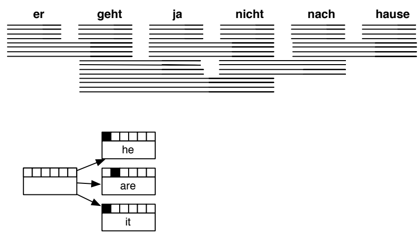
Decoding process: start with initial hypothesis



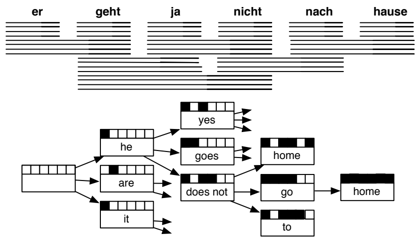
Decoding process: hypothesis expansion



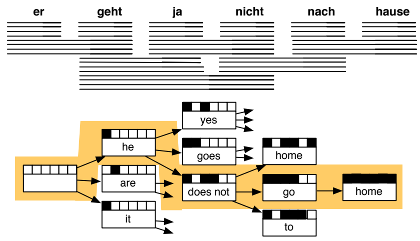
Decoding process: hypothesis expansion



Decoding process: hypothesis expansion



Decoding process: find best path



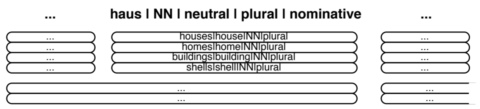
Factored model decoding

- 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*



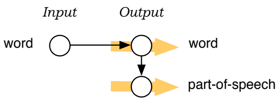
Efficient factored model decoding

- 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**

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

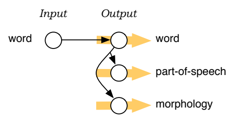
- Improvements under sparse data conditions
- Similar results with CCG supertags [Birch et al., 2007]

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

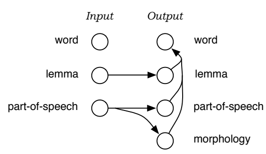
Agreement within noun phrases

- 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

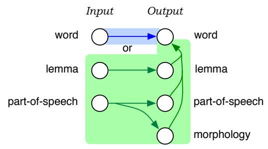
Initial results

- 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

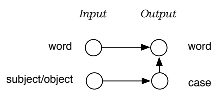
- Model now beats the baseline:

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)

72

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

73

Factored Template Models

- **Long range** reordering
  - movement often not limited to local changes
  - German-English: *SBJ AUX OBJ V* → *SBJ AUX V OBJ*
- Template 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
- published in [Hoang and Koehn, EACL 2009]

74

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
- Results presented in [Cettolo et al., AMTA 2008]