MT Marathon

26th – 30th January 2009

Prague, Czech Republic
Lectures, Talks, Labs

<table>
<thead>
<tr>
<th></th>
<th>Morning Lecture</th>
<th>Research Talks</th>
<th>Afternoon Lab Session</th>
<th>Evening</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday</td>
<td>Word Alignment (Barry Haddow)</td>
<td>PostCAT, apertium-cy, MBMT</td>
<td>Implementing IBM model 1</td>
<td>Projects: Update</td>
</tr>
<tr>
<td>Wednesday</td>
<td>Phrase-Based Models and Decoding (Chris Dyer)</td>
<td>Joshua, SAMT, MERT+</td>
<td>Installing and running Moses (Hieu Hoang and Josh Schroeder)</td>
<td></td>
</tr>
<tr>
<td>Thursday</td>
<td>TectoMT: Processing Trees (Zdeněk Žabokrtský and others)</td>
<td>RIA, Sub-Tree Aligner</td>
<td>TectoMT hands-on experience: Installation and tutorial (Jana Kravalová)</td>
<td>Projects: Update</td>
</tr>
<tr>
<td>Friday</td>
<td>Richer Models and Optimization (Philipp Koehn)</td>
<td>Projects: Final Short Presentations</td>
<td>Using factored models and MERT in Moses (Hieu Hoang, Barry Haddow, Abhishek Arun)</td>
<td></td>
</tr>
</tbody>
</table>

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
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
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

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

**Monday**
- Introduction to MT ......................................................... 4
- MT Evaluation ................................................................. 13
- Intro to Stat-XFER ............................................................ 19
- Stat-XFER: Czech to English Translation Project ...................... 29
- TectoMT for Plaintext Freaks .............................................. 33

**Tuesday**
- Word Alignment ................................................................. 35

**Wednesday**
- Phrase-Based Models and Decoding ...................................... 42

**Thursday**
- Introduction to the Software Framework ................................. 51
- TectoMT: Alignment ............................................................ 57
- Bad news, NLP Hacking and Feature Fishing ............................. 59
- TectoMT: Tutorial ............................................................... 62

**Friday**
- Discriminative Training and Factored Translation Models .......... 69
Various approaches

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

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

- Find most probable English sentence given a foreign language sentence
  \[ p(e|f) \]

Probabilities
Probabilities

- Find most probable English sentence given a foreign language sentence

\[ p(e|f) \]
\[ \hat{e} = \arg \max_e p(e|f) \]

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

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

\[ \text{p}(I \text{ like bungee jumping off high bridges}) = \]

Trigram language model

\[ \text{p}(I \text{ like bungee jumping off high bridges}) = \]
\[ \text{p}(I | <s> <s>) \times \]
\[ \text{p}(\text{like} | <s> I) \times \]
\[ \text{p}(\text{bungee} | I \text{ like}) \times \]
\[ \text{p}(\text{jumping} | \text{like bungee}) \times \]
\[ \text{p}(\text{off} | \text{bungee jumping}) \times \]

Trigram language model

\[ \text{p}(I \text{ like bungee jumping off high bridges}) = \]
\[ \text{p}(I | <s> <s>) \times \]
\[ \text{p}(\text{like} | <s> I) \times \]
\[ \text{p}(\text{bungee} | I \text{ like}) \times \]
\[ \text{p}(\text{jumping} | \text{like bungee}) \times \]
\[ \text{p}(\text{off} | \text{bungee jumping}) \times \]

Trigram language model

\[ \text{p}(I \text{ like bungee jumping off high bridges}) = \]
\[ \text{p}(I | <s> <s>) \times \]
\[ \text{p}(\text{like} | <s> I) \times \]
\[ \text{p}(\text{bungee} | I \text{ like}) \times \]
\[ \text{p}(\text{jumping} | \text{like bungee}) \times \]
\[ \text{p}(\text{off} | \text{bungee jumping}) \times \]
Trigram language model

\[ p(I \text{ like bungee jumping off high bridges}) = \]
\[ p(I | <s> <s>) \times \]
\[ p(like | <s> I) \times \]
\[ p(bungee | I \text{ like}) \times \]
\[ p(jumping | like \text{ bungee}) \times \]
\[ p(off | \text{ bungee jumping}) \times \]
\[ p(high | jumping off) \times \]
\[ p(bridges | off high) \times \]
\[ p(</s> | high bridges) \times \]

Calculating Language Model Probabilities

- **Unigram probabilities**
  \[ p(w_1) = \frac{\text{count}(w_1)}{\text{total words observed}} \]

- **Bigram probabilities**
  \[ p(w_2|w_1) = \frac{\text{count}(w_1w_2)}{\text{count}(w_1)} \]
Calculating Language Model Probabilities

- Trigram probabilities
  \[ p(w_3|w_1w_2) = \frac{\text{count}(w_1w_2w_3)}{\text{count}(w_1w_2)} \]

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

- \[.8 \times p(w_3|w_1w_2) + .15 \times p(w_3|w_2) + .049 \times p(w_3) + .001\]
- Avoids zero probs

Translation model

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

Translation model

- How do we assign values to \( p(f|e) \)?
  \[ p(f|e) = \frac{\text{count}(f,e)}{\text{count}(e)} \]
- 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 \( t(f_j|e_i) \)

- 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

- Counting! I told you probabilities were easy!
  \[ = \frac{\text{count}(f_j, e_i)}{\text{count}(e_i)} \]
- 100 times total 13 with this \( f \) 13%
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

ergeht janicht nach hause

are it he goes does not go to home

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

Tree-based models

S → NP(1) PP(2) VP(3)
NP → Baowei, Powell
PP → yu Shalong, with Sharon
VP → juxing le huitan, held a meeting
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](http://www.statmt.org) - papers, tutorials, etc.
  At [http://homepages.inf.ed.ac.uk/alopez](http://homepages.inf.ed.ac.uk/alopez)
  BibTeX at [http://github.com/alopez/smtbib](http://github.com/alopez/smtbib)
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

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.

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!

Alon Lavie
Language Technologies Institute
Carnegie Mellon University

Joint work with:
Greg Hanneman, Vamsi Ambati, Rob Perlikar, Edmund Huber, Jonathan Clark, Erik Peterson, Christian Manson, Abhaya Agarwal, Kathrin Probst, Ari Font Llitjos, Jaime Carbonell, Rob 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

1/21/2009
Alon Lavie: Stat-XFER

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
  – Automatic acquisition of 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

**Interlingua**

- Syntactic Parsing
- Semantic Analysis
- Sentence Planning
- Transfer Rules
- Statistical-XFER
- Direct: SMT, EBMT

Source (e.g. Arabic)  Target (e.g. English)

**Stat-XFER Framework**

- Source Input
- Transfer Engine
- Translation Lattice
- Second-Stage Decoder
- Weighted Features
- Language Model
- Bilingual Lexicon

**Transfer Rule Formalism**

- Type information
- Part-of-speech/constituent information
- Alignments
- x-side constraints
- y-side constraints
- xy-constraints, e.g. ((Y1 AGR) = (X1 AGR))

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

PRO::PRO |: ["AMI"] -> ["you"]

- (X1 \( \rightarrow \) Y1)
- (X1 \( \rightarrow \) Y2)
- (X1 \( \rightarrow \) Y3)
- (X1 \( \rightarrow \) Y4)

N::N |: ["HOUR"] -> ["hours"]

- (X1 \( \rightarrow \) Y1)
- (X1 \( \rightarrow \) Y2)
- (X1 \( \rightarrow \) Y3)
- (X1 \( \rightarrow \) Y4)

PRO::PRO |: ["AMI"] -> ["you"]

- (X1 \( \rightarrow \) Y1)
- (X1 \( \rightarrow \) Y2)
- (X1 \( \rightarrow \) Y3)
- (X1 \( \rightarrow \) Y4)

N::N |: ["HOUR"] -> ["hours"]

- (X1 \( \rightarrow \) Y1)
- (X1 \( \rightarrow \) Y2)
- (X1 \( \rightarrow \) Y3)
- (X1 \( \rightarrow \) Y4)
Translation Lexicon: French-to-English Examples (Automatically-acquired)

<table>
<thead>
<tr>
<th>Rule</th>
<th>SL</th>
<th>TL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DET::DET</td>
<td>&quot;le&quot;</td>
<td>&quot;the&quot;</td>
</tr>
<tr>
<td>Prep::Prep</td>
<td>&quot;dans&quot;</td>
<td>&quot;in&quot;</td>
</tr>
<tr>
<td>N::N</td>
<td>&quot;principes&quot;</td>
<td>&quot;principles&quot;</td>
</tr>
<tr>
<td>N::N</td>
<td>&quot;respect&quot;</td>
<td>&quot;accordance&quot;</td>
</tr>
<tr>
<td>NP::NP</td>
<td>&quot;le respect&quot;</td>
<td>&quot;accordance&quot;</td>
</tr>
<tr>
<td>PP::PP</td>
<td>&quot;dans le respect&quot;</td>
<td>&quot;in accordance&quot;</td>
</tr>
<tr>
<td>PP::PP</td>
<td>&quot;des principes&quot;</td>
<td>&quot;with the principles&quot;</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Rule</th>
<th>SL</th>
<th>TL</th>
</tr>
</thead>
<tbody>
<tr>
<td>NPI</td>
<td>&quot;MLH ADWMH&quot;</td>
<td>&quot;A RED DRESS&quot;</td>
</tr>
<tr>
<td>NP1::NP1</td>
<td>[NP1 ADJ]</td>
<td>[ADJ NP1]</td>
</tr>
<tr>
<td>NP1::NP1</td>
<td>[NP1 &quot;H&quot; ADJ]</td>
<td>[ADJ NP1]</td>
</tr>
</tbody>
</table>

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

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---|
```

The Transfer Engine

- Input: B$WRH

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

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
  - German-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

Syntax-based MT Resource Acquisition in Resource-rich Scenarios

- Scenario: Significant amounts of parallel-text at sentence-level are available
- Goal: Acquire both broad-coverage translation lexicons and transfer rule grammars automatically from the data
  - 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
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 \(<S>\) 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

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:
  \(NP \# NP :: \)
  \# Australia

All Phrases from this tree pair:

1. IP # S :: /g2 /g3 /g4 /g5 /g6 /g7/g8 /g9 /g10/g11 /g12/g13 /g14/g15 /g16 # Australia is one of the few countries that have diplomatic relations with North Korea
2. VP # VP :: /g3 /g4 /g5 /g6 /g7/g8 /g9 /g10/g11 /g12/g13 /g14/g15 # is one of the few countries that have diplomatic relations with North Korea
3. NP # NP :: /g4 /g5 /g6 /g7/g8 /g9 /g10/g11 /g12/g13 /g14/g15 # one of the few countries that have diplomatic relations with North Korea
4. VP # VP :: /g4 /g5 /g6 /g7 # have diplomatic relations with North Korea
5. NP # NP :: /g7 # diplomatic relations
6. NP # NP :: /g5 # North Korea
7. NP # NP :: /g2 # 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**

<table>
<thead>
<tr>
<th>TYPE</th>
<th>Total</th>
<th>TnS</th>
<th>% TnS</th>
<th>TnT</th>
<th>% TnT</th>
<th>O%</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADVP</td>
<td>600104</td>
<td>412250</td>
<td>68.6</td>
<td>176677</td>
<td>29.4</td>
<td>90.7</td>
</tr>
<tr>
<td>ADVP</td>
<td>1010107</td>
<td>696106</td>
<td>68.9</td>
<td>106512</td>
<td>10.5</td>
<td>82.1</td>
</tr>
<tr>
<td>NP</td>
<td>11204763</td>
<td>8377759</td>
<td>74.7</td>
<td>4152363</td>
<td>37.1</td>
<td>93.8</td>
</tr>
<tr>
<td>VP</td>
<td>4650993</td>
<td>2916282</td>
<td>62.7</td>
<td>2386597</td>
<td>5.1</td>
<td>67.9</td>
</tr>
<tr>
<td>PP</td>
<td>7772964</td>
<td>2766654</td>
<td>75.9</td>
<td>842508</td>
<td>22.3</td>
<td>89.4</td>
</tr>
<tr>
<td>S</td>
<td>2233075</td>
<td>150832</td>
<td>67.4</td>
<td>2482291</td>
<td>11.1</td>
<td>94.5</td>
</tr>
<tr>
<td>SBAR</td>
<td>912240</td>
<td>591755</td>
<td>64.8</td>
<td>42407</td>
<td>4.6</td>
<td>71.9</td>
</tr>
<tr>
<td>SINTQ</td>
<td>19935</td>
<td>9084</td>
<td>45.5</td>
<td>7576</td>
<td>38</td>
<td>99.6</td>
</tr>
</tbody>
</table>

- **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.

**Extracted Syntactic Phrases**

- **TnS**
  - The principles
  - Principles
  - Accordance with the principles
  - Required

- **TnT**
  - The principles
  - Principles

- **TnT**

Comparative Results
French-to-English

<table>
<thead>
<tr>
<th>System</th>
<th>Dev-Set BLEU</th>
<th>Test-Set BLEU</th>
<th>Test-Set METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlonLavie: Stat-XFER37</td>
<td>26.57</td>
<td>27.02</td>
<td>37.08</td>
</tr>
<tr>
<td>AlonLavie: Stat-XFER38</td>
<td>24.44</td>
<td>27.40</td>
<td>37.28</td>
</tr>
<tr>
<td>AlonLavie: Stat-XFER39</td>
<td>20.54</td>
<td>30.18</td>
<td>38.13</td>
</tr>
</tbody>
</table>

- 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)

1/21/2009
Alon Lavie: Stat-XFER

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
- Direct Combination appears to be slightly better so far
- Grammar builds upon syntactic phrases, decoder uses both

1/21/2009
Alon Lavie: Stat-XFER

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
- Algorithm:
  - Find all possible treelet decompositions from the node aligned trees
  - "Flatten" the treelets into synchronous CFG rules

1/21/2009
Alon Lavie: Stat-XFER

Rule Extraction Algorithm

1/21/2009
Alon Lavie: Stat-XFER

Rule Extraction Algorithm

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

### 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

### 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
**French-English System Example Grammar Rules**

NP::NP [N "de" N] -> [N N]

(*sgtrule* 0.736382560)
(*tgsrule* 0.292253105); (*freq* 232772)

X3::Y1
X1::Y2

**English-French System Translation Example**

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

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

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

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 publicly 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

Stat-XFER Data Processing

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

Stat-XFER Data Processing

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

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
Stat-XFER Data Processing

- Phrase extraction:
  - Syntactic extraction by PFA node alignment algorithm, 12ts mode
  - Non-syntactic extraction with Moses package

Phrase Table Combination

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

Final Result

- Two phrase tables, with counts:

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PHR</td>
<td>rozumem</td>
<td>brains</td>
</tr>
<tr>
<td>2</td>
<td>PHR</td>
<td>rozumem</td>
<td>reason</td>
</tr>
<tr>
<td>3</td>
<td>PHR</td>
<td>rozumem</td>
<td>sense</td>
</tr>
<tr>
<td>4</td>
<td>PHR</td>
<td>rozumem</td>
<td>sensible policy</td>
</tr>
<tr>
<td>5</td>
<td>ADJP</td>
<td>rozumou</td>
<td>x &amp; a</td>
</tr>
<tr>
<td>6</td>
<td>ADJP</td>
<td>rozumou</td>
<td>x &amp; a</td>
</tr>
<tr>
<td>7</td>
<td>ADJP</td>
<td>rozumou</td>
<td>x &amp; a</td>
</tr>
<tr>
<td>8</td>
<td>ADJP</td>
<td>rozumou</td>
<td>x &amp; a</td>
</tr>
</tbody>
</table>

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

Stat-XFER Data Processing

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

WMT tuning, development, and test sets

= Baseline Stat-XFER system ready to analyse and expand
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?

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?

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 proti zítře
of work decided to protest

“Peter decided to protest against the dismissal of work tomorrow.”

Synchronous Grammars: Manual Grammar Writing

VP → [ADV VP] → [VP ADV]

\((X1::Y2)\)\n\((X2::Y1)\)
\(*tgsrule* 0.2\)
\(*sgtrule* 0.6\)
\(((X0 tense) = (X1 tense))\)
\(((X0 tense) = (X2 tense))\)

\(VP\)\n\(\ldots\)decided\ldots\)tomorrow
\(ADV\)\n\(\ldots\)tomorrow
\(VP\)\n\(\ldots\)decided\ldots\)

Czech Morphology: Example

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

nero+zumí
ne+rozum=ím
NEG+understand=1SG

“I do not understand”

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
(Your Idea Here)

- Any ideas about applying the statistical transfer framework to Czech-English translation are welcome!
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” ⇒ paintoparallelize.
- 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 mld. |
| 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.

E.g. tools/format_convertors/czeng07/czeng07.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 "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 stderr.
2. Submit parallel printing, e.g.:

   ```bash
   tools/cluster-utils/qrunblocks \
   filelist \
   --jobs 20 --no-save \
   --join \
   > joined_output
   ```

   • `--no-save` avoids saving TMT files,
   • `--join` waits for all the jobs to succeed and joins their stderrs 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.:

   ```bash
   /net/cluster/COMPUTER/tmp/
   ```

   ⇒ 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:

   ```bash
   $TMT_ROOT/tools/tests/auto_diagnose.pl --cleanup \
   file.tmt.gz targetdir 'block1 block2'
   ```

3. Run the test as instructed:

   ```bash
   ./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 → Czenglish → ISI ReWrite → English (Cuřín, 2006)
• Move verbs to the end of the clause: English → Hindi → 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_converters 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 Blaž. 2008a. CzEng 0.7: Parallel Corpus with Community-Supplied Translations. In Proceedings of the Sixth International Language Resources and Evaluation (LREC’08), Marrakesh, Morocco. May. ELRA.


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)

<table>
<thead>
<tr>
<th>Translation of Haus</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>house</td>
<td>8,000</td>
</tr>
<tr>
<td>building</td>
<td>1,500</td>
</tr>
<tr>
<td>home</td>
<td>200</td>
</tr>
<tr>
<td>household</td>
<td>150</td>
</tr>
<tr>
<td>shell</td>
<td>50</td>
</tr>
</tbody>
</table>

Estimate translation probabilities
• Maximum likelihood estimation
  \[ p_f(e) = \begin{cases} 
  0.8 & \text{if } e = \text{house,} \\
  0.16 & \text{if } e = \text{building,} \\
  0.02 & \text{if } e = \text{home,} \\
  0.015 & \text{if } e = \text{household,} \\
  0.005 & \text{if } e = \text{shell.} 
\end{cases} \]

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

\[
\begin{array}{c|c|c|c|c}
1 & 2 & 3 & 4 \\
\hline
\text{das} & \text{Haus} & \text{ist} & \text{klein} \\
\text{the} & \text{house} & \text{is} & \text{small} \\
1 & 2 & 3 & 4 \\
\end{array}
\]
• Word positions are numbered 1-4

Alignment function
• Formalizing alignment with an alignment function
• Mapping an English target word at position i to a German source word at position j with a function \( a : i \rightarrow j \)
• Example
  \( a : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4 \} \)
Reordering
- Words may be reordered during translation

```
klein ist das Haus 4
the house is small
```

\( \alpha : \{1 \rightarrow 3, 2 \rightarrow 4, 3 \rightarrow 2, 4 \rightarrow 1 \} \)

One-to-many translation
- A source word may translate into multiple target words

```
das Haus ist klitzeklein 4
the house is very small
```

\( \alpha : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 4, 5 \rightarrow 4 \} \)

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

```
das Haus ist klein
```

```
house is small
```

\( \alpha : \{1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4 \} \)

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

```
null das Haus ist klein
```

```
the house is just small
```

\( \alpha : \{1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 4 \rightarrow 0, 5 \rightarrow 4 \} \)

IBM Model 1
- **Generative model**: break up translation process into smaller steps
  - IBM Model 1 only uses lexical translation
- Translation probability
  - for a foreign sentence \( f = (f_1, ..., f_L) \) of length \( L \)
  - to an English sentence \( e = (e_1, ..., e_l_e) \) of length \( l_e \)
  - with an alignment of each English word \( e_j \) to a foreign word \( f_i \), according to the alignment function \( \alpha : j \rightarrow i \)

\[
p(e, a|f) = \frac{e}{(l_e + 1)!} \prod_j t(e_j|a(f_i))
\]

- parameter \( \epsilon \) is a normalization constant

Example

```
das Haus ist klein
```

```
house is small
```

\( p(e|a|f) = \frac{0.7 \times 0.8 \times 0.8 \times 0.4}{4} = 0.0028 \)
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.

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

... la maison ... la maison blue ... la fleur ...
... the house ... the blue house ... the flower ...

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

... la maison ... la maison blue ... la fleur ...
... the house ... the blue house ... the flower ...

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

... la maison ... la maison blue ... la fleur ...
... the house ... the blue house ... the flower ...

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

... la maison ... la maison blue ... la fleur ...
... the house ... the blue house ... the flower ...

- Convergence
- Inherent hidden structure revealed by EM
**EM algorithm**

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

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

\[ p(\text{la} | \text{the}) = 0.453 \]
\[ p(\text{la} | \text{la}) = 0.324 \]
\[ p(\text{maison} | \text{house}) = 0.876 \]
\[ p(\text{bleu} | \text{blue}) = 0.563 \]

- Parameter estimation from the aligned corpus

---

**IBM Model 1 and EM**

**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**

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

**Alignments**

<table>
<thead>
<tr>
<th>La</th>
<th>Maision</th>
<th>The</th>
<th>House</th>
</tr>
</thead>
<tbody>
<tr>
<td>la</td>
<td>maison</td>
<td>the</td>
<td>house</td>
</tr>
<tr>
<td>bleu</td>
<td>bleu</td>
<td>bleu</td>
<td>bleu</td>
</tr>
</tbody>
</table>

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

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

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

---

**IBM Model 1 and EM: Expectation Step**

We need to compute \( p(\text{e} | \text{f}) \)

- Applying the chain rule

\[ p(\text{e} | \text{f}) = \frac{p(\text{e}, \text{a} | \text{f})}{p(\text{e} | \text{a}, \text{f})} \]

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

---

**IBM Model 1 and EM: Expectation Step**

We need to compute \( p(\text{e} | \text{f}) \)

\[
p(\text{e} | \text{f}) = \sum_{\text{a} | \text{a}} p(\text{e}, \text{a} | \text{f})
\]
\[
= \sum_{\text{a} | \text{a}} \frac{p(\text{e}, \text{a} | \text{f})}{p(\text{e} | \text{a}, \text{f})}
\]
\[
= \sum_{\text{a} | \text{a}} \frac{\sum_{\text{a} | \text{a}} p(\text{e}, \text{a} | \text{f})}{p(\text{e} | \text{a}, \text{f})}
\]
\[
= \sum_{\text{a} | \text{a}} \frac{\sum_{\text{a} | \text{a}} p(\text{e}, \text{a} | \text{f})}{p(\text{e} | \text{a}, \text{f})}
\]
\[
= \sum_{\text{a} | \text{a}} \frac{\sum_{\text{a} | \text{a}} p(\text{e}, \text{a} | \text{f})}{p(\text{e} | \text{a}, \text{f})}
\]
IBM Model 1 and EM: Expectation Step

\[ p(\theta|f) = \sum_{e} \frac{1}{s(f)+1} \prod_{i} \theta(e_i|f) \]

- 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

\[ \sum_{i=1}^{L} \sum_{j=0}^{I} \prod_{l=0}^{j} \frac{1}{p(f)} \prod_{l=0}^{j} \theta(e_i|f) = \sum_{i=1}^{L} \prod_{l=0}^{j} \theta(e_i|f) \]

\[ \text{for all } f \]

IBM Model 1 and EM: Maximization Step

- Combine what we have:
  \[ p(\theta|e, f) = \frac{p(\theta|e, f)}{p(\theta|f)} \]

\[ \sum_{e} \theta(e|f) \sum_{e} \theta(e|f) \]

\[ \prod_{e} \theta(e|f) \]

- After collecting these counts over a corpus, we can estimate the model:
  \[ \theta(e|f) = \frac{\sum_{g} \theta(e|g, f)}{\sum_{g} \theta(g|f)} \]

IBM Model 1 and EM: Pseudocode

1. initialization \( \theta(1) \) uniformly
2. do until convergence
   - set count(\( a|v \)) to 0 for all \( a, v \)
   - set total(e) to 0 for all \( e \)
   - for all word pairs \( (w, x) \)
     - total(\( x|w \)) \( \rightarrow \) \( \theta(1) \)
     - total(\( y|w \)) \( \rightarrow \) \( \theta(1) \)
   - for all word pairs \( (w, x) \)
     - total(\( y|w \)) \( \rightarrow \) \( \theta(1) \)
     - count(\( a|v \)) \( \rightarrow \) \( \theta(1) / \text{total}(e) \)
     - \( \text{total}(e) = \text{total}(e) + 1 \)
   - for all \( a \)
     - \( \theta(a|e) = \text{count}(a|e) / \text{total}(e) \)

MTMarathonWinterSchool, Lecture 2
27 Jan 2009
Higher IBM Models

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

- Only IBM Model 1 has global maximum
- Training of a higher IBM model builds on previous model
- Computationally biggest change in Model 3
  - Trick to simplify estimation does not work anymore
  - Exhaustive count collection becomes computationally too expensive
  - Sampling over high probability alignments is used instead

IBM Model 4

- Mary did not slap the green witch
- Mary not slap slap slap the green witch
- Mary did not slap slap slap NULL the green witch

Maria no daba una bofetada a la bruja verde

n(3|slap)
p-null

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
  - Function may return the same value for different input
    (one-to-many mapping)
  - 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

- 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
forenglishword e = 0...en
forforeignword f = 0...fn
if ( e aligned with f )
foreach 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):
forenglishword e-new = 0...en
forforeignword 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
### Statistical Machine Translation
- Components: Translation model, language model, decoder

#### Components
- **Foreign/English parallel text**
- **Statistical analysis**
- **Translation Model**
- **Language Model**
- **Decoding Algorithm**

### 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 | (e|f) | English | (e|f) |
|--------------------------|-----|---------|-----|
| the proposal            | 0.0627 | the suggestions | 0.0114 |
| a proposal              | 0.0186 | the support   | 0.0016 |
| an idea                 | 0.0280 | the idea      | 0.0001 |
| this proposal           | 0.0207 | the proposal  | 0.0008 |
| of the proposal         | 0.0150 | its proposal  | 0.0008 |
| the proposal            | 0.0100 | it           | 0.0008 |

### Decoding Process
- Build translation left to right
  - select foreign words to be translated

---

Winter School  
Day 3: Decoding / Phrase-based models  
MT Marathon  
28 January 2009

---

Statistical Machine Translation

Phrase-Based Translation

Phrase Translation Table

Decoding Process
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
  - mark foreign words as translated

- One to many translation

- Many to one translation

- 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

A Quick Word on Probabilities

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

Hypothesis Expansion

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

Hypothesis Expansion

- Add another hypothesis
Hypothesis Expansion

- Until all foreign words covered
  - Find best hypothesis that covers all foreign words
  - Backtrack to read off translation

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
  ⇒ Combine paths
    - Drop weaker path
    - Keep pointer from weaker path (for lattice generation)
  ⇒ 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 (affects 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

Maria no di una bofetada a la bruja verde

- 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 reranking model cost
    - LM * TM = \( p(\text{to}) \) * \( p(\text{the/to}) \) * \( p(\text{a/to}) \)

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

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

Limit 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:
  - Simple phrase lattice...
Moses: Open Source Toolkit

- Open source statistical machine translation system developed from scratch 2006
  - state-of-the-art phrase-based approach
  - novel methods: factorized translation models, confusion network decoding
  - support for very large models through memory-efficient data structures

- 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

- Major components of phrase-based model
  - phrase translation model \( \phi(f|e) \)
  - reordering model \( \omega \)
  - language model \( p_{lm}(e) \)

- Bayes rule
  \[
  \arg\max_e p(e|f) = \arg\max_e p(f|e) p_{lm}(e) \omega
  \]

- Sentence \( f \) is decomposed into \( I \) phrases \( \bar{f}_1, \ldots, \bar{f}_I \)

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-based translation model

- \( \phi(f|e) \) is a distribution over \( e \)
- \( \omega \) is a permutation of \( e \)
- \( p_{lm}(e) \) is a language model

- Bayes rule
  \[
  \arg\max_e p(e|f) = \arg\max_e p(f|e) p_{lm}(e) \omega
  \]

- Sentence \( f \) is decomposed into \( I \) phrases \( \bar{f}_1, \ldots, \bar{f}_I \)

- Decomposition of \( \phi(f|e) \)
  \[
  \phi(\bar{f}|\bar{e}) = \prod_{i=1}^I \phi(\bar{f}_i|\bar{e}_i) \omega
  \]

Phrase translation table

- Phrase translations for der Vorschlag

| German       | English       | \( \phi(e|f) \) |
|--------------|---------------|----------------|
| der Vorschlag| the proposal  | 0.0207         |
| der Vorschlag| the proposals | 0.0159         |
| der Vorschlag| a proposal    | 0.0191         |
| der Vorschlag| the proposals | 0.0114         |
| der Vorschlag| the idee      | 0.0227         |
| der Vorschlag| the idee of   | 0.0091         |
| der Vorschlag| the ideea    | 0.0091         |
| der Vorschlag| its proposal  | 0.0206         |
| der Vorschlag| this proposal | 0.0088         |
| der Vorschlag| the proposals | 0.0088         |
| der Vorschlag| the ideea of  | 0.0088         |
How to learn the phrase translation table?

- Start with the word alignment:

![Word alignment matrix]

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

  
- Consistent with word alignment:

  $$\begin{align*}
  (\text{Maria, Mary}), (\text{no, did not}), (\text{alabrujaverde}, \text{the green witch}), (\text{una bofetada}, \text{the slap}), (\text{ala}, \text{the}), (\text{bruja, witch}), (\text{verde, green}), \\
  (\text{Mariano, Mary did not}), (\text{nodabaunabofetada, did not slap}), (\text{alabrujaverde, the green witch}), (\text{Mariano dabauna bofetada, Mary did not slap}), (\text{alabrujaverde, the green witch})
  \end{align*}$$
Word alignment induced phrases (5)

(Mariano, Mary), (yes, did not), (slap, daba una bofetada), (ala, the), (bruja, witch), (verde, green).
(Mariano no, Mary did not), (yes, daba una bofetada a la, slap the), (bruja verde, green witch).
(Mariano no daba una bofetada a la, Mary did not slap), (ala bruja verde, the green witch).
(Mariano daba una bofetada a la, Mary did not slap the), (daba una bofetada a la bruja verde, slap the green witch).
(Mariano no daba una bofetada a la bruja verde, did not slap the green witch).
(Mariano daba una bofetada a la bruja verde, Mary did not slap the green witch).

Probability distribution of phrase pairs

- We need a probability distribution $\phi(f|e)$ over the collected phrase pairs
- Possible choices
  - relative frequency of collected phrases: $\phi(f|e) = \frac{\text{count}(f,e)}{\text{count}(\cdot,e)}$
  - or, conversely $\phi(e|f)$
- 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 $\omega^n$
- Lexicalized reordering model

Lexicalized reordering models

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

Learning lexicalized reordering models

- Orientation type is learned during phrase extractions
- Alignment point to the top left (monotone) or top right (swap)?
- For more, see [Koehn et al., 2005] or [Koehn et al., 2005]
TectoMT
Software framework for developing MT systems (and other NLP applications)

Zdeněk Žabokrtský
ÚFAL MFF UK

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

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)

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.

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: We would have gone into forest]

Tectogrammar in a nutshell

- tectogrammatical layer of language representation
  - introduced by Petr Sigal 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 pre-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.
**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...)

**MT triangle in vivo**

- Illustration: analysis-transfer-synthesis in TectoMT

**How could tecto help?**

- Vague assumption:
  - Tectogrammar 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?

**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 other grammar rules
  - Inflectional endings, agglutivitive affixes, functional words, word order, punctuation orthographic rules...
  - ...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 exploitation of the (limited) training data

**Tecto transfer factorization**

- Three transfer "channels" can be separated:
  - Translation of lexicalization
  - E.g. keep't goes to buy
  - Translation of syntacticization
  - 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.

**Tecto transfer factorization (cont.)**

- Example: three ways to express future tense in Czech
  - (1) aux. verb: budu...chodí - I will walk...
  - (2) prefix: paletím - I will fly...
  - (3) ending: uvařím - I will boil...

- Nontrivial tense translation from the n-gram view

- But once we work with tecto analysis, 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.
Tecto transfer factorization (cont.)

- we introduce the notion of formemes - morphosyntactic language means expressing the dependency relation
- example values:
  - form (in Polish): semantic noun which is on the surface expressed in the form of prepositional group in locative with preposition "of"
  - form (in English): semantic verb expressed in active voice or as a head of subordinate clause introduced with the sub-conjunction "that"
  - both (in Polish): main in subject position
  - adj (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

<table>
<thead>
<tr>
<th>form</th>
<th>K2</th>
<th>adj</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.67</td>
<td>0.93</td>
<td>0.91</td>
</tr>
<tr>
<td>1.00</td>
<td>1.00</td>
<td>0.99</td>
</tr>
<tr>
<td>0.95</td>
<td>0.99</td>
<td>0.98</td>
</tr>
<tr>
<td>0.84</td>
<td>0.89</td>
<td>0.88</td>
</tr>
</tbody>
</table>

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

Using tree context (cont.)

Example 1:
- The grass around your house should be cut soon.
- Google translation: Trávy kolem vašeho domu by se mělo snížit
- 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

<table>
<thead>
<tr>
<th>Example 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zitra se vašte Synat! Trejice kudy brat Mare a Hana.</td>
</tr>
<tr>
<td>Google translation: Tomorrow is the Holy Trinity church will take Mary and John</td>
</tr>
<tr>
<td>Incorrect lexical choice: presence of the &quot;se&quot; clitic at the close-second position is crucial, but it is too far</td>
</tr>
</tbody>
</table>

Hybrid MT with TectoMT

- Hybrid MT with TectoMT (cont.)

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)
Part II:

TectoMT System

Architecture

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

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, tags, lemmatizers etc.
  - just provide them with a Perl "wrapper" with the prescribed interface

Layers of sentence description

- in each bundle, there can be at most one tree for each "layer"
- set of possible layers: \( (S,T) \times \{\text{English, Czech, ...}\} \times \{M,P,A,T,N\} \)
  - 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

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 topics 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 attribute-value pair

Hierarchy of processing units

- black
  - the smallest individually executable unit
  - with well-defined input and output
  - block parameterization 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 to the data
    3. conversion into the output format
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.
    - English::Czech::Sentence
    - English::Czech::Sentence::Blocks
    - English::Czech::Sentence::Blocks::Tags
  - 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 consists of roughly 80 blocks

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
    - Brants’s TnT tagger, Schmid’s Tree tagger, Coburn’s
      - English::Czech::Sentence::Blocks::Tags
    - parsers
      - Collins’ phrase structure parser, McDonald’s 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 C5u-based C5u labeller, Ptáček’s C5u-based C5u labeller

Other TectoMT components
- "core" - Perl libraries forming the core of TectoMT infrastructure, esp. for memory representation of (and interface to) the data structures
- numerous file-format converters (e.g. from PDT, Penn treebank, Czeng corpus, WMT shared task data etc. to our xml format)
- TectoMT-customized Pajá’s Tree Editor TEd
- 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...

TectoMT directory structure
- everything under one directory tree specified in system variable TMT_ROOT
  - versioned part (in a svn repo)
    - share/installed_libs/
      - lib/(core, blocks, packageed, other)/
      - tools/
      - applications/
      - doc/
      - perldoc/
      - tests/
      - training/
      - release_builds/
      - evaluation/
  - shared part (unversioned)
    - share/tred/
      - share/data/models, resources...
      - share/installed_tools/
      - share/data/modeled, resources...

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
Prague English Dependency Treebank

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 CaEn (parsed
  Czech-English parallel corpus, around 60kW)

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

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

Preproc. data for PEDT
- Prague English Dependency Treebank
  - PDT-style annotation project at UFAL
  - currently 10000 English tectogrammatically analyzed sentences,
  - saving annotators’ work by outsourcing a part of the analysis
    in TectoMT

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 for finding bugs anywhere along the scenario
  - very preliminary stage

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.
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:
  * Extracting probabilistic translation dictionary from tectogrammatically aligned parallel corpora.

Tecto-alignment x word-alignment

T-Aligner

- Greedy algorithm based on features
- A score is assigned to each possible connection (pair of Czech and English node)
  \[ \text{score}(en, cs) = \sum_w f(en, cs) \]
- The weights 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 (EU-news, 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
- \( \text{lex rf} \) – attribute of a tectogrammatical node, refers to the analytical node from which it acquired its lexical meaning.

<table>
<thead>
<tr>
<th>Aligner</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our T-aligner</td>
<td>88.5 %</td>
</tr>
<tr>
<td>GIZA++ word-alignment transferred to t-trees</td>
<td>85.7 %</td>
</tr>
<tr>
<td>Our T-aligner using also GIZA++ word-alignment</td>
<td>91.0 %</td>
</tr>
<tr>
<td>(Inter-annotator agreement)</td>
<td>94.8 %</td>
</tr>
</tbody>
</table>

Tectogrammatical alignment results
References


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...

# Asociace uvedla, že domácí poptávka v září stoupla.
# The association said domestic demand grew in September.

2: ... Decompose into Treelets...

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:

Synchronous Tree Substitution Grammar, e.g. Čmejrek (2006).
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 Scores for STSG Transfer
- Identical decoder, only the structure + node labels differ.

<table>
<thead>
<tr>
<th>Layers \ Language Models</th>
<th>no LM</th>
<th>with LM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ecpp, atomic nodes</td>
<td>8.65±0.55</td>
<td>10.90±0.63</td>
</tr>
<tr>
<td>eaca, atomic nodes</td>
<td>6.59±0.52</td>
<td>8.75±0.61</td>
</tr>
<tr>
<td>etc, generated attrs, fixed structure</td>
<td>5.31±0.53</td>
<td>5.61±0.50</td>
</tr>
<tr>
<td>etc, atomic nodes, all attributes</td>
<td>1.61±0.33</td>
<td>2.56±0.35</td>
</tr>
<tr>
<td>etc, atomic nodes, just t-lemmas</td>
<td>0.67±0.19</td>
<td>-</td>
</tr>
</tbody>
</table>

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-best list varies in unimportant attributes.

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.

Don’t Dump Deep Syntax Yet

WMT08 Results

<table>
<thead>
<tr>
<th>BLEU Rank</th>
<th>In-domain</th>
<th>Out-of-domain</th>
<th>BLEU Rank</th>
<th>In-domain</th>
<th>Out-of-domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factored Moses</td>
<td>15.91</td>
<td>-2.62</td>
<td>11.93</td>
<td>-2.89</td>
<td></td>
</tr>
<tr>
<td>PC Translator</td>
<td>8.48</td>
<td>-2.78</td>
<td>8.41</td>
<td>-2.60</td>
<td></td>
</tr>
<tr>
<td>TectoMT</td>
<td>9.28</td>
<td>-2.79</td>
<td>9.64</td>
<td>-3.26</td>
<td></td>
</tr>
<tr>
<td>Vanilla Moses</td>
<td>12.96</td>
<td>-3.33</td>
<td>9.64</td>
<td>-3.26</td>
<td></td>
</tr>
<tr>
<td>etc</td>
<td>4.98</td>
<td>-3.36</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Harian Rank: 1-2, 2-3, 3-5

- TectoMT ranked comparably to vanilla Moses (= BLEU is using anywhere).
- TectoMT great for preparing rich data.

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

<table>
<thead>
<tr>
<th>BLEU Rank</th>
<th>WMT08 Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline Moses, Distance Reordering</td>
<td>18.8±0.25</td>
</tr>
<tr>
<td>Baseline Moses, Reordering Using en+hi Forms</td>
<td>19.77±0.33</td>
</tr>
<tr>
<td>Suffix LM + Reord</td>
<td>20.99±0.18</td>
</tr>
<tr>
<td>Rule-based Reordering + Suffix LM + Reord</td>
<td>21.01±0.18</td>
</tr>
</tbody>
</table>

Join TectoMT tutorial lab session for SVO—SOV in 12 lines of Perl.
NLPHacking: Valency Information

Bring non-local information closer based on dependency edges:

# The associations aided domestic demand grew in September.

To produce "verbosetokens":

- the | said
- said | said
- domestic | grew
- grew | grew
- in | September

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

Baseline 9.77 ± 0.69 14.57 ± 0.83
With source valency 9.98 ± 0.67 14.52 ± 0.85

Thu 29, 2009 Bad News, NLPHacking and Feature Fishing 12

Fishing: Phrase Table Marking

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

Thu 29, 2009 Bad News, NLPHacking and Feature Fishing 13

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: Marie, Mari, Marii, ... .
- Given "Mary" is object, "Marii,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.

Thu 29, 2009 Bad News, NLPHacking and Feature Fishing 14

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 NLPHacking.

Thu 29, 2009 Bad News, NLPHacking and Feature Fishing 15

References


NLP Association of India.


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.

Prerequisites

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 \( \{S,T\} \times \{\text{English}, \text{Czech}, \ldots\} \times \{W,M,P,A,T,\ldots\} \), 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).
- \( \{\text{English}, \text{Czech}, \ldots\} \) represents the language in question
- \( \{W,M,P,A,T,\ldots\} \) 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).

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 \( \{S,T\} \times \{\text{English}, \text{Czech}, \ldots\} \times \{W,M,P,A,T,\ldots\} \), 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).
- \( \{\text{English}, \text{Czech}, \ldots\} \) represents the language in question
- \( \{W,M,P,A,T,\ldots\} \) 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 $\text{SEnglishA}$ 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

\begin{verbatim}
SEnglishM_to_SEnglishA::McD_parser_local \\
\end{verbatim}

with

\begin{verbatim}
SEnglishM_to_SEnglishA::McD_parser_local TMT_PARAM_MCD_EN_MODEL=conll_mcd_order2_0.1.model \\
\end{verbatim}

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

\begin{verbatim}
tmttred sample.tmt
\end{verbatim}

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.

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

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

\begin{verbatim}
./run_all.sh
\end{verbatim}

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 (`$EnglishA`), 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?
The birth of SMT: generative models

- The definition of translation probability follows a mathematical derivation
  \[ \arg\max_{\text{e}} p(\text{e}|\text{f}) = \arg\max_{\text{e}} p(\text{f}|\text{e}) p(\text{e}) \]

- Occasionally, some independence assumptions are thrown in
  for instance IBM Model 1: word translations are independent of each other
  \[ p(\text{f}, a) = \frac{1}{Z} \prod_i p(\text{f}_i | \text{a}_i) \]

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

Log-linear models

- IBM Models provided mathematical justification for factoring components together
  \[ p_{LM} \times p_{TM} \times p_{D} \]

- These may be weighted
  \[ \sum_i \lambda_i p(\text{e}_i) \]

- Many components \( p_i \) with weights \( \lambda_i \)
  \[ \prod_i p_i = e^{\sum_i \lambda_i \log(p_i)} \]
  \[ \log \prod_i p_i = \sum_i \lambda_i \log(p_i) \]

Knowledge sources

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

Set feature weights

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

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

- Generate n-best list
- Score translations
- Find feature weights that move up good translations
- Change feature weights

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 consists 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

<table>
<thead>
<tr>
<th>Translation</th>
<th>Feature values</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>He does not go home</td>
<td>-32.36</td>
<td>9.43</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-34.03</td>
<td>10.02</td>
</tr>
<tr>
<td>He is not packing</td>
<td>-32.59</td>
<td>8.12</td>
</tr>
<tr>
<td>He is not packing</td>
<td>-32.56</td>
<td>9.14</td>
</tr>
<tr>
<td>He does nothome</td>
<td>-32.67</td>
<td>9.15</td>
</tr>
<tr>
<td>He is not packing</td>
<td>-34.59</td>
<td>10.05</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-32.28</td>
<td>8.05</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-32.78</td>
<td>9.08</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-32.54</td>
<td>8.04</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-33.14</td>
<td>9.09</td>
</tr>
<tr>
<td>He does nothome</td>
<td>-32.75</td>
<td>9.10</td>
</tr>
<tr>
<td>He is not packing</td>
<td>-32.58</td>
<td>8.07</td>
</tr>
<tr>
<td>He is not for home</td>
<td>-32.84</td>
<td>9.08</td>
</tr>
</tbody>
</table>

Och’s minimum error rate training (MERT)

- Line search for best feature weights

- Find best feature weight

Find Best Feature Weight

- Core task:
  - Find optimal value for one parameter weight \( \lambda \)
  - While leaving all other weights constant
- Score of translation \( i \) for a sentence \( f \):
  \[ p(e_i|f) = \lambda a_i + b_i \]
- Recall that:
  - We deal with 100s or 1000s of sentences \( f \)
  - We are trying to find the value \( \lambda \) so that over all sentences, the error score is optimized

Translations for one Sentence

- Each translation is a line \( p(e_i|f) = \lambda a_i + b_i \)
- The model-best translation for a given \( \lambda \) (x-axis), is highest line at that point
- There are a few threshold points \( \lambda^\ast \), where the model-best line changes
Finding the Optimal Value for $\lambda$

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

$\Rightarrow$ 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

<table>
<thead>
<tr>
<th>component</th>
<th>run 1</th>
<th>run 2</th>
<th>run 3</th>
<th>run 4</th>
<th>run 5</th>
<th>run 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>lexdist1</td>
<td>0.093565</td>
<td>0.044724</td>
<td>0.097312</td>
<td>0.108922</td>
<td>0.108922</td>
<td>0.062848</td>
</tr>
<tr>
<td>lexdist2</td>
<td>0.021165</td>
<td>0.008882</td>
<td>0.008607</td>
<td>0.013950</td>
<td>0.013950</td>
<td>0.030890</td>
</tr>
<tr>
<td>lexdist3</td>
<td>0.083298</td>
<td>0.049741</td>
<td>0.024822</td>
<td>-0.000598</td>
<td>-0.000598</td>
<td>0.023018</td>
</tr>
<tr>
<td>lexdist4</td>
<td>0.051842</td>
<td>0.108107</td>
<td>0.090298</td>
<td>0.111243</td>
<td>0.111243</td>
<td>0.047508</td>
</tr>
<tr>
<td>lexdist5</td>
<td>0.043290</td>
<td>0.047801</td>
<td>0.020211</td>
<td>0.028672</td>
<td>0.028672</td>
<td>0.050748</td>
</tr>
<tr>
<td>lexdist6</td>
<td>0.083848</td>
<td>0.056161</td>
<td>0.103767</td>
<td>0.032869</td>
<td>0.032869</td>
<td>0.050240</td>
</tr>
<tr>
<td>lm1</td>
<td>0.052111</td>
<td>0.045096</td>
<td>0.046655</td>
<td>0.054519</td>
<td>0.054519</td>
<td>0.046538</td>
</tr>
<tr>
<td>lm2</td>
<td>0.052888</td>
<td>0.036831</td>
<td>0.040820</td>
<td>0.058003</td>
<td>0.058003</td>
<td>0.066308</td>
</tr>
<tr>
<td>lm3</td>
<td>0.042151</td>
<td>0.066256</td>
<td>0.043265</td>
<td>0.047271</td>
<td>0.047271</td>
<td>0.052853</td>
</tr>
<tr>
<td>lm4</td>
<td>0.034067</td>
<td>0.031048</td>
<td>0.050794</td>
<td>0.037589</td>
<td>0.037589</td>
<td>0.031939</td>
</tr>
<tr>
<td>phrase-pen.</td>
<td>0.059151</td>
<td>0.062019</td>
<td>-0.037950</td>
<td>0.023414</td>
<td>0.023414</td>
<td>-0.069425</td>
</tr>
<tr>
<td>word-pen</td>
<td>-0.200963</td>
<td>-0.249531</td>
<td>-0.247089</td>
<td>-0.228469</td>
<td>-0.228469</td>
<td>-0.252579</td>
</tr>
</tbody>
</table>

Unstable outcomes: scores vary

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

<table>
<thead>
<tr>
<th>run</th>
<th>iterations</th>
<th>dev score</th>
<th>test score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>50.16</td>
<td>51.99</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>50.27</td>
<td>51.78</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>50.13</td>
<td>51.50</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>50.15</td>
<td>51.43</td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>50.09</td>
<td>51.60</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>50.25</td>
<td>51.10</td>
</tr>
<tr>
<td>7</td>
<td>11</td>
<td>50.31</td>
<td>51.32</td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>50.42</td>
<td>51.79</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>50.42</td>
<td>51.79</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>50.31</td>
<td>51.32</td>
</tr>
</tbody>
</table>

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
- 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

```plaintext
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 [Ittycheria and Salim Roukos, 2007]

Problem: reference translation

- Reference translation may be anywhere in this box

  all English sentences
  producable by model
  covered by search

- If producable by model — we can compute feature scores
- If not — we can not

Some solutions

- Skip sentences, for which reference cannot 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

- Skippingsentences with unproduceable reference hurts
  
<table>
<thead>
<tr>
<th>Handling of reference</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>with skipping</td>
<td>25.81</td>
</tr>
<tr>
<td>w/o skipping</td>
<td>29.81</td>
</tr>
</tbody>
</table>

- 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 wordorder 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
  
<table>
<thead>
<tr>
<th>Reference:</th>
<th>The group attended the meeting in Najaf ...,</th>
<th>System:</th>
<th>The group meeting was attended in UNKNOWN ...,</th>
</tr>
</thead>
<tbody>
<tr>
<td>~ only update features involved in this part</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 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 producable
  - 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
  - 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 representation of words

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>word</td>
<td>word</td>
</tr>
<tr>
<td>lemma</td>
<td>lemma</td>
</tr>
<tr>
<td>part-of-speech</td>
<td>part-of-speech</td>
</tr>
<tr>
<td>morphology</td>
<td>morphology</td>
</tr>
<tr>
<td>word class</td>
<td>word class</td>
</tr>
</tbody>
</table>

- 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.)
- 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.)
  [Doh et al. 2004, Koehn and Knight, 2005]
  → we pursue an integrated approach
- Use of syntactic tree structure
  → may be combined with our approach

Decomposing translation: example

- Translate lemma and syntactic information separately

Translation process: example

Input: (Autos, Auto, NNS)
1. Translation step: lemma ⇒ lemma
   (7, car, 7), (7, auto, 7)
2. Generation step: lemma ⇒ part-of-speech
   (7, car, NN), (7, car, NNS), (7, auto, NN), (7, auto, NNS)
3. Translation step: part-of-speech ⇒ part-of-speech
   (7, car, NN), (7, car, NNS), (7, auto, NNP), (7, auto, NNS)
4. Generation step: lemma, part-of-speech ⇒ surface
   (car, car, NN), (car, car, NNS), (auto, auto, NNP), (auto, 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

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 man sleeps
  - count(ADV), count(V), count(NNP)
  - evidence for probability distributions (max. likelihood estimation)
  - p(ADV|the), p(V|sleeps), p(NNP|man), p(V|sleeps), p(V|sleeps|the)

Factored training

- Annotate training with factors, extract phrase

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 man sleeps
  - count(ADV), count(V), count(NNP)
  - evidence for probability distributions (max. likelihood estimation)
  - p(ADV|the), p(V|sleeps), p(NNP|man), p(V|sleeps), p(V|sleeps|the)
**Factored Translation Models**

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

---

**Phrase-based translation**

- Task: translate this sentence from German into English

  er geht ja nicht nach hause

---

**Translation step 1**

- Task: translate this sentence from German into English

  er geht ja nicht nach hause

- Pick phrase in input, translate

---

**Translation step 2**

- Task: translate this sentence from German into English

  er geht ja nicht nach hause

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

---

**Translation step 3**

- Task: translate this sentence from German into English

  er geht ja nicht nach hause

- Pick phrase in input, translate

---

**Translation step 4**

- Task: translate this sentence from German into English

  er geht ja nicht nach hause

- Pick phrase in input, translate
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

The machine translation decoder does not know the right answer

→ Search problem solved by heuristic beam search
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: house|NN|neutral|plural|nominative
  --> { houses|NN|plural, homes|NN|plural, buildings|NN|plural, shells|NN|plural }
- Each time, a hypothesis is expanded, these mapping steps have to be 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

- Problem: Explosion of translation options
  - originally limited to 20 per input phrase
  - even with simple model, now 100s 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

Output Input

- 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
  
<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>18.04</td>
</tr>
<tr>
<td>surface + POS</td>
<td>18.15</td>
</tr>
</tbody>
</table>

- German–English, News Commentary data (WMT 2007), 1 million word
  
<table>
<thead>
<tr>
<th>Model</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>18.19</td>
</tr>
<tr>
<td>With POS LM</td>
<td>19.05</td>
</tr>
</tbody>
</table>

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

Sequence models over morphological tags

- die (the) | heller (bright)
- Sterne (stars) | erhellten (illuminated)
- das (the) | schwarz (black)
- Himmel (sky)

- Violation of noun phrase agreement in gender
  - das schwarze and schwarz 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: \( p(N\text{-male} | J\text{-male}) > p(N\text{-male} | J\text{-neutral}) \)

Local agreement (esp. within noun phrases)

- High order language models over POS and morphology
- Motivation:
  - DET-ag NOUN-agl good sequence
  - DET-agl NOUN/plural bad sequence

Agreement within noun phrases

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Agreement errors in NP</th>
<th>devtest</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>factored model</td>
<td>4% in NP &lt; 3 words</td>
<td>18.25 BLEU</td>
<td>18.22 BLEU</td>
</tr>
<tr>
<td>baseline</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- Example:
  - baseline: ... zur zwischenstaatlichen methoden ...
  - factored model: ... zu zwischenstaatlichen methoden ...

- Example:
  - baseline: ... das zweite wichtige \( \tilde{\text{a}} \)nderung ...
  - factored model: ... die zweite wichtige \( \tilde{\text{a}} \)nderung ...

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)

<table>
<thead>
<tr>
<th>System</th>
<th>In-domain</th>
<th>Out-of-domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>18.19</td>
<td>15.01</td>
</tr>
<tr>
<td>With POS LM</td>
<td>19.05</td>
<td>15.03</td>
</tr>
<tr>
<td>Morphgen model</td>
<td>14.38</td>
<td>11.65</td>
</tr>
</tbody>
</table>

- 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

Adding annotation to the source

- Source words may lack sufficient information to map phrases
  - English-German: what case for noun phrases?
  - Chinese-English: plural or singular
  - pronoun translation: what do they refer to?

- Idea: add additional information to the source that makes the required information available locally (where it is needed)
- see [Avramidis and Koehn, ACL 2008] for details

Case Information for English–Greek

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

Obtaining Case Information

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

- Automatic BLEU scores
  
<table>
<thead>
<tr>
<th>System</th>
<th>dev</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>18.13</td>
<td>18.05</td>
</tr>
<tr>
<td>enriched</td>
<td>18.21</td>
<td>18.20</td>
</tr>
</tbody>
</table>

- Improvement in verb inflection

<table>
<thead>
<tr>
<th>System</th>
<th>Verb count</th>
<th>Errors</th>
<th>Missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>311</td>
<td>19.0%</td>
<td>7.4%</td>
</tr>
<tr>
<td>enriched</td>
<td>294</td>
<td>5.4%</td>
<td>2.7%</td>
</tr>
</tbody>
</table>

- Improvement in noun phrase inflection

<table>
<thead>
<tr>
<th>System</th>
<th>NPs</th>
<th>Errors</th>
<th>Missing</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>247</td>
<td>8.1%</td>
<td>3.2%</td>
</tr>
<tr>
<td>enriched</td>
<td>239</td>
<td>5.0%</td>
<td>5.0%</td>
</tr>
</tbody>
</table>

- Also successfully applied to English-Czech MT Marathon Winter School, Lecture 5 30 January 2009

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]

Shallow syntactic features

- the paintings of the old man are beautiful

<table>
<thead>
<tr>
<th>the</th>
<th>paintings</th>
<th>of</th>
<th>the</th>
<th>old</th>
<th>man</th>
<th>are</th>
<th>beautiful</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-NP</td>
<td>I-NP</td>
<td>L-NP</td>
<td>L-NP</td>
<td>L-NP</td>
<td>L-NP</td>
<td>L-NP</td>
<td>V</td>
</tr>
<tr>
<td>SBJ</td>
<td>OBJ</td>
<td>OBJ</td>
<td>OBJ</td>
<td>OBJ</td>
<td>OBJ</td>
<td>OBJ</td>
<td>V</td>
</tr>
</tbody>
</table>

- 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]