# Wild Experimenting in Machine Translation

Ondřej Bojar, Aleš Tamchyna, Jan Berka
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University, Prague

Wed Feb 15, 2012

# Outline

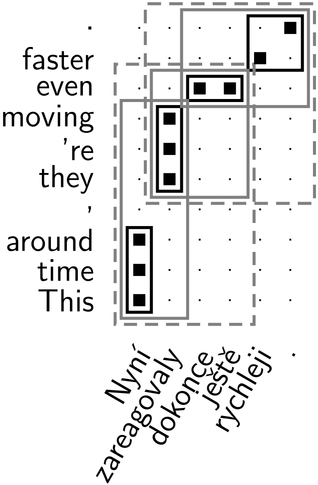
Statistical Machine Translation:

- Word order issues:
    - of PBMT, RBMT and hierarchical MT.
- Morphology issues of PBMT:
    - Along the whole MT pipeline.
    - With focus on target-side rich morphology.

Wild Experimenting:

- Motivation for experiment management.
- Key features of Eman.

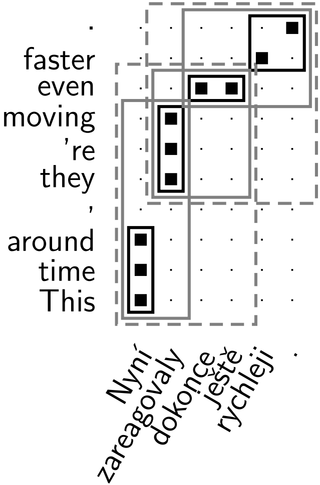
# Phrase-Based Machine Translation



**Training data**:

- a parallel corpus (Czech sent = English sent)
- automatic word alignment (Czech word ∼ English word)

# Phrase-Based Machine Translation

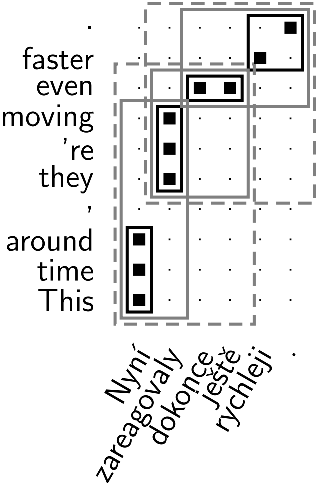


This time around = Nyní
they 're moving = zareagovaly
even = dokonce ještě
even faster = dokonce ještě rychleji
... = ...

**Training data**:

- a parallel corpus (Czech sent = English sent)
- automatic word alignment (Czech word ∼ English word)

# Phrase-Based Machine Translation



This time around = Nyní
they 're moving = zareagovaly
even = dokonce ještě
even faster = dokonce ještě rychleji
. . . = . . .

**Training data**:

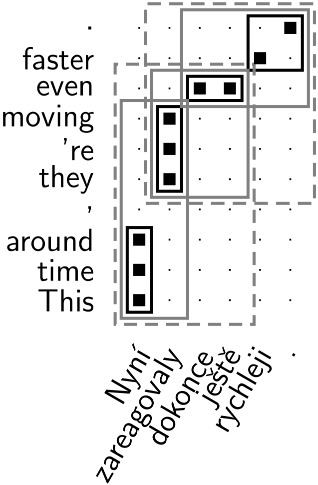
- a parallel corpus (Czech sent = English sent)
- automatic word alignment (Czech word ∼ English word)

**When translating** we search for:

- such a segmentation of the input sentence into "phrases"
- and such phrase translations

to make the output most probable.

# Phrase-Based Machine Translation



This time around = Nyní
they 're moving = zareagovaly
even = dokonce ještě
even faster = dokonce ještě rychleji
. . . = . . .

**Training data**:

- a parallel corpus (Czech sent = English sent) . . . 9 mil. sent. pairs
- automatic word alignment (Czech word ∼ English word) ∼ 2×90 M

**When translating** we search for:

- such a segmentation of the input sentence into "phrases"
- and such phrase translations

to make the output most probable.

# Warm-Up: Prove Google is Phrase-Based

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |

# Warm-Up: Prove Google is Phrase-Based

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |

# Warm-Up: Prove Google is Phrase-Based

Word form variations:

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |
| Proč natáhl bačkory? | Why stretched slippers? | × |

# Warm-Up: Prove Google is Phrase-Based

Word form variations:

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |
| Proč natáhl bačkory? | Why stretched slippers? | × |

Pumping words into phrases:

| | | |
|---|---|---|
| Jan s Marií se vzali. | John and Mary were married. | ✓ |

# Warm-Up: Prove Google is Phrase-Based

Word form variations:

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |
| Proč natáhl bačkory? | Why stretched slippers? | ✗ |

Pumping words into phrases:

| | | |
|---|---|---|
| Jan s Marií se vzali. | John and Mary were married. | ✓ |
| Jan s Marií se včera vzali. | John and Mary married yesterday. | ✓ |

# Warm-Up: Prove Google is Phrase-Based

Word form variations:

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |
| Proč natáhl bačkory? | Why stretched slippers? | × |

Pumping words into phrases:

| | | |
|---|---|---|
| Jan s Marií se vzali. | John and Mary were married. | ✓ |
| Jan s Marií se včera vzali. | John and Mary married yesterday. | ✓ |
| Jan s Marií se včera v kostele vzali. | John and Mary are married in church yesterday. | ∼ |

# Warm-Up: Prove Google is Phrase-Based

## Word form variations:

| | | |
|---|---|---|
| Natáhnout bačkory. | Kick the bucket. | ✓ |
| Proč musel natáhnout bačkory? | Why did he kick the bucket? | ✓ |
| Proč natáhl bačkory? | Why stretched slippers? | × |

## Pumping words into phrases:

Jan s Marií se vzali.

John and Mary were married. ✓

Jan s Marií se včera vzali.

John and Mary married yesterday. ✓

Jan s Marií se včera v kostele vzali.

John and Mary are married in church yesterday. ~

Jan s Marií se včera v kostele svatého Ducha vzali.

John and Mary yesterday in the Church of the Holy Spirit took. ×

# PBMT vs. RBMT

(Prove Systran is not phrase-based.)

# PBMT vs. RBMT

(Prove Systran is not phrase-based.)

| | | |
|---|---|---|
| | Stell dir das vor. | |
| Google | Imagine that. | √ |
| Systran | Imagine. | √ |

# PBMT vs. RBMT

(Prove Systran is not phrase-based.)

| | | |
|---|---|---|
| | Stell dir das vor. | |
| Google | Imagine that. | ✓ |
| Systran | Imagine. | ✓ |
| | Stell dir ein Haus vor. | |
| Google | Imagine a house before. | × |
| Systran | Imagine a house. | ✓ |

# PBMT vs. RBMT

(Prove Systran is not phrase-based.)

| | | |
|---|---|---|
| | Stell dir das vor. | |
| Google | Imagine that. | ✓ |
| Systran | Imagine. | ✓ |
| | Stell dir ein Haus vor. | |
| Google | Imagine a house before. | × |
| Systran | Imagine a house. | ✓ |
| | Stell dir ein kleines Haus vor. | |
| Google | Imagine a small house in front. | × |
| Systran | Imagine a small house. | ✓ |

# PBMT vs. RBMT

(Prove Systran is not phrase-based.)

| | | |
|---|---|---|
| | Stell dir das vor. | |
| Google | Imagine that. | √ |
| Systran | Imagine. | √ |
| | Stell dir ein Haus vor. | |
| Google | Imagine a house before. | × |
| Systran | Imagine a house. | √ |
| | Stell dir ein kleines Haus vor. | |
| Google | Imagine a small house in front. | × |
| Systran | Imagine a small house. | √ |
| | Stell dir ein kleines Haus mit vierzehn Fenster vor. | |
| Google | Imagine a small house with fourteen windows in front. | × |
| Systran | Imagine a small house with fourteen windows. | √ |

# Limitations of RBMT

- "Pump" grammatical constructions, not just words.

# Limitations of RBMT

- “Pump” grammatical constructions, not just words.

Stell dir ein Haus vor.
$\Rightarrow$ Imagine a house. ✓

# Limitations of RBMT

- "Pump" grammatical constructions, not just words.

Stell dir ein Haus vor.
⇒ Imagine a house. ✓

Stell dir ein Haus, das einen Garten hat, vor.
⇒ Imagine a house, which has a garden. ✓

# Limitations of RBMT

- "Pump" grammatical constructions, not just words.

Stell dir ein Haus vor.
⇒ Imagine a house. ✓

Stell dir ein Haus, das einen Garten hat, vor.
⇒ Imagine a house, which has a garden. ✓

Stell dir ein Haus, das einen Garten, der berühmt ist, hat, vor.
⇒ Place to you a house, which a garden, which has is famous, forwards. ×

# Limitations of RBMT

- “Pump” grammatical constructions, not just words.

Stell dir ein Haus vor.
⇒ Imagine a house. ✓

Stell dir ein Haus, das einen Garten hat, vor.
⇒ Imagine a house, which has a garden. ✓

Stell dir ein Haus, das einen Garten, der berühmt ist, hat, vor.
⇒ Place to you a house, which a garden, which has is famous, forwards. ×

- What’s worse: non-grammatical input breaks it.

Stell dir ein Haus, das ⊘ Garten hat, vor.
⇒ Place to you a house, the garden intends. ×

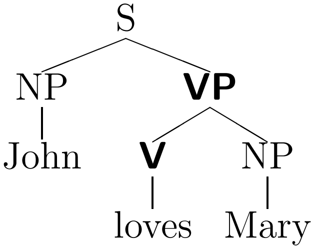
# Constituency vs. Dependency

Constituency trees (CFG) represent only bracketing:
= which adjacent constituents are glued to each other.
Dependency trees represent which words depend on which.
+ usually, some agreement/conditioning along the edge.

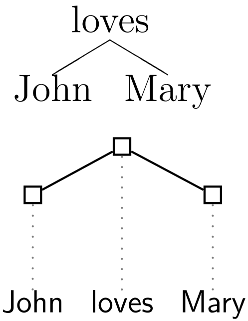
**Constituency**

John (loves Mary)

John ${}_{\text{VP}}$(loves Mary)



**Dependency**



# What Dependency Trees Tell Us

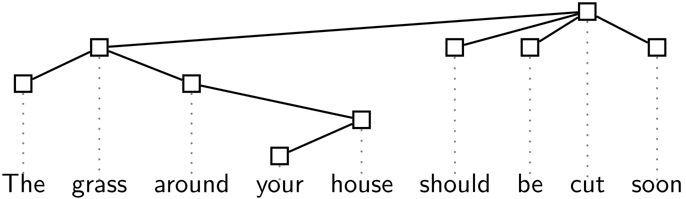
Input: The **grass** around your house should be **cut** soon.
Google: **Trávu** kolem vašeho domu by se měl **snížit** brzy.

- Bad lexical choice for *cut* = *sekat/snížit/krájet/řezat/. .*
  - Due to long-distance lexical dependency with *grass*.
  - One can "pump" many words in between.
  - Could be handled by full source-context (e.g. maxent) model.
- Bad case of *tráva*.
  - Depends on the chosen active/passive form:

| active⇒accusative | passive⇒nominative |
|---|---|
| trávu . . . by**ste** ~~se~~ měl posekat | tráva . . . by **se** měl**a** posekat |
| | tráva . . . by měl**a** **být** posek**ána** |

Examples by Zdeněk Žabokrtský, Karel Oliva and others.

# Tree vs. Linear Context

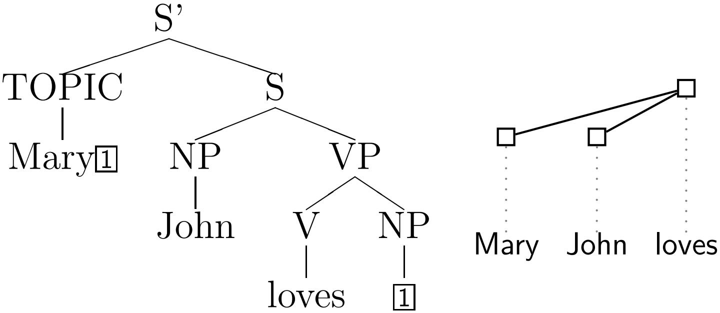


- Tree context (neighbours in the dependency tree):
  - is better at predicting lexical choice than *n*-grams.
  - often equals linear context:
    Czech manual trees: 50% of edges link neighbours,
    80% of edges fit in a 4-gram.
- Phrase-based MT is a very good approximation.
- Hierarchical MT can even capture the dependency in one phrase:

$X \rightarrow <$ the grass $X$ should be cut, trávu $X$ byste měl posekat $>$

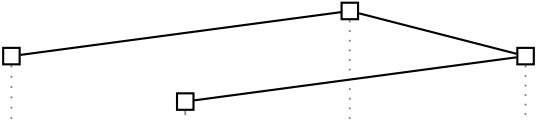
# “Crossing Brackets”

- Constituent outside its father’s span causes “crossing brackets.”
    - Linguists use “traces” ($\boxed{1}$) to represent this.
- Sometimes, this is not visible in the dependency tree:
    - There is no “history of bracketing”.
    - See Holan et al. (1998) for dependency trees including derivation history.



# Non-Projectivity

= a gap in a subtree span, filled by a node higher in the tree. Ex. Dutch “cross-serial” dependencies, a non-projective tree with one gap caused by *saw* within the span of *swim*.



| | | | | |
|---|---|---|---|---|
| . . . dat | Jan | kinderen | zag | zwemmen |
| . . . that | John | children | saw | swim |

. . . that John saw children swim.

- 0 gaps = projective tree ⇒ representable in CFG.
- ≤ 1 gap & “well-nested” ⇒ mildly context sensitive (TAG). See Kuhlmann and Möhl (2007) and Holan et al. (1998).

# Why Non-Projectivity Matters?

- CFGs cannot handle non-projective constructions:

  Imagine John **grass** saw **being cut**!

- No way to glue these crossing dependencies together:
  - Lexical choice:

    $$X \rightarrow < \text{grass } X \text{ being cut}, \text{trávu } X \text{ sekat} >$$

  - Agreement in gender:

    $$X \rightarrow < \text{John } X \text{ saw}, \text{Jan } X \text{ viděl} >$$
    $$X \rightarrow < \text{Mary } X \text{ saw}, \text{Marie } X \text{ viděl}\textbf{a} >$$

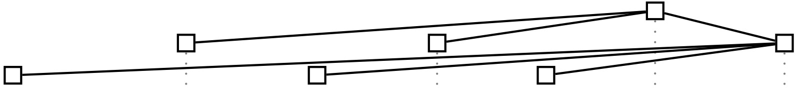
- Phrases can memorize <u>fixed</u> sequences containing:
  - the non-projective construction
  - and all the words in between! ($\Rightarrow$ extreme sparseness)

# Is Non-Projectivity Severe?

Depends on the language.

In principle unlimited:

- Czech allows long gaps as well as many gaps in a tree.



| | | | | | | |
|---|---|---|---|---|---|---|
| Proti odmítnutí | se | zítra | Petr | v práci | rozhodl | protestovat |
| Against dismissal | aux-refl | tomorrow | Peter | at work | decided | to object |

Peter decided to object against the dismissal at work tomorrow.

In treebank data:

⊖ 23% of Czech sentences contain a non-projectivity.

⊕ 99.5% of Czech sentences are well nested with $\leq 1$ gap.

# Parallel View

- Ignoring formal linguistic grammar, do we have to reorder beyond swapping constituents?
  - This is the ITG (Hiero with $\leq 2$ nonterminals) limitation.

| | | English-Czech Parallel Sents | |
|---|---|---|---|
| Domain | Alignment | Total | Beyond ITG |
| WSJ | manual Sure | 515 | 2.9% |
| WSJ | manual S+P | 515 | 15.9% |
| News | GIZA++, gdfa | 126k | 10.6% |
| Mixed | GIZA++, gdfa | 6.1M | 3.5% |

- searched for (discontinuous) 4-tuples of alignment points in forbidden shapes (3142 and 2413).
- additional alignment links were allowed to intervene (and could force different segmentation to phrases) $\Rightarrow$ we overestimate.
- no larger sequences of tokens were considered as a unit $\Rightarrow$ we underestimate.

# Don't Care Approach (cs→en)

Input: Zítra **se** v kostele Sv. Trojice budou **brát** Marie a Honza.
Ref: Mary and John get married in the Holy Trinity church tomorrow.

---

Goog: Tomorrow **is** the Holy Trinity church will **take** Mary and John.

- Bad lexical choice:
  *brát* = *take* vs. *brát se* = *get married*
- Superfluous *is*:
  - *se* is very often mis-aligned with the auxiliary *is*.

The straightforward bag-of-source-words model fails here:

- *se* is very frequent and it often means just *with*.
- An informed model would use the source parse tree.
  - Remember to use a non-projective parser!

# Tentative Conclusion on Reordering

For Indo-European languages, PBMT seems acceptable.

- Dependencies are most often local enough.
- Distant dependencies can be non-projective
  $\Rightarrow$ Hierarchical model does not help much either.

Other languages?

- We will try Tamil (Dravidian language, SOV) in the lab.
- ...but you'll see we will first hit another issue:
  rich morphology.

# Rich Morphology in PBMT Pipeline

- Word Alignment.
- Extraction of Translation Units.
- Translation of New Text.
- (Reordering.)
- Language Modelling.
- MT Evaluation.
- Model Optimization.

# Rich Morphology in PBMT Pipeline

- Word Alignment.
- Extraction of Translation Units.
- Translation of New Text.
    - New forms of known words.
    - Unknown words.
- (Reordering.)
- Language Modelling.
    - Sparser unigrams and higher-grams (reordering).
- MT Evaluation.
    - Fewer matches with the reference.
- Model Optimization.

. . . rich morphology makes everything harder.

# Rich Morphology in PBMT Pipeline

- Word Alignment. ⇒ Lab: Stem, chop (or lemmatize or LEAF).
- Extraction of Translation Units.
- Translation of New Text.
    - New forms of known words. ⇒ Here: Two-Step; Lab: Split+Join.
    - Unknown words. ⇒ Word derivations in Treex.
- (Reordering.)
- Language Modelling.
    - Sparser unigrams and higher-grams (reordering).
- MT Evaluation. ⇒ Here: Problems of BLEU.
    - Fewer matches with the reference.
- Model Optimization. ⇒ Here: SemPOS+BLEU.

. . . rich morphology makes everything harder.

# Morphological Explosion in Czech

**(In)flective lang.**: suffix encodes many categories:

- Czech nouns and adjs: 7 cases, 4 genders, 3 nums, . . .
- Czech verbs: gender, num, aspect (im/perfective), . . .

| I | saw | two | green | striped | cats | . |
|---|---|---|---|---|---|---|
| já | pila | dva | zelený | pruhovaný | **kočky** | . |
| | pily | **dvě** | zelená | pruhovaná | koček | |
| | . . . | dvou | **zelené** | **pruhované** | kočkám | |
| | viděl | dvěma | zelení | pruhovaní | kočkách | |
| | viděla | dvěmi | zeleného | pruhovaného | kočkami | |
| | . . . | | zelených | pruhovaných | | |
| | uviděl | | zelenému | pruhovanému | | |
| | uviděla | | zeleným | pruhovaným | | |
| | . . . | | zelenou | pruhovanou | | |
| **viděl jsem** | | | zelenými | pruhovanými | | |
| viděla jsem | | | . . . | . . . | | |

# Result: Out-of-Vocabulary Rates

| Dataset | | *n*-grams Out of: Corpus Voc. | | Phrase-Table Voc. | |
|---|---|---|---|---|---|
| (# Sents) | Language | 1 | 2 | 1 | 2 |
| 7.5M | Czech | 2.2% | 30.5% | 3.9% | 44.1% |
| | English | 1.5% | 13.7% | 2.1% | 22.4% |
| | Czech + English input sent | 1.5% | 29.4% | 3.1% | 42.8% |
| 126k | Czech | 6.7% | 48.1% | 12.5% | 65.4% |
| | English | 3.6% | 28.1% | 6.3% | 45.4% |
| | Czech + English input sent | 5.2% | 46.6% | 10.6% | 63.7% |
| 126k | Czech lemmas | 4.1% | 36.3% | 5.8% | 52.6% |
| | English lemmas | 3.4% | 24.6% | 6.9% | 53.2% |
| | Czech + English input lemmas | 3.1% | 35.7% | 5.1% | 38.1% |

# Result: Out-of-Vocabulary Rates

| Dataset (# Sents) | Language | *n*-grams Out of: Corpus Voc. 1 | 2 | Phrase-Table Voc. 1 | 2 |
|---|---|---|---|---|---|
| 7.5M | Czech | 2.2% | 30.5% | 3.9% | 44.1% |
| | English | 1.5% | 13.7% | 2.1% | 22.4% |
| | Czech + English input sent | 1.5% | 29.4% | 3.1% | 42.8% |
| 126k | Czech | 6.7% | 48.1% | 12.5% | 65.4% |
| | English | 3.6% | 28.1% | 6.3% | 45.4% |
| | Czech + English input sent | 5.2% | 46.6% | 10.6% | 63.7% |
| 126k | Czech lemmas | 4.1% | 36.3% | 5.8% | 52.6% |
| | English lemmas | 3.4% | 24.6% | 6.9% | 53.2% |
| | Czech + English input lemmas | 3.1% | 35.7% | 5.1% | 38.1% |

- Significant vocabulary loss during phrase extraction:
  - e.g. 2.2%→3.9% for 7.5M Czech.

# Result: Out-of-Vocabulary Rates

| Dataset<br>(# Sents) | Language | *n*-grams Out of: Corpus Voc. 1 | 2 | Phrase-Table Voc. 1 | 2 |
|---|---|---|---|---|---|
| 7.5M | Czech | 2.2% | 30.5% | 3.9% | 44.1% |
| | English | 1.5% | 13.7% | 2.1% | 22.4% |
| | Czech + English input sent | 1.5% | 29.4% | 3.1% | 42.8% |
| 126k | Czech | 6.7% | 48.1% | 12.5% | 65.4% |
| | English | 3.6% | 28.1% | 6.3% | 45.4% |
| | Czech + English input sent | 5.2% | 46.6% | 10.6% | 63.7% |
| 126k | Czech lemmas | 4.1% | 36.3% | 5.8% | 52.6% |
| | English lemmas | 3.4% | 24.6% | 6.9% | 53.2% |
| | Czech + English input lemmas | 3.1% | 35.7% | 5.1% | 38.1% |

- Significant vocabulary loss during phrase extraction:
  - e.g. 2.2%→3.9% for 7.5M Czech.
- OOV of Czech forms ∼twice as bad as in English.

# Result: Out-of-Vocabulary Rates

| Dataset (# Sents) | Language | *n*-grams Out of: Corpus Voc. 1 | 2 | Phrase-Table Voc. 1 | 2 |
|---|---|---|---|---|---|
| 7.5M | Czech | 2.2% | 30.5% | 3.9% | 44.1% |
| | English | 1.5% | 13.7% | 2.1% | 22.4% |
| | Czech + English input sent | 1.5% | 29.4% | 3.1% | 42.8% |
| 126k | Czech | 6.7% | 48.1% | 12.5% | 65.4% |
| | English | 3.6% | 28.1% | 6.3% | 45.4% |
| | Czech + English input sent | 5.2% | 46.6% | 10.6% | 63.7% |
| 126k | Czech lemmas | 4.1% | 36.3% | 5.8% | 52.6% |
| | English lemmas | 3.4% | 24.6% | 6.9% | 53.2% |
| | Czech + English input lemmas | 3.1% | 35.7% | 5.1% | 38.1% |

- Significant vocabulary loss during phrase extraction:
  - e.g. 2.2%→3.9% for 7.5M Czech.
- OOV of Czech forms ∼twice as bad as in English.
- OOV of Czech lemmas lower than in English.

# Result: Out-of-Vocabulary Rates

| Dataset | | *n*-grams Out of: Corpus Voc. | | Phrase-Table Voc. | |
|---|---|---|---|---|---|
| (# Sents) | Language | 1 | 2 | 1 | 2 |
| | Czech | 2.2% | 30.5% | 3.9% | 44.1% |
| 7.5M | English | 1.5% | 13.7% | 2.1% | 22.4% |
| | Czech + English input sent | 1.5% | 29.4% | 3.1% | 42.8% |
| | Czech | 6.7% | 48.1% | 12.5% | 65.4% |
| 126k | English | 3.6% | 28.1% | 6.3% | 45.4% |
| | Czech + English input sent | 5.2% | 46.6% | 10.6% | 63.7% |
| | Czech lemmas | 4.1% | 36.3% | 5.8% | 52.6% |
| 126k | English lemmas | 3.4% | 24.6% | 6.9% | 53.2% |
| | Czech + English input lemmas | 3.1% | 35.7% | 5.1% | 38.1% |

- Significant vocabulary loss during phrase extraction:
    - e.g. 2.2%→3.9% for 7.5M Czech.
- OOV of Czech forms ∼twice as bad as in English.
- OOV of Czech lemmas lower than in English.
- Free word order of Czech apparent.

# Two-Step Moses 1/2

- English → lemmatized Czech
  - meaning-bearing morphology preserved
  - max phrase len 10, distortion limit 6
  - large target-side (lemmatized LM)
- Lemmatized Czech → Czech
  - max phrase len 1, monotone

| | | | | |
|---|---|---|---|---|
| 0 | **Src** | after a sharp drop | | |
| 1 | **Mid** | po+6 | ASA1.prudký | NSA-.pokles |
| | **Gloss** | after+voc | adj+sg...sharp | noun+sg...drop |
| 2 | **Out** | po | prudkém | poklesu |

- Only 1-best output passed, lattices on our todo list.
- See also works by Alex Fraser for targetting German.
- Alternative: Exponential models (Subotin, 2011).

# Two-Step Moses 2/2

| Data Size | | Simple | | Two-Step | | Diff |
|---|---|---|---|---|---|---|
| Parallel | Mono | BLEU | SemPOS | BLEU | SemPOS | B. S. |
| 126k | 126k | 10.28±0.40 | 29.92 | 10.38±0.38 | 30.01 | ↗↗ |
| 126k | 13M | 12.50±0.44 | 31.01 | 12.29±0.47 | 31.40 | ↘↗ |
| 7.5M | 13M | 14.17±0.51 | 33.07 | 14.06±0.49 | 32.57 | ↘↘ |

Manual micro-evaluation of ↘↗, i.e. 12.50±0.44 vs. 12.29±0.47:

| | Two-Step | Both Fine | Both Wrong | Simple | Total |
|---|---|---|---|---|---|
| Two-Step | **23** | 4 | 8 | - | **35** |
| Both Fine | 7 | 14 | 17 | 5 | 43 |
| Both Wrong | 8 | 1 | 28 | 2 | 39 |
| Simple | - | 3 | 7 | **23** | 33 |
| Total | **38** | 22 | 60 | 30 | 150 |

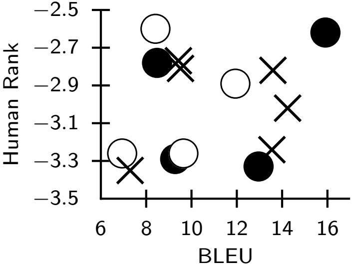
- Each annotator weakly prefers Two-step
  - but they don't agree on individual sentences.

# Effect of Rich Morphology on BLEU

- Large vocabulary impedes the performance of BLEU.

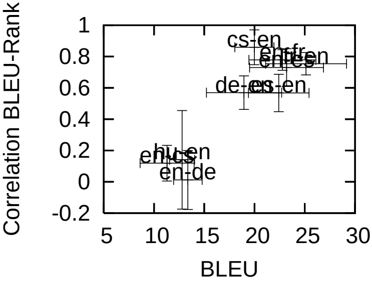
En→Cs Systems

WMT08, WMT09



Various Language Pairs

WMT08, WMT09, MetricsMATR



⇒ BLEU does not correlate with human rank if below ∼20.

# Reason 1: Focus on Forms

| | |
|---|---|
| SRC | Prague Stock Market falls to minus by the end of the trading day |
| REF | pražská burza se ke konci obchodování propadla do minusu |
| cu-bojar | praha stock market klesne k minus na konci obchodního dne |
| pctrans | praha trh cenných papírů padá minus do konce obchodního dne |

- Only a single unigram in each hyp. confirmed by the reference.
- Large chunks of hypotheses are not compared at all.

| Confirmed by Reference | Yes | Yes | No | No |
|---|---|---|---|---|
| Contains Errors | Yes | No | Yes | No |
| Running words | 6.34% | 36.93% | 22.33% | **34.40%** |

# Reason 2: Sequences Overvalued

BLEU overly sensitive to sequences:

- Gives credit for 1, 3, 5 and 8 four-, three-, bi- and unigrams,
- Two of three serious errors not noticed,
  ⇒ Quality of cu-bojar overestimated.

| | |
|---|---|
| SRC | Congress yields: US government can pump 700 billion dollars into banks |
| REF | kongres ustoupil : vláda usa může do bank napumpovat 700 miliard dolarů |
| cu-bojar | kongres výnosy : vláda usa může čerpadlo 700 miliard dolarů v bankách |
| pctrans | kongres vynáší : us vláda může čerpat 700 miliardu dolarů do bank |

⇒ Bojar et al. (2010) use SemPOS, a coarse metric that correlates better with humans for Czech and English.

# Optimizing Towards SemPOS

SemPOS compares bags of lemmas, not sequences of forms.

- Sequences not overvalued
  ⇒ better correlation with human ranking.
- Not fit for selecting best output from n-best list.
  ⇒ Need to combine with e.g. BLEU.

WMT11 Tunable Metrics Task, manual ranking:

| System | ≥others | >others |
|---|---|---|
| bleu● | **0.79** | 0.28 |
| bleu-single● | 0.77 | 0.27 |
| cmu-meteor● | 0.76 | 0.27 |
| rwth-cder | 0.76 | 0.26 |
| cu-sempos-bleu● | 0.74 | **0.29** |
| stanford-dcp● | 0.73 | 0.27 |
| nus-tesla-f | 0.68 | 0.28 |
| sheffield-rose | 0.05 | 0.00 |

- Among the many "winners" (●).
- Best in ">others", i.e. when ties are not rewarded.

# Optimizing Towards SemPOS

SemPOS compares bags of lemmas, not sequences of forms.

- Sequences not overvalued
  ⇒ better correlation with human ranking.
- Not fit for selecting best output from n-best list.
  ⇒ Need to combine with e.g. BLEU.

WMT11 Tunable Metrics Task, manual ranking:

| System | ≥others | >others |
|---|---|---|
| bleu● | **0.79** | 0.28 |
| bleu-single● | 0.77 | 0.27 |
| cmu-meteor● | 0.76 | 0.27 |
| rwth-cder | 0.76 | 0.26 |
| cu-sempos-bleu● | 0.74 | **0.29** |
| stanford-dcp● | 0.73 | 0.27 |
| nus-tesla-f | 0.68 | 0.28 |
| sheffield-rose | 0.05 | 0.00 |

- Among the many "winners" (●).
- Best in ">others", i.e. when ties are not rewarded.
- Generally hard to interpret the ranking.

# Motivation for Experiment Mgmt (1/2)

Research needs reproducibility.

- Console-based environment alone helps a lot:
  - Bash history of past commands.
  - Log files.
- Complications:
  - Experiments carried out in parallel.
    Experiments can take days.
    ⇒ Easy to lose track.
  - Should reuse large intermediate files.
  - Different versions of the research software.
    (Both daily updates as well as yearly updates.)

# Motivation for Experiment Mgmt (2/2)

Research is search.
(for the best procedure, the best configuration, . . . )

You can think of research in AI/machine-learning terms.

- Heuristics:
  - Run quick probes (small data) first, then replicate on full.
- Beam Search: Increase *your* beam size:
  - Run ˜10 variations of each experiment.
- Genetic Algorithms:
  - Clone and modify most successful experiments.
- ("The best" varies based on the metric chosen.)
  - So look at more metrics at once.

# Features of Eman

- Console-based ⇒ easily scriptable (e.g. in bash).
- Versatile: “seeds” are up to the user, any language.

- Support for the manual search through the space of experiment configurations.
- Support for finding and marking (“tagging”) experiments of interest.
- Support for organizing the results in 2D tables.

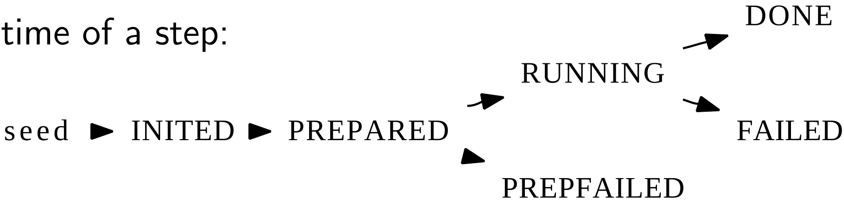
- Integrated with SGE
  ⇒ easy to run on common academic clusters.

**eman --man** will tell you some details.

# Eman's View

- Experiments consist of processing STEPS.
- Steps are:
  - of a given type, e.g. **align**, **tm**, **lm**, **mert**,
  - defined by immutable variables, e.g. **ALISYM=gdfa**,
  - all located in one directory, the "**playground**",
  - timestamped unique directories, e.g. **s.mert.a123.20120215-1632**
  - self-contained in the dir as much as reasonable.
  - dependent on other steps, e.g. first **align**, then build **tm**, then **mert**.

Lifetime of a step:



# Eman is Versatile

What types of steps should I have?

- Any, depending on your application.

What language do I write steps in?

- Any, e.g. bash.

What are the input and output files of the steps?

- Any, just make depending steps understand each other.
- Steps can have many output files and serve as prerequisites to different types of other steps.

What are measured values of my experiments?

- Anything from any of the files any step produces.

# What the User Implements: Just Seeds

Technically, a seed is any program that:

- responds to arbitrary environment variables,
- runs **eman defvar** to register step variables with eman,
- produces another program, **./eman.command** that does the real job.

The seed is actually run twice:

- At “init”: to check validity of input variables and register them with eman.
- At “prepare”: to produce **eman.command**.

The user puts all seeds in **playground/eman.seeds**.

- Eman runs a local copy of the seed in a fresh step dir.

# Why INITED→PREPARED→RUNNING?

The call to **eman init** *seed*:

- Should be quick, it is used interactively.
- Should only check and set vars, “turn a blank directory to valid eman step”.

The call to **eman prepare** *s.step.123.20120215*:

- May check for various input files.
  - Less useful with heavy experiments where even corpus preparation needs cluster.
- Has to produce **eman.command**.
  ⇒ A chance to check it: are all file paths correct etc.?

The call to **eman start** *s.step.123.20120215*:

- Sends the job to the cluster.

# Bells and Whistles

Experiment management:

- **ls**, **vars**, **stat** for simple listing,
- **select** for finding steps,
- **traceback** for full info on experiments,
- **redo** failed experiments,
- **clone** individual steps as well as whole experiments.

Meta-information on steps:

- **status**,
- **tag**s, autotags,
- **collect**ing results,
- **tabulate** for putting results into 2D tables.

# eman select

- Step dirs don't have nice names.
- You need to locate steps of given properties.

What all language models do I have?

- **eman ls lm**
- **eman select t lm**

If we need just the finished ones:

- **eman stat lm | grep DONE**
- **eman select t lm d**

And just 5-gram ones for English:

- **eman select t lm d vre ORDER=5 vre CORPAUG=en**

# eman traceback

## eman traceback s.evaluator.8102edfc.20120207-1611

```
+- s.evaluator.8102edfc.20120207-1611
|  +- s.mosesgiza.b6073a00.20120202-0037
|  +- s.translate.b17f203d.20120207-1604
|  |  +- s.mert.272f2f67.20120207-0013
|  |  |  +- s.model.3e28def7.20120207-0013
|  |  |  |  +- s.lm.608df574.20120207-0004
|  |  |  |  |  +- s.srilm.117f0cfe.20120202-0037
|  |  |  |  +- s.mosesgiza.b6073a00.20120202-0037
|  |  |  |  +- s.tm.527c9342.20120207-0012
|  |  |  |  |  +- s.align.dec45f74.20120206-0111
|  |  |  |  |  |  +- s.mosesgiza.b6073a00.20120202-0037
|  |  |  |  |  +- s.mosesgiza.b6073a00.20120202-0037
|  |  +- s.mosesgiza.b6073a00.20120202-0037
```

Options: **--vars --stat --log ... --ignore=***steptype*

# eman redo

On cluster, jobs can fail nondeterminically.

- Bad luck when scheduled to a swamped machine.
- Bad estimate of hard resource limits (RAM exceeds the limit ⇒ job killed).

Eman to the rescue:

- **eman redo** *step* creates a new instance of each failed step, preserving the experiment structure.
- **eman redo** *step* **--start** starts the steps right away.

To make sure eman will do what you expect, first try:

- **eman redo** *step* **--dry-run**

# eman clone

CLONING is initing a new step using vars of an existing one.
Cloning of individual steps is useful:

- when a step failed (used in **eman redo**),
- when the seed has changed,
- when we want to redefine some vars:
  **ORDER=4 eman clone s.lm.1d6f791c...**

Cloning of whole tracebacks:

- The text of a traceback gets instantiated as steps.
- Existing steps are reused if OK and with identical vars.
- **eman traceback** *step* | **eman clone**
- **eman traceback** *step* | **mail bojar@ufal**
  followed by **eman clone** < **the-received-mail**.

# Deriving Experiments using **clone**

The text form of traceback allows to tweak the experiment:

- **eman tb** *step* | **sed ’s/cs/de/’** | **eman clone**
  replicates our experiment on German instead of Czech.

The derivation is now available in eman itself:

- **eman tb** *step* **-s ’/cs/de/’ -s ’/form/lc/’**
  shows the traceback with the substitutions highlighted.
  - A good chance to check if the derivation does the intended.
- **eman tb** *step* **-s ’/cs/de/’ -s ’/form/lc/’** \\
  | **eman clone --dry-run**
  - Last chance to check if existing steps get reused and what vars will new steps be based on.
  - Drop **--dry-run** to actually init the new steps.

# **eman tag** or **eman ls --tag** shows tags

TAGS and AUTOTAGS are:

- arbitrary keywords assigned to individual steps,
- inherited from dependencies.

Tags are:

- added using **eman add-tag** *the-tag steps*,
- stored in `s.stepdir.123/`**eman.tag**.

⇒ Use them to manually mark exceptions.

Autotags are:

- specified in `playground/`**eman.autotags** as regexes over step vars, e.g.: **/ORDER=(.*)/$1gr/** for LM,
- (re-)observed at **eman retag**.

⇒ Use them to systematically mark experiment branches.

# eman collect

Based on rules in **eman.results.conf**, e.g.:

```
BLEU  */BLEU.opt                      BLEU\s*=\s*([^\s,]+)
Snts  s.eval*/corpus.translation      CMD: wc -l
```

eman collects results from all steps into **eman.results**:

```
# Step Name                        Status   Score Value Tags and Autotags
s.evaluator.11ccf590.20120208-1554 DONE      TER   31.04 5gr DEVwmt10 LMc-news towards-
s.evaluator.11ccf590.20120208-1554 DONE      PER   44.61 5gr DEVwmt10 LMc-news towards-
s.evaluator.11ccf590.20120208-1554 DONE      CDER  33.97 5gr DEVwmt10 LMc-news towards-
s.evaluator.11ccf590.20120208-1554 DONE      BLEU  12.28 5gr DEVwmt10 LMc-news towards-
s.evaluator.11ccf590.20120208-1554 DONE      Snts   3003 5gr DEVwmt10 LMc-news towards-
s.evaluator.29fa5679.20120207-1357 OUTDATED TER   17.66 5gr DEVwmt10 LMc-news
...                                ...       ...     ...
s.evaluator.473687bb.20120214-1509 FAILED    Snts  3003
```

- Perhaps hard to read.
- Easy to grep, sort, whatever, or **tabulate**.

# **eman tabulate** to Organize Results

The user specifies in the file **eman.tabulate**:

- which results to ignore, which to select,
- which tags contribute to col labels, e.g. **TER**, **BLEU**,
- which tags contribute to row labels, e.g. **[0-9]gr**, **towards-[A-Z]+**, **PRO**.

Eman tabulates the results, output in **eman.niceresults**:

```
                  PER   CDER  TER   BLEU
5gr towards-CDER 44.61 33.97 31.04 12.28
5gr              44.19 33.76 31.02 12.18
5gr PRO          43.91 33.87 31.49 12.09
5gr towards-PER  44.44 33.52 30.74 11.95
```

# Hacking Welcome

Eman is designed to be hacking-friendly:

- Selfcontained steps are easy to inspect:
  - all logs are there,
  - all (or most of) input files are there,
  - the main code (**eman.command**) is there,
  - often, even the binaries are there, or at least clearly identifiable.
- Step halfway failed?
  ⇒ Hack its **eman.command** and use **eman continue**.
- Seed not quite fit for your current needs?
  ⇒ Just init the step and hack **eman.seed**.
  ⇒ Or also prepare and hack **eman.command**.

Remember to **eman add-tag** *tag step* for further reference.

# Fit for Cell-Phone SSH ☺

- Experiments run long but fail often.
- You don't want to be chained to a computer.

Most eman commands have a short nickname.

- How are my last 10 merts?
  ```
  eman sel t mert l 10 --stat
  ```

Specify steps using any part of their name/hash or result:

- `s.foobar.`**a0f3**`b123.20120215-1011` failed, retry it:
  ```
  eman redo a0f3 --start
  ```
- How did I achieve this great BLEU score of 25.10?
  ```
  eman tb 25.10 --vars | less
  ```

# Related Experiment Mgmt Systems

Eman is just one of many, consider also:

- LoonyBin (Clark et al., 2010)
  - ⊖ Clickable Java tool.
  - ⊕ Support for multiple clusters and scheduler types.
- Moses EMS (Koehn, 2010)
  - Experiment Management System primarily for Moses.
  - Centered around a single experiment which consists of steps.
- Pure Makefiles

  Yes, you can easily live with fancy Makefiles.
  - You will use commands like **make init.mert** or **cp -r exp.mert.1 exp.mert.1b**
  - You need to learn to use **$***, **$@** etc.
  - You are likely to implement your own eman soon. ☺

There are also the following workflow management systems: DAGMan, Pegasus, Dryad.

# Work in Progress

- Eman is being heavily used by a rather few people.
- Eman is still evolving
    - ⇒ not everything well documented (read the source code).
    - ⇒ not everything well tested.

Halfway finished: eman teamwork!

- **eman add remote** */home/fred/playground freds-exps*
- You can re-interpret Fred's results.
- You can clone Fred's experiments.
- You can make your steps depend on Fred's steps.

# Summary

- Word order issues of PBMT, RBMT and hierarchical MT.
- Rich morphology issues in PBMT:
    - Producing target forms never seen in parallel data.
    - Evaluating MT to morphologically rich languages.
    - Model optimization.

- General motivation for experiment management.
- Introduced eman.
- Highlighted useful tricks in experimenting.
    - Experiment cloning or deriving.
    - Tabulating results.
    - ? Team experimenting.

# References

Ondřej Bojar, Kamil Kos, and David Mareček. 2010. Tackling Sparse Data Issue in Machine Translation Evaluation. In Proceedings of the ACL 2010 Conference Short Papers, pages 86–91, Uppsala, Sweden, July. Association for Computational Linguistics.

Jonathan H. Clark, Jonathan Weese, Byung Gyu Ahn, Andreas Zollmann, Qin Gao, Kenneth Heafield, and Alon Lavie. 2010. The Machine Translation Toolpack for LoonyBin: Automated Management of Experimental Machine Translation HyperWorkflows. Prague Bulletin of Mathematical Linguistics, 93:117–126.

Tomáš Holan, Vladislav Kuboň, Karel Oliva, and Martin Plátek. 1998. Two Useful Measures of Word Order Complexity. In A. Polguere and S. Kahane, editors, Proceedings of the Coling '98 Workshop: Processing of Dependency-Based Grammars, Montreal. University of Montreal.

Philipp Koehn. 2010. An Experimental Management System. Prague Bulletin of Mathematical Linguistics, 94:87–96, September.

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

Michael Subotin. 2011. An exponential translation model for target language morphology. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 230–238, Portland, Oregon, USA, June. Association for Computational Linguistics.