Machine Translation Zoo

Tree-to-tree transfer and Discriminative learning

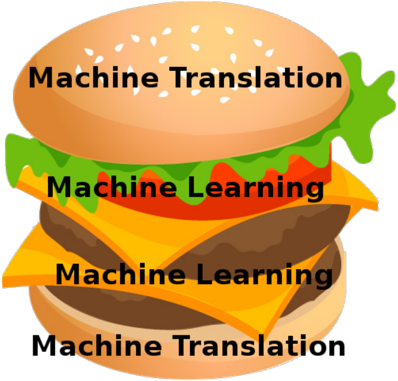
Martin Popel

ÚFAL (Institute of Formal and Applied Linguistics)
Charles University in Prague

May 5th 2013, Seminar of Formal Linguistics, Prague

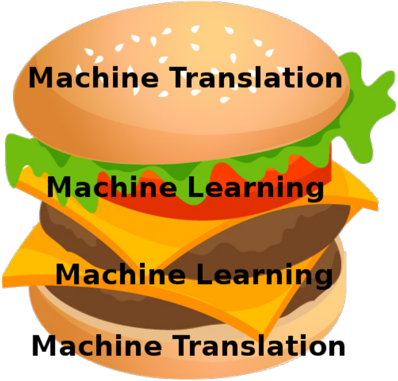
Today's Menu

1. MT Intro
   - Taxonomy
   - Hybrids
2. Online Learning
   - Perceptron
   - Structured Prediction
3. Guided Learning
4. Back to MT
   - Easy-First Decoding in MT
   - Guided Learning in MT

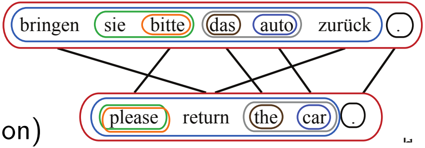


Today's Menu

1. MT Intro
   - Taxonomy
   - Hybrids
2. Online Learning
   - Perceptron
   - Structured Prediction
3. Guided Learning
4. Back to MT
   - Easy-First Decoding in MT
   - Guided Learning in MT



Phrase-based MT (Moses)



Training

- word-alignment
  (GIZA++ & symmetrization)
- phrase extraction
- tune parameters (MERT)

Decoding

- get all matching *rules*
- find one *derivation*
  with a maximum score (beam search)

TectoMT

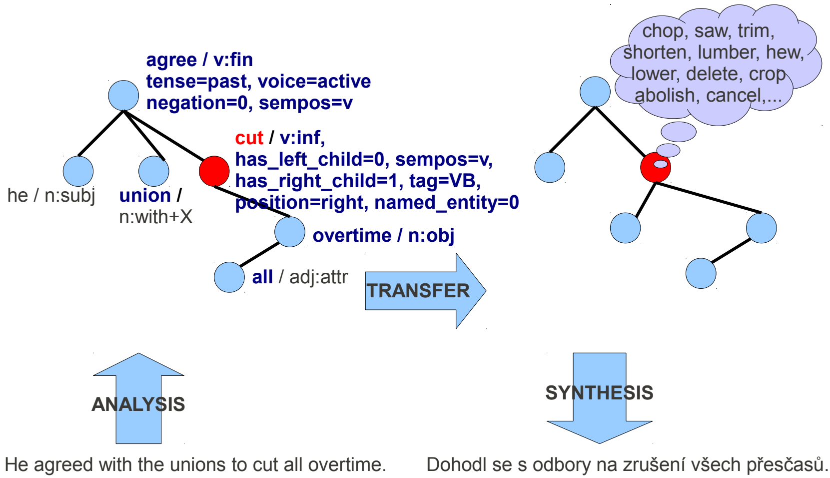
Training

- analyze CzEng to t-layer
- t-node alignment
- learn one MaxEnt model for each source lemma and formeme

Decoding

- get all translation variants for each lemma and formeme
- find a labeling with a maximum score (HMTM)

TectoMT – MaxEnt Model



Machine Translation Taxonomy

- Level of transfer:
- Base translation unit (BTU):

- Extract more segmentations in training?
- Try (search) more segmentations in decoding?
- Use more segmentations in the output translation?
- What is the context X in $P(BTU_{target} | BTU_{source}, X)$?

Machine Translation Taxonomy

- Level of transfer: surface
- Base translation unit (BTU):
  word
- Extract more segmentations in training? no
- Try (search) more segmentations in decoding? no
- Use more segmentations in the output translation? no
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: nothing

(Brown et al., 1993)
word-based

Machine Translation Taxonomy

- Level of transfer: surface
- Base translation unit (BTU):
  word, phrase
- Extract more segmentations in training? yes
- Try (search) more segmentations in decoding? yes
- Use more segmentations in the output translation? no
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: nothing

(Brown et al., 1993)
word-based

(Koehn et al., 2003)
phrase-based

Machine Translation Taxonomy

- Level of transfer: surface
- Base translation unit (BTU):
  word, phrase, phrase with gaps
- Extract more segmentations in training? yes
- Try (search) more segmentations in decoding? yes
- Use more segmentations in the output translation? no
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: nothing

IBM

(Brown et al., 1993)
word-based

(Koehn et al., 2003)
phrase-based

(Chiang, 2005)
hierarchical

Machine Translation Taxonomy

- Level of transfer: surface, shallow syntax
- Base translation unit (BTU):
  word, phrase, phrase with gaps, treelet
- Extract more segmentations in training? no
- Try (search) more segmentations in decoding? no
- Use more segmentations in the output translation? no
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: neighboring treelets

Microsoft Research
(Quirk and Menezes, 2006)
dep. treelet to string

IBM
(Brown et al., 1993)
word-based

(Koehn et al., 2003)
phrase-based

(Chiang, 2005)
hierarchical

Machine Translation Taxonomy

- Level of transfer: surface, shallow syntax, tectogrammatical
- Base translation unit (BTU):
  word, phrase, phrase with gaps, treelet, node
- Extract more segmentations in training? no
- Try (search) more segmentations in decoding? no
- Use more segmentations in the output translation? no
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: neighboring nodes

Microsoft Research

(Quirk and Menezes, 2006)
dep. treelet to string

(Mareček et al., 2010)
TectoMT

(Koehn et al., 2003)
phrase-based

(Chiang, 2005)
hierarchical

IBM

(Brown et al., 1993)
word-based

Machine Translation Taxonomy

- Level of transfer: surface, shallow syntax, tectogrammatical
- Base translation unit (BTU):
  word, phrase, phrase with gaps, treelet, node
- Extract more segmentations in training? yes
- Try (search) more segmentations in decoding? yes
- Use more segmentations in the output translation? yes
- What is the context X in $P(BTU_{target}|BTU_{source}, X)$?
  Considering just Translation Model: nothing

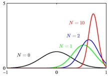
Microsoft Research

(Quirk and Menezes, 2006)
dep. treelet to string

(Mareček et al., 2010)
TectoMT

(Koehn et al., 2003)
phrase-based

(Chiang, 2005)
hierarchical



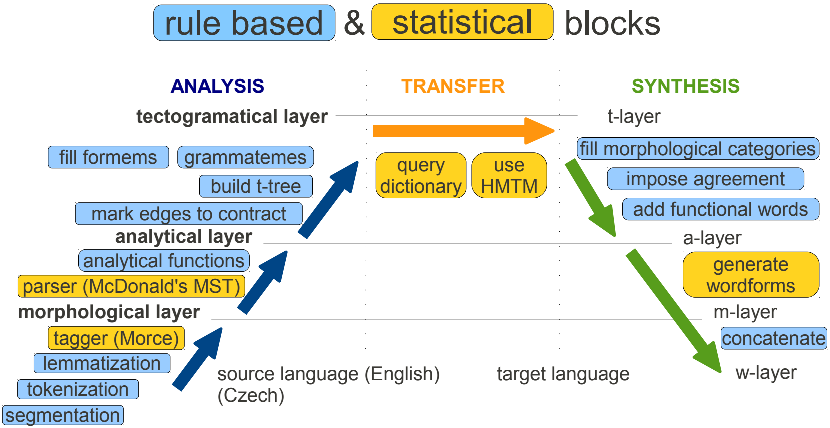
IBM

(Brown et al., 1993)
word-based

(Arun, 2011)
Monte Carlo

Hybrids: TectoMoses

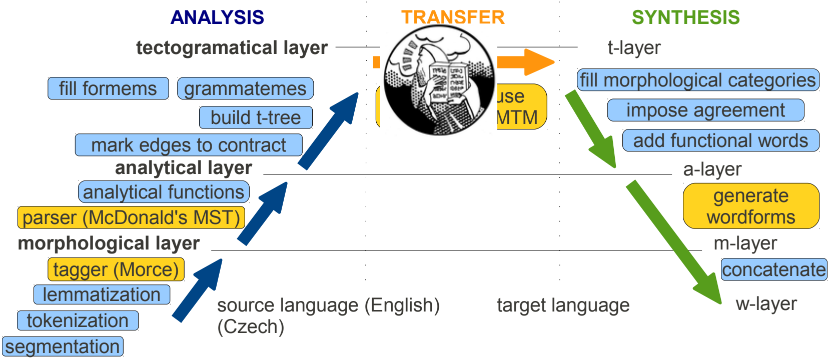
Linearize source t-trees (two factors: lemma and formeme), translate with Moses, project dependencies and use TectoMT synthesis.



Hybrids: TectoMoses

Linearize source t-trees (two factors: lemma and formeme), translate with Moses, project dependencies and use TectoMT synthesis.

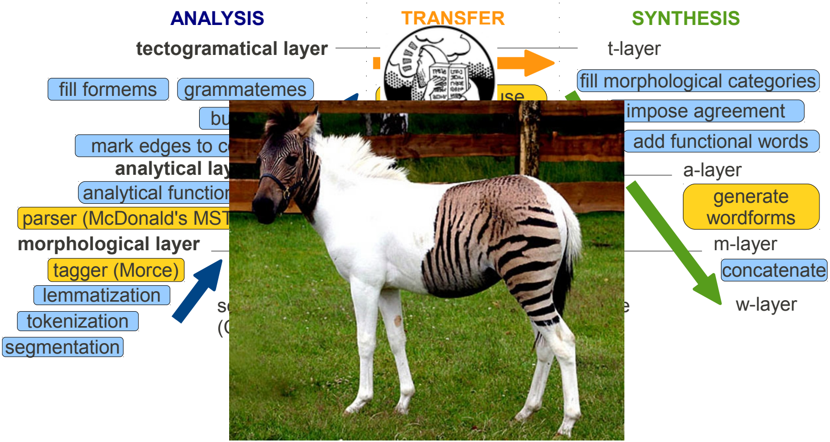
rule based & statistical blocks



Hybrids: TectoMoses

Linearize source t-trees (two factors: lemma and formeme), translate with Moses, project dependencies and use TectoMT synthesis.

rule based & statistical blocks



Hybrids: PhraseFix

Done for WMT 2013 by Petra Galuščáková:

- Post-edit TectoMT output using Moses
- trained on cs-tectomt → cs-reference (whole CzEng).
- How to post-edit only when confident?
  - filter phrase table
  - add "confidence" feature for MERT
  - improve alignment (monolingual)
  - boost phrase table (e.g. with identities)

Future work:

- use also source (English) sentences ⇒ multi-source translation
- project only content words (using TectoMT)
- factored translation with non-synchronous (overlapping) factors

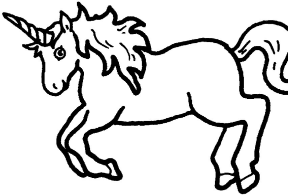
Hybrids: PhraseFix

Done for WMT 2013 by Petra Galuščáková

- Post-edit TectoMT output using Moses
- trained on cs-tectomt → cs-reference (
- How to post-edit only when confident?
  - filter phrase table
  - add "confidence" feature for MERT
  - improve alignment (monolingual)
  - boost phrase table (e.g. with identities)

Future work:

- use also source (English) sentences ⇒ multi-source translation
- project only content words (using TectoMT)
- factored translation with non-synchronous (overlapping) factors



Even More Hybrids: DepFix, AddToTrain, Chimera

DepFix (Rosa et al., 2012)

- post-edit SMT using syntactic analysis and rules
- exploit also the source sentences, robust parsing

AddToTrain (Bojar, Galuščáková)

- translate monolingual news (or WMT devsets) with TectoMT
- add this to Moses parallel training data

Chimera

- post-edit AddToTrain output with DepFix
- sent to WMT 2013 in attempt to beat Google

Even More Hybrids: DepFix, AddToTrain, Chimera

DepFix (Rosa et al., 2012)

- post-edit SMT using syntactic analysis and r
- exploit also the source sentences, robust pars

AddToTrain (Bojar, Galuščáková)

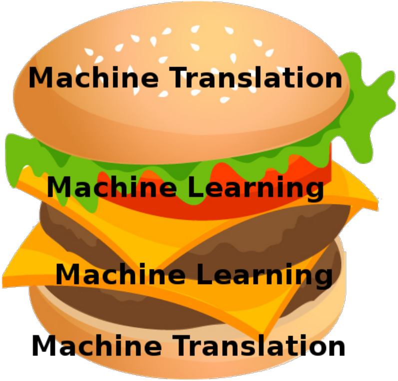
- translate monolingual news (or WMT devsets
- add this to Moses parallel training data

Chimera

- post-edit AddToTrain output with DepFix
- sent to WMT 2013 in attempt to beat Googl

Today's Menu

1. MT Intro
   - Taxonomy
   - Hybrids
2. Online Learning
   - Perceptron
   - Structured Prediction
3. Guided Learning
4. Back to MT
   - Easy-First Decoding in MT
   - Guided Learning in MT



General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
    $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w} \mathrel{+}=$ update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)
1. get its features $\mathbf{x}$

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

1. get its features $\mathbf{x}$
2. do the prediction y_{pred}

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w} \mathrel{+}=$ update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

1. get its features $\mathbf{x}$
2. do the prediction y_{pred}
3. get the correct label y_{gold}

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
    $\mathbf{w} \mathrel{+}=$ update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

1. get its features $\mathbf{x}$
2. do the prediction y_{pred}
3. get the correct label y_{gold}
4. update the weights

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
    $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

1. get its features $\mathbf{x}$
2. do the prediction y_{pred}
3. get the correct label y_{gold}
4. update the weights

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
    $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)
1. get its features $\mathbf{x}$
2. do the prediction y_{pred}
3. get the correct label y_{gold}
4. update the weights

Definition: conservative online learning

no error $\Rightarrow$ no update
i.e., if $y_{pred} = y_{gold}$ then update$(\mathbf{x}, y_{gold}, y_{pred}) = 0$

General Algorithm for Online Learning

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
    $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$
    //prediction$(\mathbf{w}, \mathbf{x}) = y_{gold}$
Output: $\mathbf{w}$

initialize all weights to zero
for each instance (observation)

1. get its features $\mathbf{x}$
2. do the prediction y_{pred}
3. get the correct label y_{gold}
4. update the weights

Definition: conservative online learning

no error $\Rightarrow$ no update
i.e., if $y_{pred} = y_{gold}$ then update$(\mathbf{x}, y_{gold}, y_{pred}) = 0$

Definition: aggressive online learning

after the update, the instance would be classified correctly

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} := \text{prediction}(\mathbf{w}, \mathbf{x})$
  $\mathbf{w} \mathrel{+}= \text{update}(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

Binary Perceptron

$$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=} [\mathbf{w} \cdot \mathbf{x} > 0]$$

$$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=} \alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} := \text{prediction}(\mathbf{w}, \mathbf{x})$
  $\mathbf{w} \mathrel{+}= \text{update}(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

dot product (similarity score) of weights and features
$\mathbf{w} \cdot \mathbf{x} = \sum_i w_i x_i$

Binary Perceptron

$$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=} [\mathbf{w} \cdot \mathbf{x} > 0]$$

$$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=} \alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

dot product (similarity score) of weights and features
$\mathbf{w} \cdot \mathbf{x} = \sum_i w_i x_i$

Iverson bracket

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true;} \\ 0 & \text{otherwise.} \end{cases}$$

Binary Perceptron

$$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=} [\mathbf{w} \cdot \mathbf{x} > 0]$$

$$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=} \alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} := \text{prediction}(\mathbf{w}, \mathbf{x})$
    $\mathbf{w} \mathrel{+}= \text{update}(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

dot product (similarity score) of weights and features
$\mathbf{w} \cdot \mathbf{x} = \sum_i w_i x_i$

Iverson bracket

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true;} \\ 0 & \text{otherwise.} \end{cases}$$

Binary Perceptron

$$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=} [\mathbf{w} \cdot \mathbf{x} > 0]$$

$$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=} \alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$$

learning rate (step size) $\alpha > 0$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} := \text{prediction}(\mathbf{w}, \mathbf{x})$
  $\mathbf{w} \mathrel{+}= \text{update}(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

	Binary Perceptron	Multi-class Perceptron
$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=}$	$[\mathbf{w} \cdot \mathbf{x} > 0]$	$\arg\max_y \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=}$	$\alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$	$\alpha\left(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\right)$

learning rate (step size) $\alpha > 0$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
  $y_{pred} :=$ prediction$(\mathbf{w}, \mathbf{x})$
  $\mathbf{w}$ += update$(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

Special case:
multi-prototype features

$$\mathbf{f}(\mathbf{x}, y) \stackrel{\text{def}}{=} \begin{array}{l} [y = class_1] \cdot \mathbf{x}\,, \\ [y = class_2] \cdot \mathbf{x}\,, \\ \cdots \\ [y = class_C] \cdot \mathbf{x} \end{array}$$

	Binary Perceptron	Multi-class Perceptron
prediction$(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=}$	$[\mathbf{w} \cdot \mathbf{x} > 0]$	$\arg\max_y \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
update$(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=}$	$\alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$	$\alpha\big(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\big)$

$$\mathbf{w} := \mathbf{w} + \alpha\mathbf{f}(\mathbf{x}, y_{gold}) - \alpha\mathbf{f}(\mathbf{x}, y_{pred})$$

Perceptron

$\mathbf{w} := 0$
while $(\mathbf{x}, y_{gold}) :=$ get_new_data()
    $y_{pred} := \text{prediction}(\mathbf{w}, \mathbf{x})$
    $\mathbf{w} \mathrel{+}= \text{update}(\mathbf{x}, y_{gold}, y_{pred})$

Output: $\mathbf{w}$

General case:
any *label-dependent* features, e.g.

$$f_{101}(\mathbf{x}, y) \stackrel{\text{def}}{=} [(y{=}\text{NNP or } y{=}\text{NNPS}) \text{ and } \mathbf{x} \text{ capitalized }]$$

	Binary Perceptron	Multi-class Perceptron
$\text{prediction}(\mathbf{w}, \mathbf{x}) \stackrel{\text{def}}{=}$	$[\mathbf{w} \cdot \mathbf{x} > 0]$	$\arg\max_y \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
$\text{update}(\mathbf{x}, y_{gold}, y_{pred}) \stackrel{\text{def}}{=}$	$\alpha(y_{gold} - y_{pred}) \cdot \mathbf{x}$	$\alpha\big(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\big)$

Structured Prediction

- the number of possible labels is huge
- labels $\mathbf{y}$ have a structure (graph, tree, sequence,...)
- usually can be decomposed (factorized) into subproblems
- local features
  - $f_i(\mathbf{x}, \mathbf{y}, j)$ can use whole $\mathbf{x}$, but only such y_k where k is "near" j
  - $f_{101}(\mathbf{x}, \mathbf{y}, j) \stackrel{\text{def}}{=} [\ (y_j{=}\text{NNP or } y_j{=}\text{NNPS}) \text{ and word } x_j \text{ capitalized}\ ]$
  - $f_{102}(\mathbf{x}, \mathbf{y}, j) \stackrel{\text{def}}{=} [\ y_j{=}\text{NNP and } y_{j-1}{=}\text{NNP and } |\mathbf{x}| \leq 6\ ]$
- global features
  - $F_i(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \sum_j f_i(\mathbf{x}, \mathbf{y}, j)$
  - F_{101} ... number of capitalized words with tag NNP or NNPS
  - F_{102} ... number of NNP followed by NNP
    or 0 if the sentence is longer than six words
  - We can define also features that cannot be decomposed

Structured Prediction using Online Learning

1. local approach
   - update after each local decision
   - output of previous decisions used in local features
   - e.g. Structured Perceptron (Collins, 2002)
   - $y_{pred} = \arg\max_y \sum_i w_i f_i(\mathbf{x}, y_j, y_{j-1}, \ldots)$

2. global approach
   - generate n-best list (lattice) of outputs **y** for the whole **x**
   - compute global features, do update for each **x** (sentence)
   - we are re-ranking the n-best list
   - e.g. MIRA (Crammer and Singer, 2003)
   - $\mathbf{y}_{pred} = \arg\max_{\mathbf{y}} \sum_i w_i F_i(\mathbf{x}, \mathbf{y})$

Margin-based Online Learning

Definitions

- $\textit{score}(y) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
- $\textit{margin}(y) = \textit{score}(y_{gold}) - \textit{score}(y)$
  - $\textit{margin} > 0 \Rightarrow$ no error
  - $|\textit{margin}| \sim$ confidence
- $\textit{hinge\_loss}(y) = \textit{max}(0, 1 - \textit{margin}(y))$

Online Prediction and Update

$$y_{pred} \stackrel{\text{def}}{=} \arg\max \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$$
$$\mathbf{w} \mathrel{+}= \alpha\big(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\big)$$

Margin-based Online Learning

Definitions

- $score(y) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
- $margin(y) = score(y_{gold}) - score(y)$
  - $margin > 0 \Rightarrow$ no error
  - $|margin| \sim$ confidence
- $hinge\_loss(y) = max\left(0, 1 - margin(y)\right)$

Online Prediction and Update

$$y_{pred} \stackrel{\text{def}}{=} \arg\max \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$$
$$\mathbf{w} \mathrel{+}= \alpha\left(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\right)$$

Margin-based Online Learning

Definitions

- $score(y) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
- $margin(y) = score(y_{gold}) - score(y)$
  - $margin > 0 \Rightarrow$ no error
  - $|margin| \sim$ confidence
- $hinge\_loss(y) = max(0, 1 - margin(y))$

Online Prediction and Update

$$y_{pred} \stackrel{\text{def}}{=} \arg\max \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$$

$$\mathbf{w} \mathrel{+}= \alpha(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred}))$$

Margin-based Online Learning

Definitions

- $score(y) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$
- $margin(y) = score(y_{gold}) - score(y)$
  - $margin > 0 \Rightarrow$ no error
  - $|margin| \sim$ confidence
- $hinge\_loss(y) = max(0, 1 - margin(y))$

Online Prediction and Update

$$y_{pred} \stackrel{\text{def}}{=} \arg\max \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$$
$$\mathbf{w} \mathrel{+}= \alpha(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred}))$$

Perceptron

$$\alpha_{\text{Perc}} \stackrel{\text{def}}{=} 1 \text{ (or any fixed value } > 0)$$

Passive Aggressive (PA)

$$\alpha_{\text{PA}} \stackrel{\text{def}}{=} \frac{hinge\_loss(y_{pred})}{||\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})||^2}$$

Passive Aggressive I

$$\alpha_{\text{PA-I}} \stackrel{\text{def}}{=} min\{C, \alpha_{\text{PA}}\}$$

Passive Aggressive II

$$\alpha_{\text{PA-II}} \stackrel{\text{def}}{=} \frac{hinge\_loss(y_{pred})}{||\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})||^2 + \frac{1}{2C}}$$

Margin-based Online Learning

Definitions

Online Prediction and Update

$$y_{pred} \stackrel{\text{def}}{=} \arg\max \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, y)$$

$$\mathbf{w} \mathrel{+}= \alpha\left(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\right)$$

Perceptron

$$\alpha_{\text{Perc}} \stackrel{\text{def}}{=} 1 \text{ (or any fixed value } > 0)$$

Passive Aggressive (PA)

$$\alpha_{\text{PA}} \stackrel{\text{def}}{=} \frac{hinge\_loss(y_{pred})}{||\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})||^2}$$

Passive Aggressive I

$$\alpha_{\text{PA-I}} \stackrel{\text{def}}{=} min\{C, \alpha_{\text{PA}}\}$$

Passive Aggressive II

$$\alpha_{\text{PA-II}} \stackrel{\text{def}}{=} \frac{hinge\_loss(y_{pred})}{||\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})||^2 + \frac{1}{2C}}$$

Cost-sensitive Online Learning

Definitions

- $cost(y)$ = external error metric (non-negative)
  e.g. 1 - similarity of y and y_{gold}
- $hinge\_loss(y) = max\big(0, cost(y) - margin(y)\big)$

Hope and Fear

- $\mathbf{w} \mathrel{+}= \alpha\big(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred})\big)$
- min-cost $y_{hope} \stackrel{\text{def}}{=} \arg\max_y -cost(y)$
- max-score $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y)$
- cost-diminished $y_{hope} \stackrel{\text{def}}{=} \arg\max_y score(y) - cost(y)$
- cost-augmented $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y) + cost(y)$
- max-cost $y_{fear} \stackrel{\text{def}}{=} \arg\max_y cost(y)$

Cost-sensitive Online Learning

Definitions

- $cost(y)$ = external error metric (non-negative)
  e.g. 1 - similarity of y and y_{gold}
- $hinge\_loss(y) = max(0, cost(y) - margin(y))$

Hope and Fear

- $\mathbf{w} \mathrel{+}= \alpha(\mathbf{f}(\mathbf{x}, y_{gold}) - \mathbf{f}(\mathbf{x}, y_{pred}))$
- min-cost $y_{hope} \stackrel{\text{def}}{=} \arg\max_y -cost(y)$
- max-score $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y)$
- cost-diminished $y_{hope} \stackrel{\text{def}}{=} \arg\max_y score(y) - cost(y)$
- cost-augmented $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y) + cost(y)$
- max-cost $y_{fear} \stackrel{\text{def}}{=} \arg\max_y cost(y)$

Cost-sensitive Online Learning

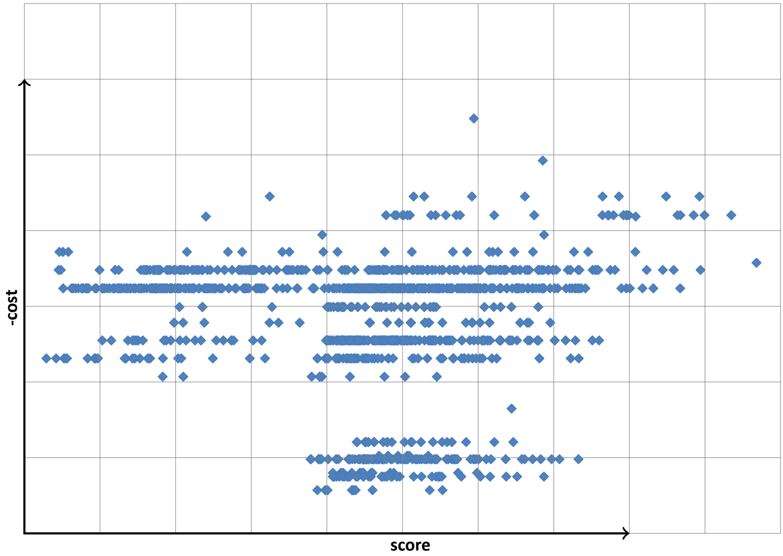
Definitions

- $cost(y)$ = external error metric (non-negative)
  e.g. 1 - similarity of y and y_{gold}
- $hinge\_loss(y) = max\big(0, cost(y) - margin(y)\big)$

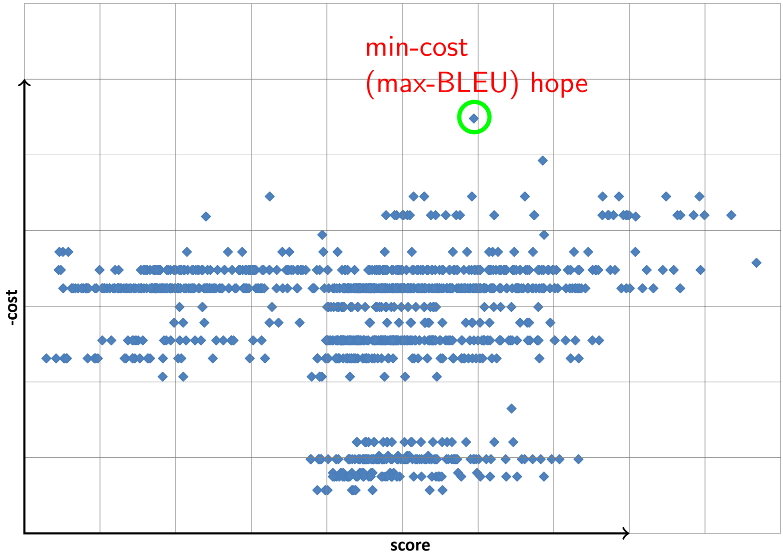
Hope and Fear

- $\mathbf{w} \mathrel{+}= \alpha\big(\mathbf{f}(\mathbf{x}, y_{hope}) - \mathbf{f}(\mathbf{x}, y_{fear})\big)$
- min-cost $y_{hope} \stackrel{\text{def}}{=} \arg\max_y -cost(y)$
- max-score $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y)$
- cost-diminished $y_{hope} \stackrel{\text{def}}{=} \arg\max_y score(y) - cost(y)$
- cost-augmented $y_{fear} \stackrel{\text{def}}{=} \arg\max_y score(y) + cost(y)$
- max-cost $y_{fear} \stackrel{\text{def}}{=} \arg\max_y cost(y)$

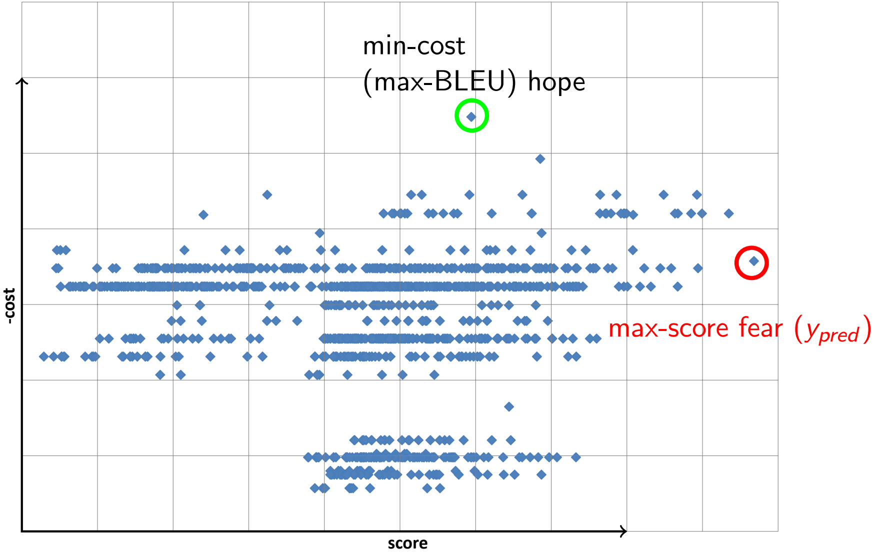
Cost-sensitive Online Learning



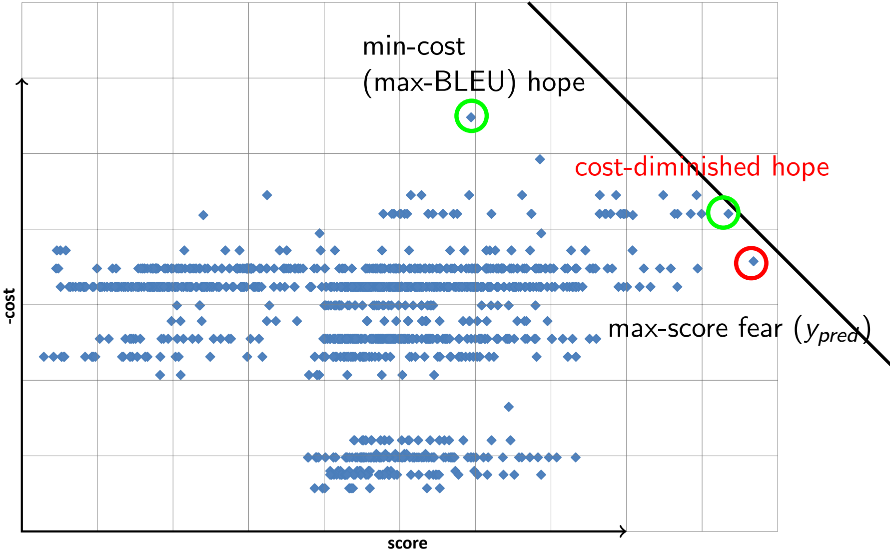
Cost-sensitive Online Learning



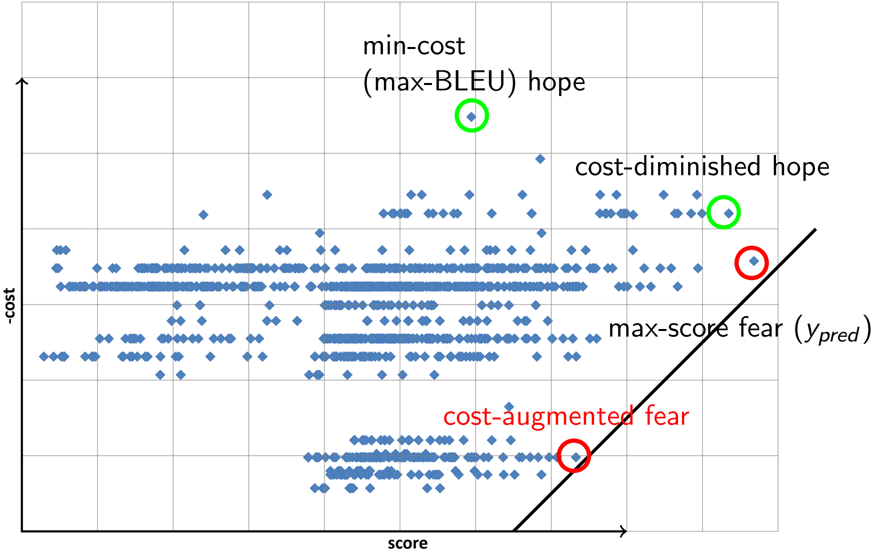
Cost-sensitive Online Learning



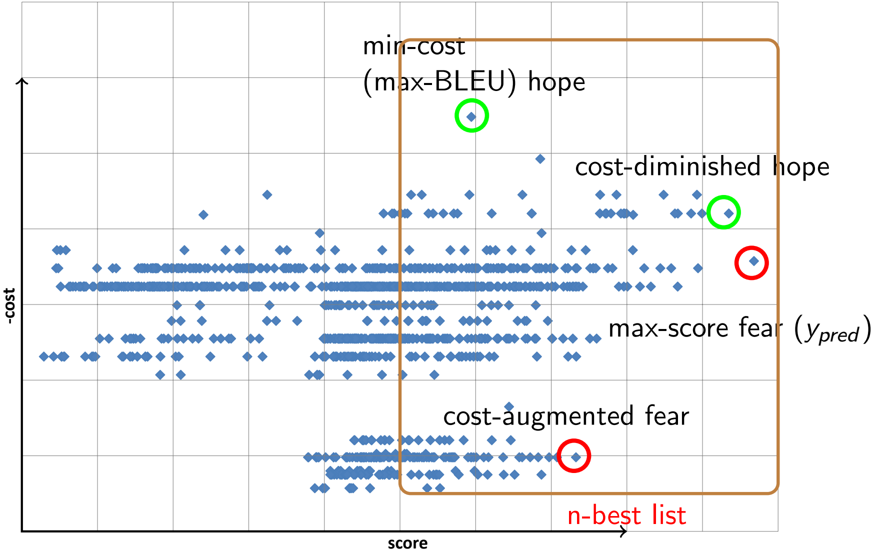
Cost-sensitive Online Learning



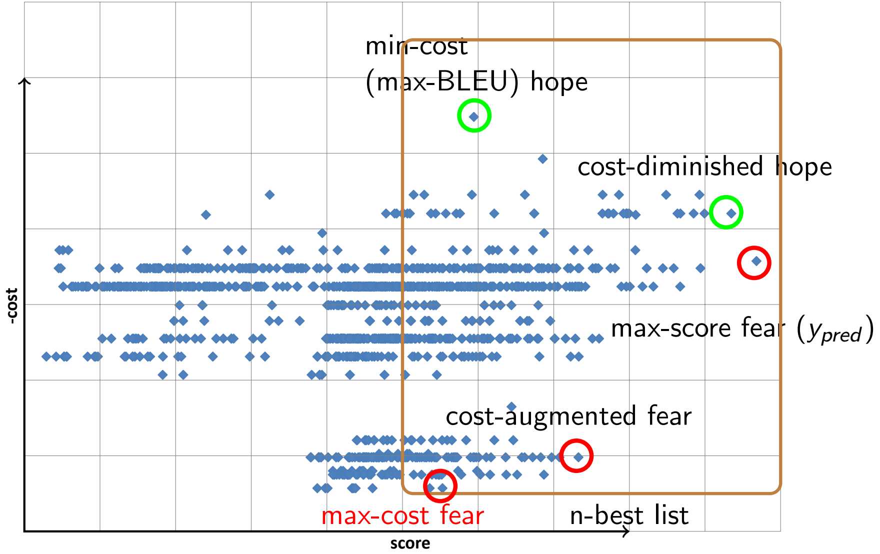
Cost-sensitive Online Learning



Cost-sensitive Online Learning



Cost-sensitive Online Learning



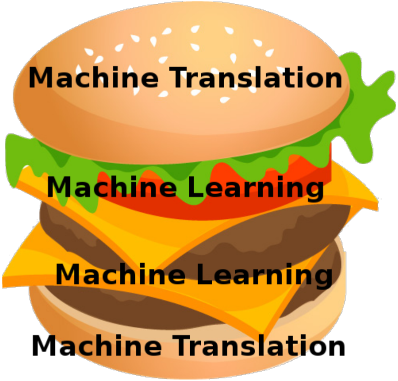
Application to MT

x = source sentence
y_{gold} = its reference translation

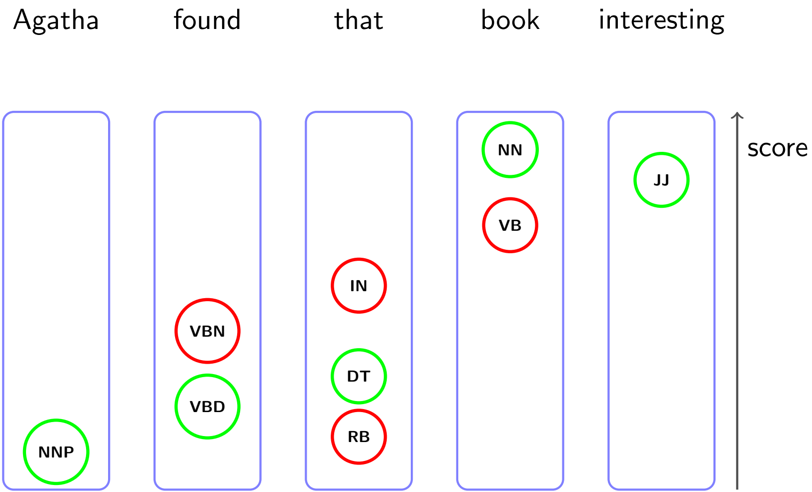
- more references sometimes available
- reference may be *unreachable*
- we score *derivations* (which include latent variables)
  one translation may have more derivations

Today's Menu

1. MT Intro
   - Taxonomy
   - Hybrids
2. Online Learning
   - Perceptron
   - Structured Prediction
3. Guided Learning
4. Back to MT
   - Easy-First Decoding in MT
   - Guided Learning in MT

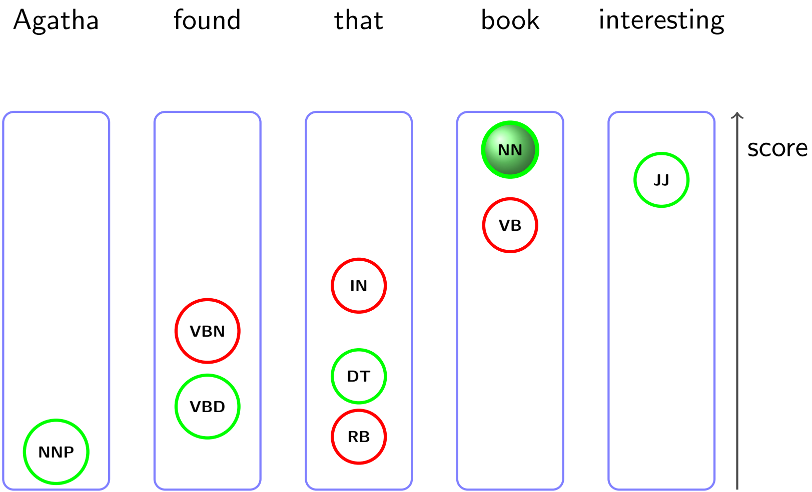


Easy-First Decoding (PoS Tagging)



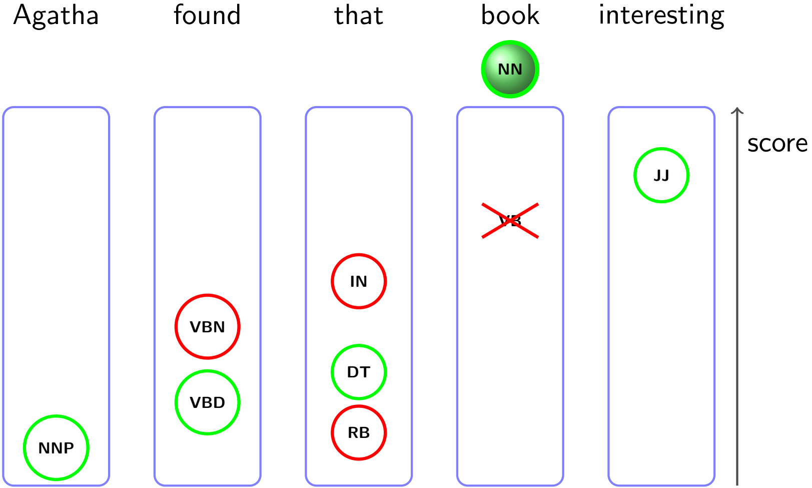
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



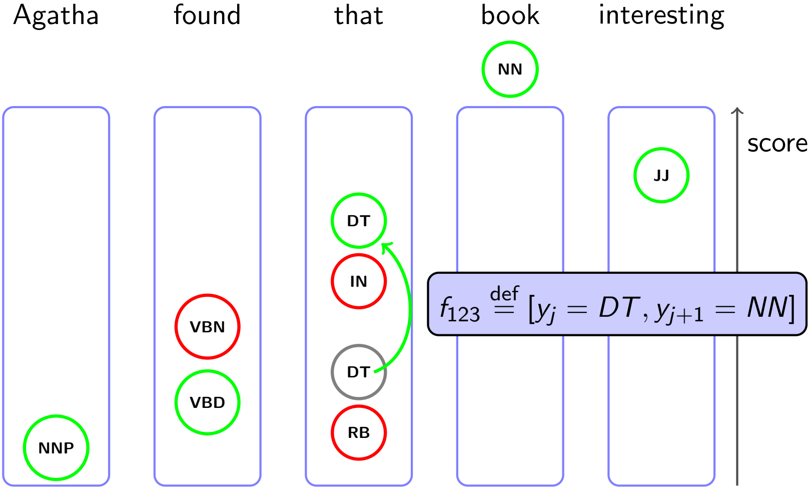
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



(Shen, Satta and Joshi, 2007)

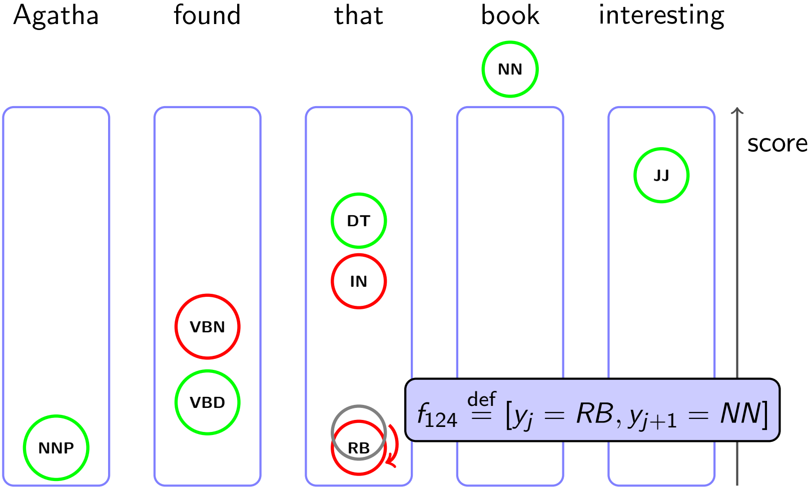
Easy-First Decoding (PoS Tagging)



$$f_{123} \stackrel{\text{def}}{=} [y_j = DT, y_{j+1} = NN]$$

(Shen, Satta and Joshi, 2007)

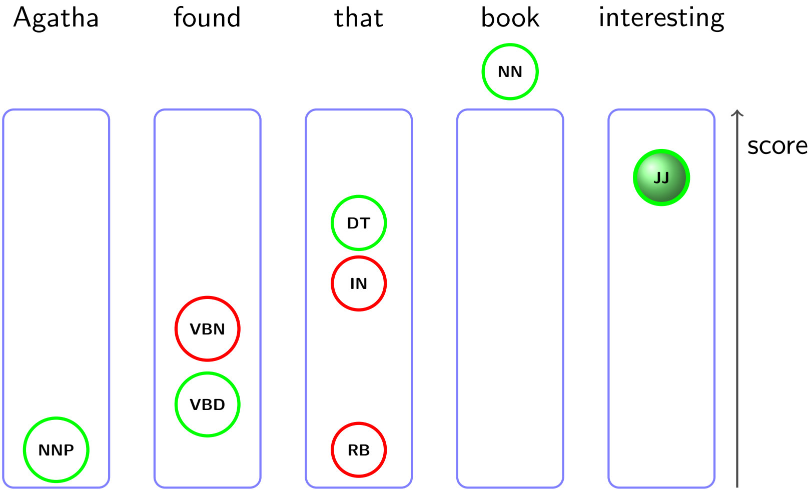
Easy-First Decoding (PoS Tagging)



$$f_{124} \stackrel{\text{def}}{=} [y_j = RB, y_{j+1} = NN]$$

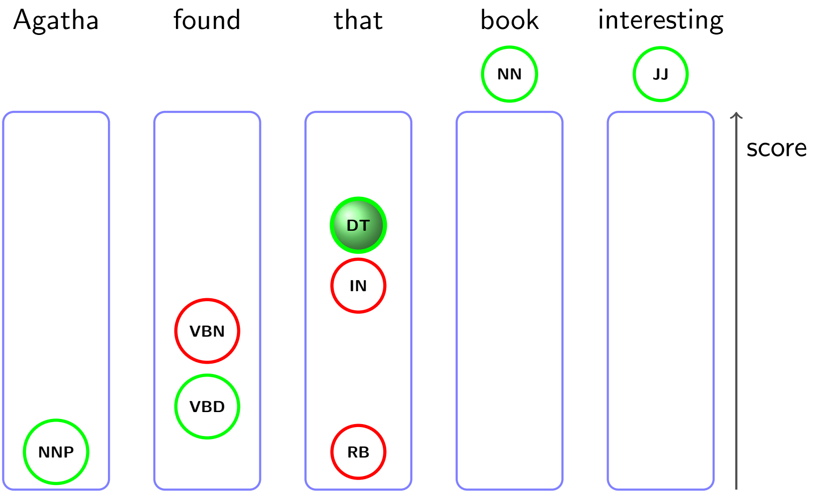
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



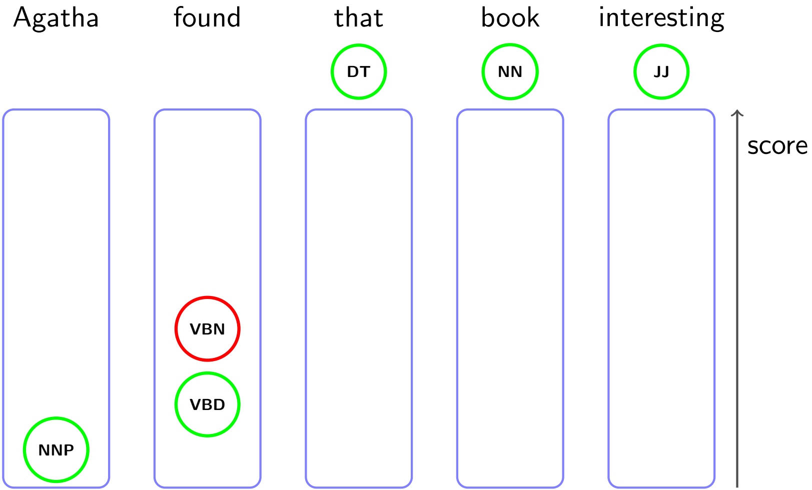
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



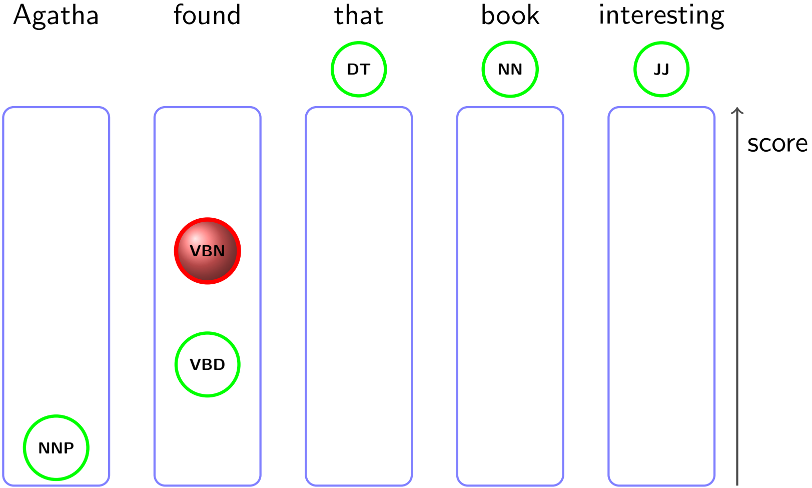
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



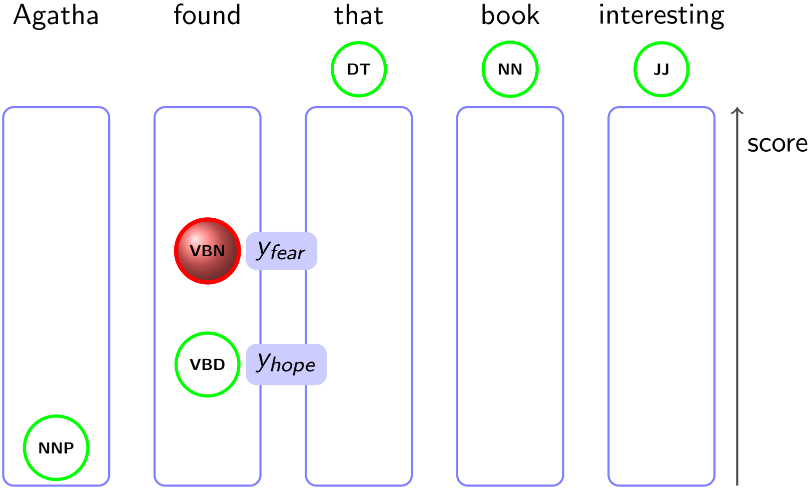
(Shen, Satta and Joshi, 2007)

Easy-First Decoding (PoS Tagging)



(Shen, Satta and Joshi, 2007)

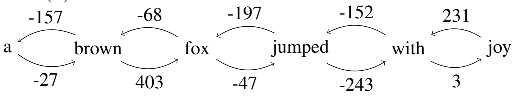
Guided Learning (PoS Tagging)



(Shen, Satta and Joshi, 2007)

Easy-First Decoding (Dependency Parsing)

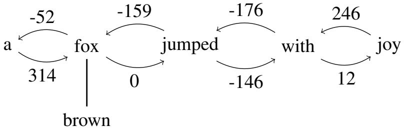
(1) ATTACHRIGHT(2)



(Goldberg and Elhadad, 2010)

Easy-First Decoding (Dependency Parsing)

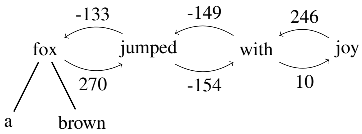
(2) ATTACHRIGHT(1)



(Goldberg and Elhadad, 2010)

Easy-First Decoding (Dependency Parsing)

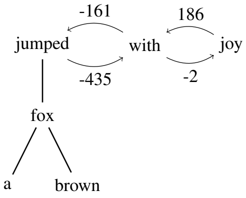
(3) ATTACHRIGHT(1)



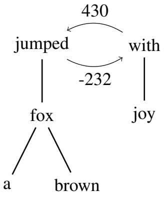
(Goldberg and Elhadad, 2010)

Easy-First Decoding (Dependency Parsing)

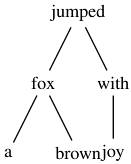
(4) ATTACHLEFT(2)



(5) ATTACHLEFT(1)



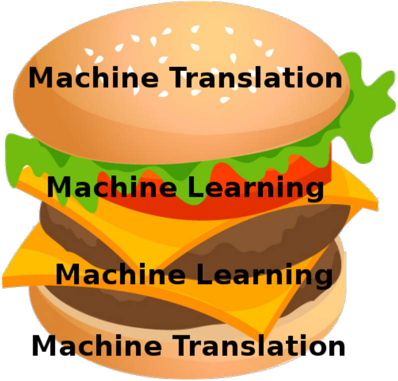
(6)



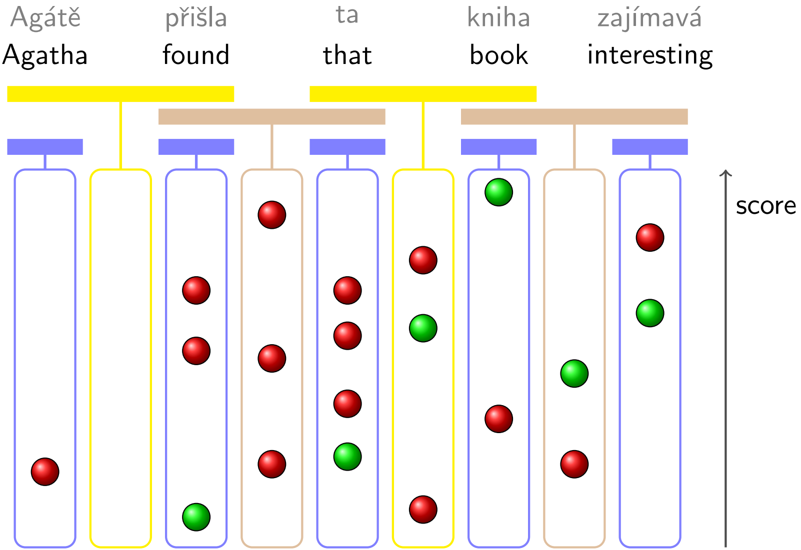
(Goldberg and Elhadad, 2010)

Today's Menu

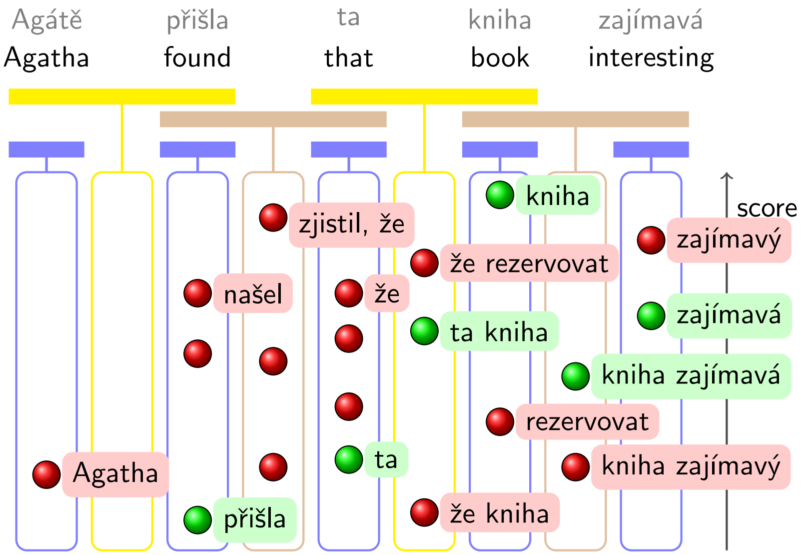
1. MT Intro
   - Taxonomy
   - Hybrids
2. Online Learning
   - Perceptron
   - Structured Prediction
3. Guided Learning
4. Back to MT
   - Easy-First Decoding in MT
   - Guided Learning in MT



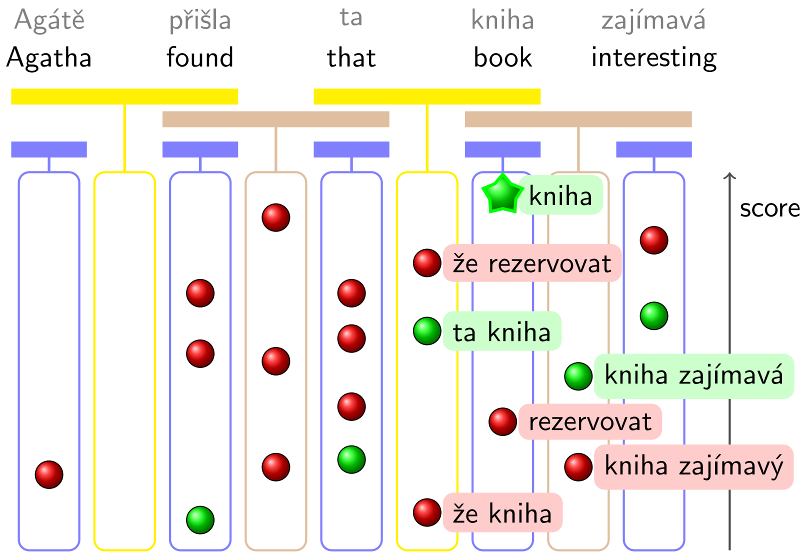
Easy-First Decoding (Phrase-Based MT)



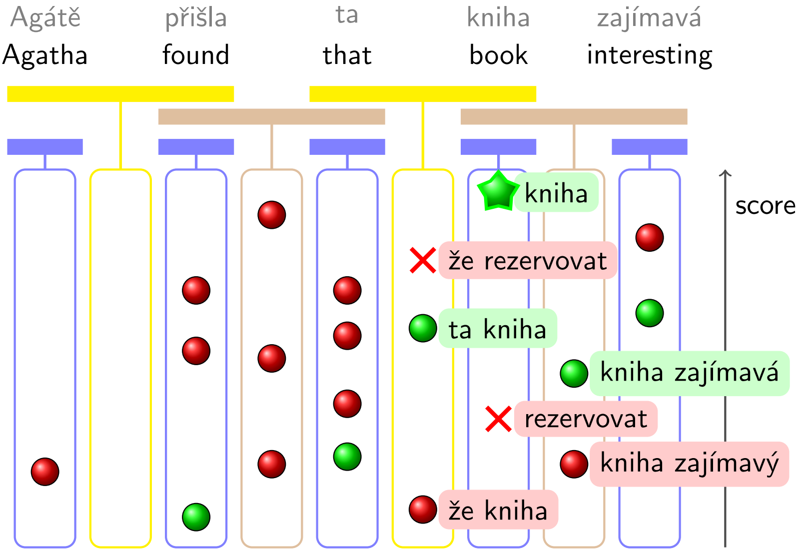
Easy-First Decoding (Phrase-Based MT)



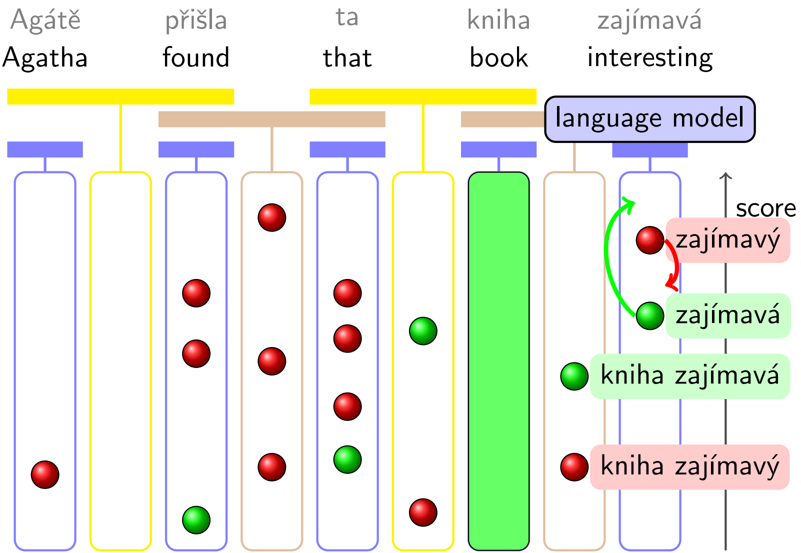
Easy-First Decoding (Phrase-Based MT)



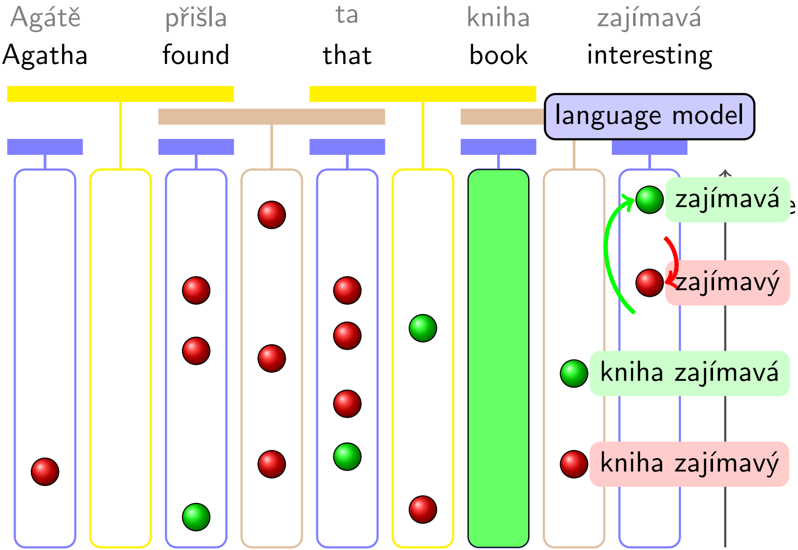
Easy-First Decoding (Phrase-Based MT)



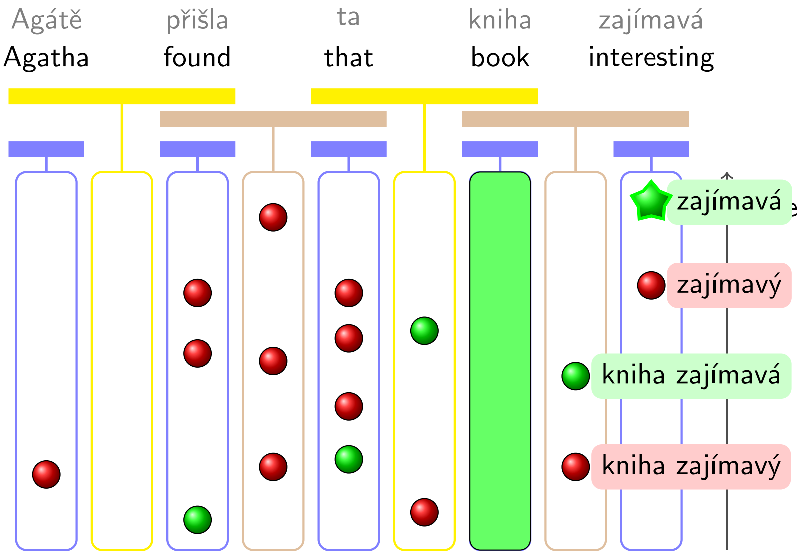
Easy-First Decoding (Phrase-Based MT)



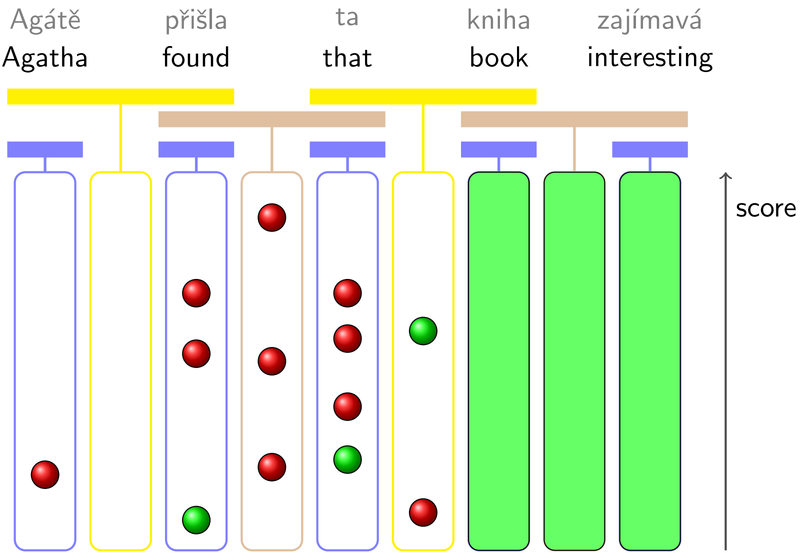
Easy-First Decoding (Phrase-Based MT)



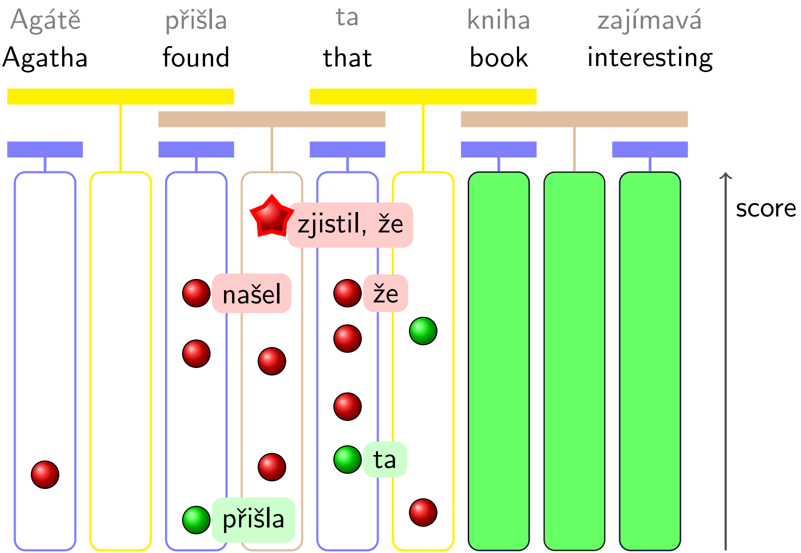
Easy-First Decoding (Phrase-Based MT)



Easy-First Decoding (Phrase-Based MT)



Easy-First Decoding (Phrase-Based MT)



Features for Guided Learning in MT

Source Segment Features

- segment size (number of words)
- entropy $P(target|source) = -\sum_i P(src, trg_i) \cdot \log P(trg_i|src)$
- log count(source)
- source language model: log P(source)
- word identity, e.g. $f_{42} \stackrel{\text{def}}{=}$ [src=found that]
- PoS identity, e.g. $f_{43} \stackrel{\text{def}}{=}$ [src_pos=VBD IN]

Target-dependent Features

- log P(trg|src)
- target language model: log P(target | previous segment)
- log count(target)?
- identity, e.g. $f_{142} \stackrel{\text{def}}{=}$ [src=found that & trg=zjistil]

Features for Guided Learning in MT

Source Segment Features

- segment size (number of words)
- entropy $P(target|source) = -\sum_i P(src, trg_i) \cdot \log P(trg_i|src)$
- log count(source)
- source language
- word identity, e.
- PoS identity, e.g

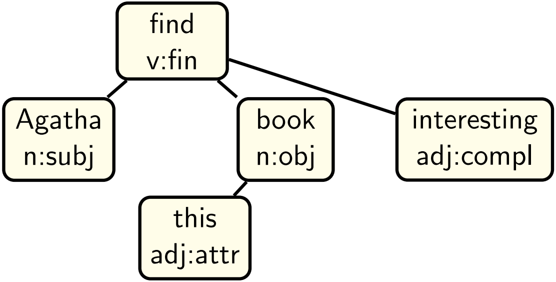
Combinations and Quantizations

$[size(src) = 3] \cdot \log P(trg|src)$
$[size(src) = 3 \quad \& -3 < \log P(trg|src) < -2]$
etc.

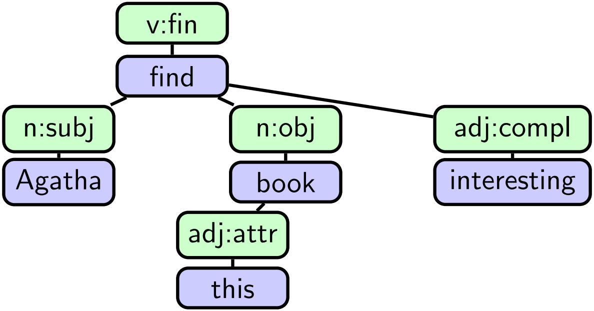
Target-dependent Features

- log P(trg|src)
- target language model: log P(target | previous segment)
- log count(target)?
- identity, e.g. $f_{142} \stackrel{\text{def}}{=}$ [src=found that & trg=zjistil]

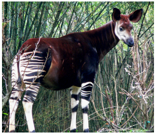
Application to Tecto Trees

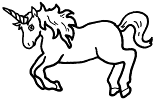


Application to Tecto Trees

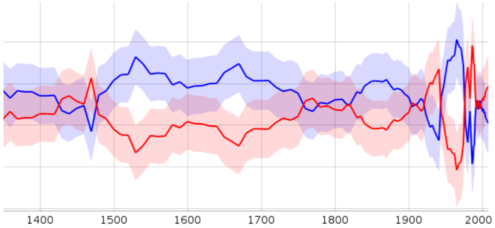


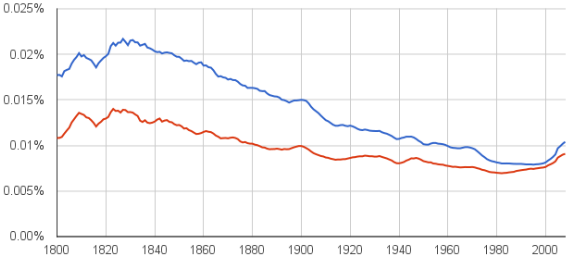
What have you seen in the Zoo





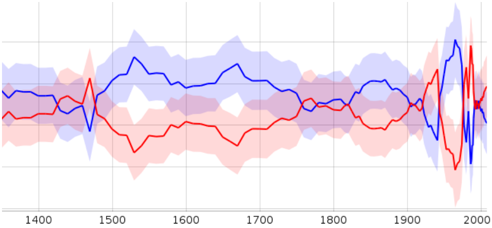
Predictions?



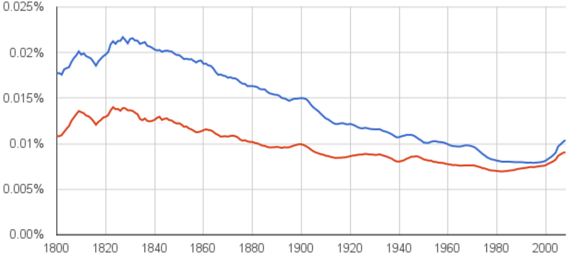


Predictions?

Hope and Fear



http://syd.korpus.cz/



http://books.google.com/ngrams/