Weighted Automata Techniques in Statistical Machine Translation

Andreas Maletti

Institute of Computer Science
Universität Leipzig, Germany

maletti@informatik.uni-leipzig.de

Leipzig — May 8, 2014

Statistical Machine Translation

Definition

Statistical machine translation:

- translation of natural language text by computers
- using statistical models
- learnt from parallel corpora.

Statistical Machine Translation

Definition

Statistical machine translation:

- translation of natural language text by computers
- using ~~statistical models~~ weighted automata
- learnt from parallel corpora.

Parallel Corpus

We can help countries catch up, but not by putting their neighbors on hold.	Wir können Ländern beim Aufholen helfen, aber nicht, indem wir ihre Nachbarn in den Wartesaal schicken.
We must bear in mind the Community as a whole.	Wir müssen uns davor hüten, alles vergemeinschaften zu wollen

Parallel Corpus

EUROPARL German-English parallel corpus

- 2 million parallel sentences
- 45 million words in German
- 48 million words in English
- parliament proceedings

MULTIUN Chinese-English parallel corpus

- 10 million parallel sentences
- 257 million words in English
- official UN documents

Noisy-Channel Model

Example (Input in Catalan)

Benvolguda i benvolgut membre de la comunitat universitària,
Avui dilluns es duu a terme el darrer Consell de Govern del meu mandat com a rector; el proper dia 6 de maig, com correspon, hi haurà una nova elecció on tota la comunitat universitària podrà escollir nou rector o rectora.
Aquest darrer consell té, naturalment, un caràcter marcadament tècnic; l'ordre del dia complet el trobaràs adjunt al final d'aquest text. A continuació et comento només els punts que, al meu parer, poden ser més del teu interès.

Translation (Google Translate) to English

Dear and beloved member of the university community,
Today is Monday carried out by the Governing Council last of my term as rector, the next day, May 6, as appropriate, there will be another election where the entire university community can choose new rector.
This last advice is, of course, a markedly technician complete agenda can be found attached to the end of this text. Then I said only the points that I believe may be of interest.

Noisy-Channel Model

Example (Input in Catalan)

Benvolguda i benvolgut membre de la comunitat universitària,
Avui dilluns es duu a terme el darrer Consell de Govern del meu mandat com a rector; el proper dia 6 de maig, com correspon, hi haurà una nova elecció on tota la comunitat universitària podrà escollir nou rector o rectora.
Aquest darrer consell té, naturalment, un caràcter marcadament tècnic; l'ordre del dia complet el trobaràs adjunt al final d'aquest text. A continuació et comento només els punts que, al meu parer, poden ser més del teu interès.

Translation (GOOGLE TRANSLATE) to English

Dear and beloved member of the university community,
Today is Monday carried out by the Governing Council last of my term as rector, the next day, May 6, as appropriate, there will be another election where the entire university community can choose new rector.
This last advice is, of course, a markedly technician complete agenda can be found attached to the end of this text. Then I said only the points that I believe may be of interest.

Noisy-Channel Model

Input sentence (*Benvolguda i benvolgut ...*)

$\Downarrow$

Translation system

$\Downarrow$

Output sentence (*Dear and beloved ...*)

Noisy-Channel Model

Input sentence (*Benvolguda i benvolgut ...*) F

$$\Downarrow$$

Translation system

$$\Downarrow$$

Output sentence (*Dear and beloved ...*) E_{max}

Statistical translation system

$$E_{\text{max}} = \arg\max_E p(E \mid F)$$

Noisy-Channel Model

Input sentence (*Benvolguda i benvolgut ...*) F

$\Downarrow$

Identity translation	$\Leftarrow$	Error (Noise)

$\Downarrow$

Output sentence (*Dear and beloved ...*) $E_{\max}$

Noisy-Channel Model

Input sentence (*Benvolguda i benvolgut ...*) F

$\Downarrow$

Identity translation $\Leftarrow$ Error (Noise)

$\Downarrow$

Output sentence (*Dear and beloved ...*) E_{max}

Bayes' theorem

$$E_{\text{max}} = \arg\max_E p(E|F) = \arg\max_E \frac{p(F|E) \cdot p(E)}{p(F)}$$

$$= \arg\max_E p(F|E) \cdot p(E)$$

Noisy-Channel Model

Optimization problem

$$E_{\max} = \arg\max_E p(F|E) \cdot p(E)$$

Required models

- $p(E)$ — language model
  (weighted automaton — determinization, minimization, etc.)
- $p(F|E)$ — translation model

Noisy-Channel Model

Optimization problem

$$E_{\text{max}} = \arg\max_E p(F|E) \cdot p(E)$$

Required models

- $p(E)$ — language model
  (weighted automaton — determinization, minimization, etc.)
- $p(F|E)$ — translation model

Language Model

Toolkits

- SRILM
  https://code.google.com/p/moses-suite/downloads/detail?name=srilm-1.6.0.tar.gz&can=2&q=
- IRSTLM https://hlt.fbk.eu/technologies/irstlm-irst-language-modelling-toolkit
- KenLM kheafield.com/code/kenlm/

Language models are readily available

Translation Model

What about the translation model?

Translation Model

Existing translation models

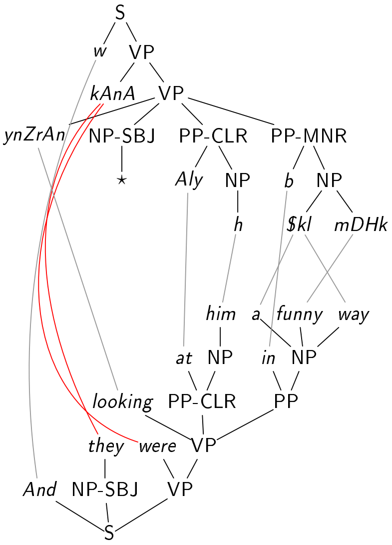
- word-based models (IBM models)
- phrase-based models
- hierarchical phrase-based models
- syntax-based models
- semantics-based models

Translation Model

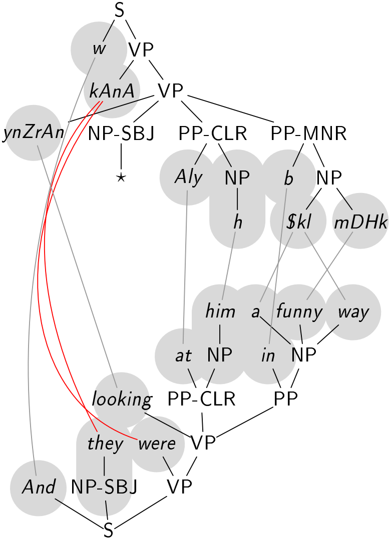
Existing translation models

- word-based models (IBM models)
- phrase-based models
- hierarchical phrase-based models
- syntax-based models
- semantics-based models

Syntax-based Statistical Machine Translation



Syntax-based Statistical Machine Translation

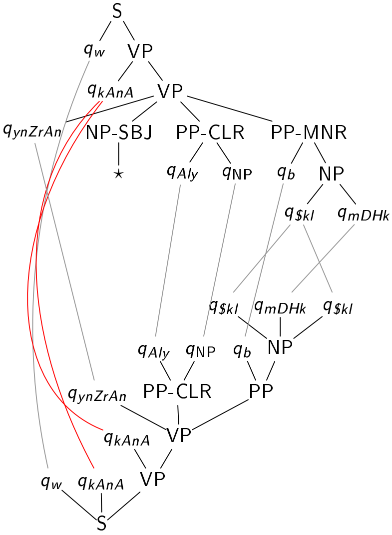


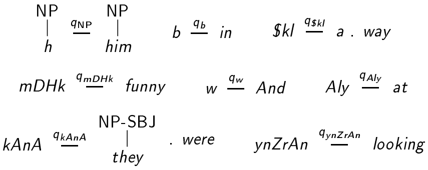
NP(h) $\xrightarrow{q_{\mathrm{NP}}}$ NP(him) b $\xrightarrow{q_b}$ in $kl $\xrightarrow{q_{\$kl}}$ a . way

mDHk $\xrightarrow{q_{mDHk}}$ funny w $\xrightarrow{q_w}$ And Aly $\xrightarrow{q_{Aly}}$ at

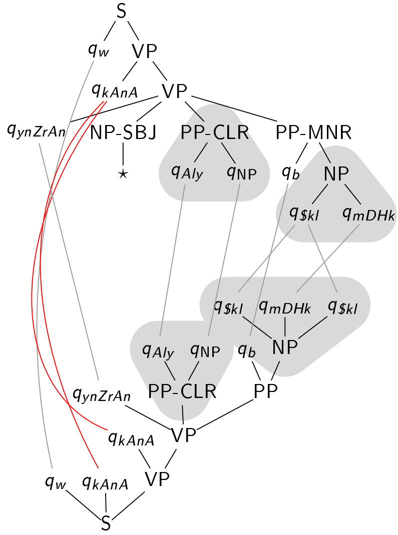
kAnA $\xrightarrow{q_{kAnA}}$ NP-SBJ(they) . were ynZrAn $\xrightarrow{q_{ynZrAn}}$ looking

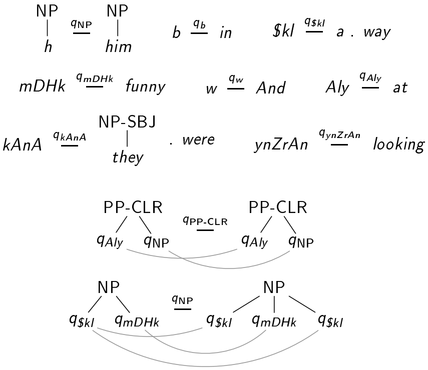
Syntax-based Statistical Machine Translation



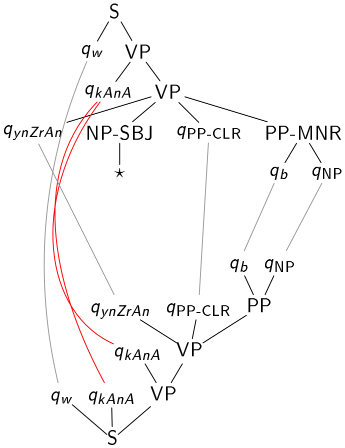


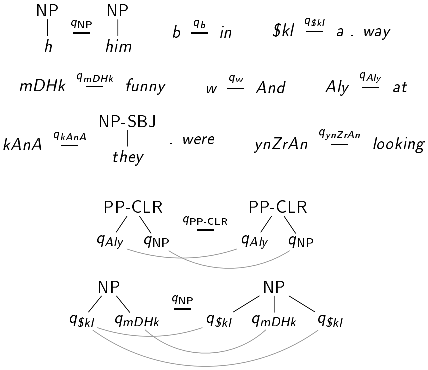
Syntax-based Statistical Machine Translation



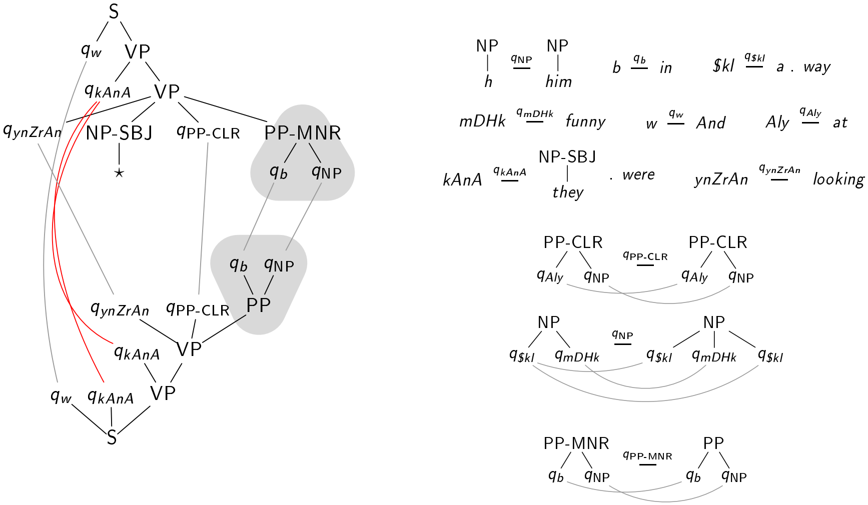


Syntax-based Statistical Machine Translation



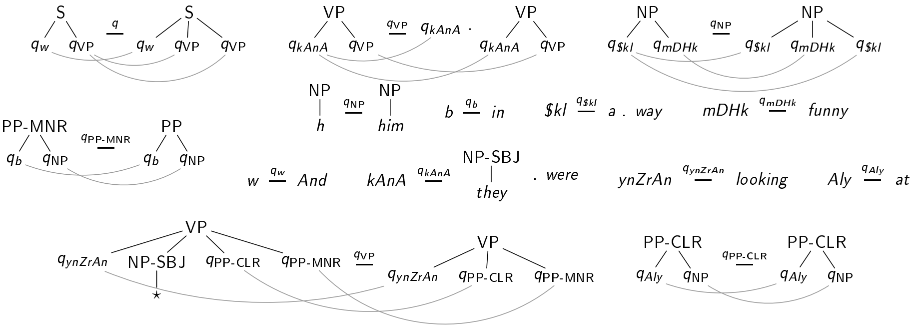


Syntax-based Statistical Machine Translation



Syntax-based Statistical Machine Translation

Extracted rules

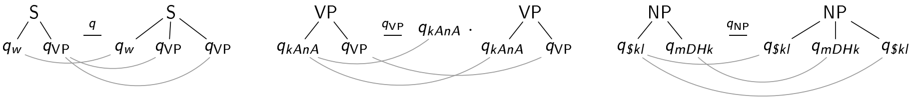


Multi Bottom-up Tree Transducer

Definition (MBOT)

Multi bottom-up tree transducer (Q, Σ, I, R)

- finite set Q — *states*
- alphabet Σ — *input and output symbols*
- $I \subseteq Q$ — *initial states*
- finite set $R \subseteq T_\Sigma(Q) \times Q \times T_\Sigma(Q)^*$ — *rules*
    - each $q \in Q$ occurs at most once in ℓ — $(\ell, q, \vec{r}) \in R$
    - each $q \in Q$ that occurs in $\vec{r}$ also occurs in ℓ — $(\ell, q, \vec{r}) \in R$



Multi Bottom-up Tree Transducer

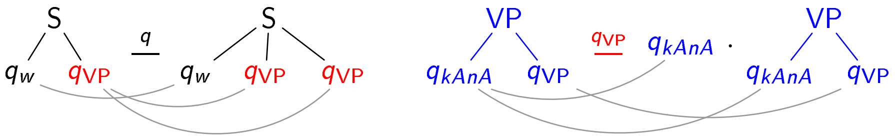
Definition (MBOT)

Multi bottom-up tree transducer $(Q, \Sigma, I, R, \mathrm{wt})$

- finite set Q *states*
- alphabet Σ *input and output symbols*
- $I \subseteq Q$ *initial states*
- finite set $R \subseteq T_\Sigma(Q) \times Q \times T_\Sigma(Q)^*$ *rules*
  - each $q \in Q$ occurs at most once in ℓ $(\ell, q, \vec{r}) \in R$
  - each $q \in Q$ that occurs in $\vec{r}$ also occurs in ℓ $(\ell, q, \vec{r}) \in R$
- $\mathrm{wt}\colon R \to \mathbb{Q}$ weight assignment

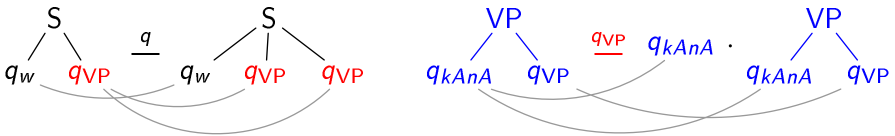
Semantics — Synchronous Generation

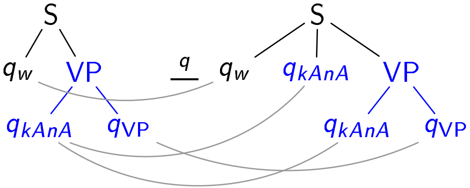
Rules



Semantics — Synchronous Generation

Rules





Semantics — Synchronous Generation

Definition (Generation step)

$$\langle t, A, u \rangle \Rightarrow_M \langle t', A', u' \rangle$$

if and only if $\exists q \in Q$, $\exists v \in \mathrm{pos}(t)$ labeled by q, and $\exists \ell \stackrel{q}{\to} \vec{r} \in R$

- $|\vec{r}| = |A(v)|$ and $\vec{w} = \vec{A(v)}$
- $t' = t[\ell]_v$ and $u' = u[\vec{r}]_{\vec{w}}$
- $A' = (A \setminus L) \cup \mathrm{links}_{v,\vec{w}}(\ell \stackrel{q}{\to} \vec{r})$ with

$$L = \{(v, w) \mid w \in A(v)\}$$

Semantics — Synchronous Generation

Definition

- Weight of a generation step sequence
  is the product of the weights of the used rules
- Initial situation $\langle q, \{(\varepsilon, \varepsilon)\}, q\rangle$ for some $q \in I$
- All states must disappear in a complete generation
- Sum over all weights of generations generating the same tree pair
- Need to use deterministic generation strategy

Setting the Weights

Scoring

- during rule extraction we keep a count $c(\rho)$ how often the rule $\rho \in R$ was extracted
- given an equivalence $\equiv$ on rules (same left-hand side, etc.)
- we set

$$\mathrm{wt}(\rho) = \frac{c(\rho)}{\sum_{\rho' \in [\rho]_\equiv} c(\rho')}$$

We have the translation model

Decoding

Pipeline

Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

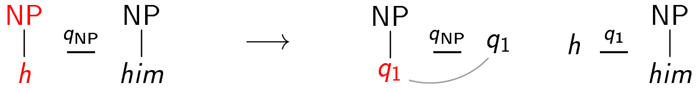
Decode Preparation

One-symbol normal form

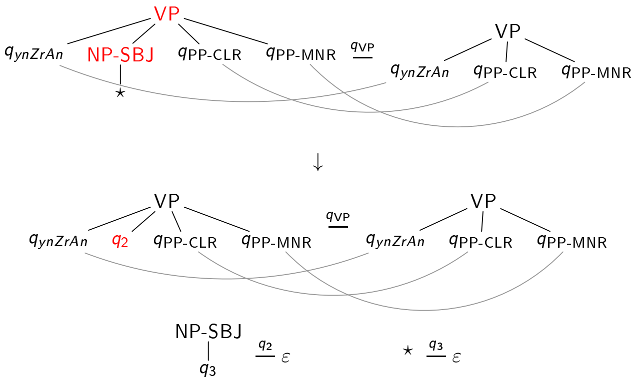
MBOT $(Q, \Sigma, I, R, \mathrm{wt})$ in one-symbol normal form
if ℓ contains at most one (occurrence of a) symbol of Σ

Theorem

For every MBOT there exists an equivalent MBOT in one-symbol normal form



Decode Preparation



Transformation into one-symbol normal form

Linear-time procedure

Decoding

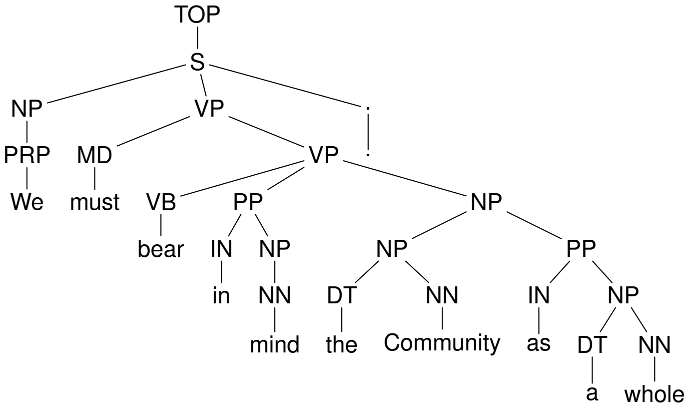
Pipeline

Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

Parser

Notes

- constructs parses for sentences
- each parse also has a score



Regular Tree Grammar

Definition (RTG)

Regular tree grammar (Q, Σ, I, R)

- finite set Q — *states*
- alphabet Σ — *input symbols*
- $I \subseteq Q$ — *initial states*
- finite set $R \subseteq T_\Sigma(Q) \times Q$ — *rules*

Regular Tree Grammar

Definition (RTG)

Regular tree grammar $(Q, \Sigma, I, R, \mathrm{wt})$

- finite set Q *states*
- alphabet Σ *input symbols*
- $I \subseteq Q$ *initial states*
- finite set $R \subseteq T_\Sigma(Q) \times Q$ *rules*
- $\mathrm{wt} \colon R \to \mathbb{Q}$ weight assignment

Regular Tree Grammar

Definition (RTG)

Regular tree grammar $(Q, \Sigma, I, R, \mathrm{wt})$

- finite set Q *states*
- alphabet Σ *input symbols*
- $I \subseteq Q$ *initial states*
- finite set $R \subseteq T_\Sigma(Q) \times Q$ *rules*
- $\mathrm{wt}\colon R \to \mathbb{Q}$ weight assignment

Notes

- MBOT without output
- can also be brought into one-symbol normal form

$\to$ (weighted) tree automaton

Parser

Productions of the Berkeley parser

S-1 $\rightarrow$ ADJP-2 S-1	$0.0035453455987323125 \cdot 10^{0}$
S-1 $\rightarrow$ ADJP-1 S-1	$2.108608433271444 \cdot 10^{-6}$
S-1 $\rightarrow$ VP-5 VP-3	$1.6367163259885093 \cdot 10^{-4}$
S-2 $\rightarrow$ VP-5 VP-3	$9.724998692152419 \cdot 10^{-8}$
S-1 $\rightarrow$ PP-7 VP-0	$1.0686659961009547 \cdot 10^{-5}$
S-9 $\rightarrow$ “ NP-3	$0.012551243773149695 \cdot 10^{0}$

Parser

Toolkits

- Berkeley parser
  `https://code.google.com/p/berkeleyparser/`
- BitPar
  `www.cis.uni-muenchen.de/~schmid/tools/BitPar/`
- MateTools `https://code.google.com/p/mate-tools/`
- Egret
  `https://sites.google.com/site/zhangh1982/egret`

Parsers are readily available

Parser

Implementation

- is often a weighted tree automaton
- yields an (unambiguous) weighted tree automaton for the parses of the input sentence
- efficient representation of $L\colon T_\Sigma \to \mathbb{Q}$

Example (Berkeley parser — English grammar)

S-1 $\to$ ADJP-2 S-1	$0.0035453455987323125 \cdot 10^{0}$
S-1 $\to$ ADJP-1 S-1	$2.108608433271444 \cdot 10^{-6}$
S-1 $\to$ VP-5 VP-3	$1.6367163259885093 \cdot 10^{-4}$
S-2 $\to$ VP-5 VP-3	$9.724998692152419 \cdot 10^{-8}$

Decoding

Pipeline

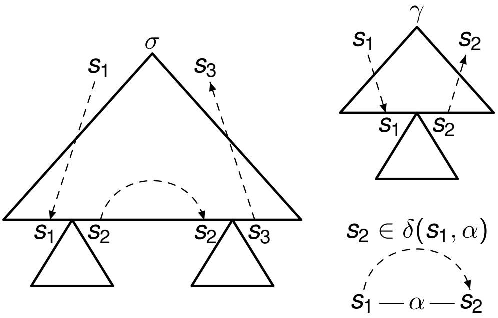
Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

Input Product

Definition

1. weighted translation $\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q}$ and
2. weighted language $p \colon \Sigma^* \to \mathbb{Q}$ (language model)

$$_p\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q} \qquad (t, u) \mapsto \tau(t, u) \cdot p(\mathrm{yd}(t))$$



Input and Output Product

Definition

1. weighted translation $\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q}$ and
2. weighted tree language $L \colon T_\Sigma \to \mathbb{Q}$ (parses)

$$ {}_L\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q} \qquad (t, u) \mapsto \tau(t, u) \cdot L(t) $$

Theorem

... *product of MBOT M with ... is*

side	wA A	wTA A
input	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^3)$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert)$
output	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{2\,\mathrm{rk}(M)+2})$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{\mathrm{rk}(M)})$

Input and Output Product

Definition

1. weighted translation $\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q}$ and
2. weighted tree language $L \colon T_\Sigma \to \mathbb{Q}$ (parses)

$$_L\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q} \qquad (t, u) \mapsto \tau(t, u) \cdot L(t)$$

Theorem

. . . product of MBOT M with . . . is

side	*wA A*	*wTA A*
input	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^3)$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert)$
output	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{2\,\mathrm{rk}(M)+2})$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{\mathrm{rk}(M)})$

Input and Output Product

Definition

1. weighted translation $\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q}$ and
2. weighted tree language $L \colon T_\Sigma \to \mathbb{Q}$ (parses)

$$_L\tau \colon T_\Sigma \times T_\Sigma \to \mathbb{Q} \qquad (t, u) \mapsto \tau(t, u) \cdot L(t)$$

Theorem

. . . product of MBOT M with . . . is

side	wA A	wTA A
input	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^3)$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert)$
output	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{2\,\mathrm{rk}(M)+2})$	$\mathcal{O}(\lvert M\rvert \cdot \lvert A\rvert^{\mathrm{rk}(M)})$

Input Product with wTA

Definition

MBOT $M = (Q, \Sigma, I, R, \text{wt})$ and wTA $A = (P, \Sigma, J, R', \text{wt}')$

- M in 1-symbol normal form
- A in 1-symbol normal form

$$\boxed{{}_A M = (P \times Q, \Sigma, J \times I, R'', \text{wt}'')}$$

with

- input-consuming rules from input-consuming rules of R with parallel processing by A
- ε-rules from ε-rules of R without processing by A

$\rightarrow$ standard product construction

Input Product with wTA

Definition

MBOT $M = (Q, \Sigma, I, R, \mathrm{wt})$ and wTA $A = (P, \Sigma, J, R', \mathrm{wt}')$

- M in 1-symbol normal form
- A in 1-symbol normal form

$$\boxed{{}_A M = (P \times Q, \Sigma, J \times I, R'', \mathrm{wt}'')}$$

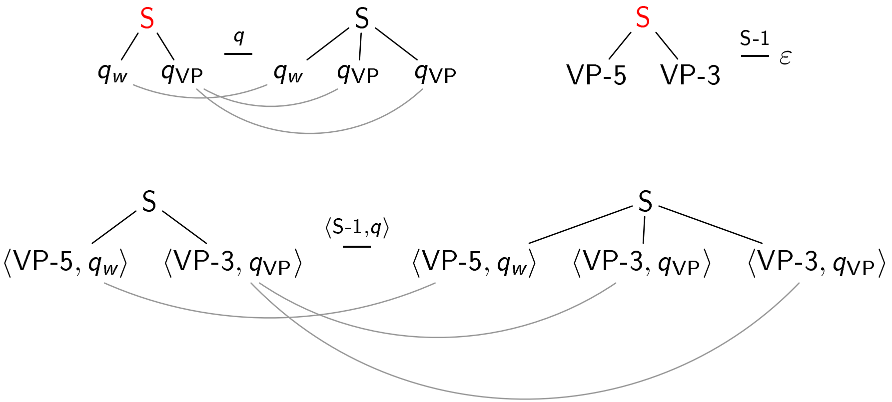
with

- input-consuming rules from input-consuming rules of R with parallel processing by A
- ~~ε-rules from ε-rules of R without processing by A~~

$\rightarrow$ standard product construction

Input Product with wTA

Example



Decoding

Pipeline

Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

- exact decoding of the red part
- integrating the language model would require
  1. in general: $\mathcal{O}(|M| \cdot |A|^{2\,\mathrm{rk}(M)+2})$
  2. for wTA: $\mathcal{O}(|M| \cdot |A|^3)$
  3. for wA: $\mathcal{O}(|M| \cdot |A|)$

Decoding

Pipeline

Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

- exact decoding of the red part
- integrating the language model would require
  1. in general: $\mathcal{O}(|M| \cdot |A|^{2\,\mathrm{rk}(M)+2})$
  2. for wTA: $\mathcal{O}(|M| \cdot |A|^3)$
  3. for wA: $\mathcal{O}(|M| \cdot |A|)$

Decoding

Pipeline

Input $\longrightarrow$ Parser $\longrightarrow$ Translation model $\longrightarrow$ Language model $\longrightarrow$ Output

- exact decoding of the red part
- integrating the language model would require
  1. in general: $\mathcal{O}(|M| \cdot |A|^{2\,\mathrm{rk}(M)+2})$
  2. for wTA: $\mathcal{O}(|M| \cdot |A|^3)$
  3. for wA: $\mathcal{O}(|M| \cdot |A|)$

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1.
2. language model not integrated (needs strict structure)
3. strictness $\rightarrow$ coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1.
2. language model not integrated (needs strict structure)
3. strictness $\to$ coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the *k*-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1.
2. language model not integrated (needs strict structure)
3. strictness → coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1.
2. language model not integrated (needs strict structure)
3. strictness $\rightarrow$ coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s...
2. language model not integrated (needs strict structure)
3. strictness → coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the *k*-best derivations
3. reranks them by the language model (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s. . . l. . .
2. language model not integrated (needs strict structure)
3. strictness → coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s. . . l. . . o . . .
2. language model not integrated (needs strict structure)
3. strictness → coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s. . . l. . . o . . . w
2. language model not integrated (needs strict structure)
3. strictness → coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the *k*-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s. . . l. . . o . . . w
2. language model not integrated (needs strict structure)
3. strictness $\rightarrow$ coverage problems

Exact Decoder

Implementation

1. exactly computes a wTA representing the derivations for the first two models
2. extracts the k-best derivations
3. reranks them by the language model
   (i.e., multiplies their score with the LM score and resorts)

Disadvantages

1. s. . . l. . . o . . . w
2. language model not integrated (needs strict structure)
3. strictness $\rightarrow$ coverage problems

Exact Decoder

Toy?

1. competed in the WMT 2014 challenge
   Training data: 4 million sentence pairs
2. long preparation stage
3. but decoding only for 1 week
   Decode data: 10,000 sentences

Exact Decoder

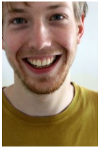
Evaluation

System	BLEU
Baseline (tree-to-tree)	13.07
Baseline (string-to-tree)	14.67
Baseline (phrase-based)	17.51
ExactMBOT	16.23

Larger BLEU-score is better

Open Problems

Directions



- language model integration
- decode heuristics (A^* search, etc.)

- more general rule extraction
- tree-to-string and string-to-tree setting