Extended HVS parser

Filip Jurčíček

UWB - University of West Bohemia, Center of Applied Cybernetics
Pilsen, 306 14, Czech Republic

filip@kky.zcu.cz

Jun 2, 2008

Outline

Introduction

- Semantic analysis of spoken dialogues
- Statistical semantic parsing
- The Hidden Vector State parser

The baseline

Negative examples

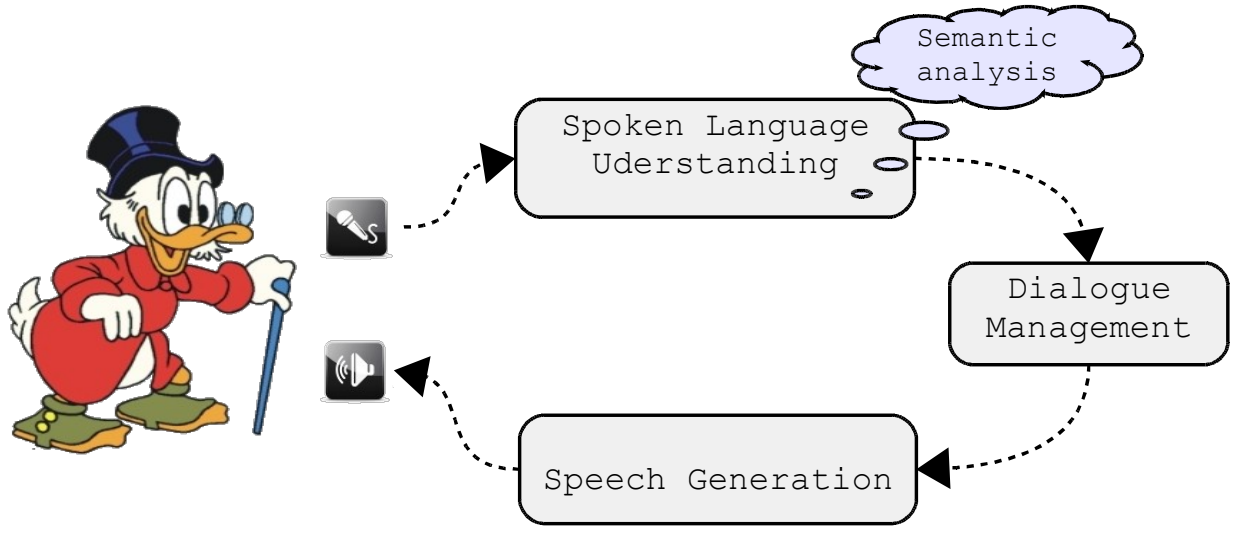
Left-right-branching parsing

Semantic parser input parametrization

HVS parser & GMTK

Conclusion

Dialogue systems



Spoken language understanding

- continuous spontaneous spoken speech,
- spoken speech is often ungrammatical, includes disfluences,
- the transformation into text form contains errors.

Systems using ATIS data

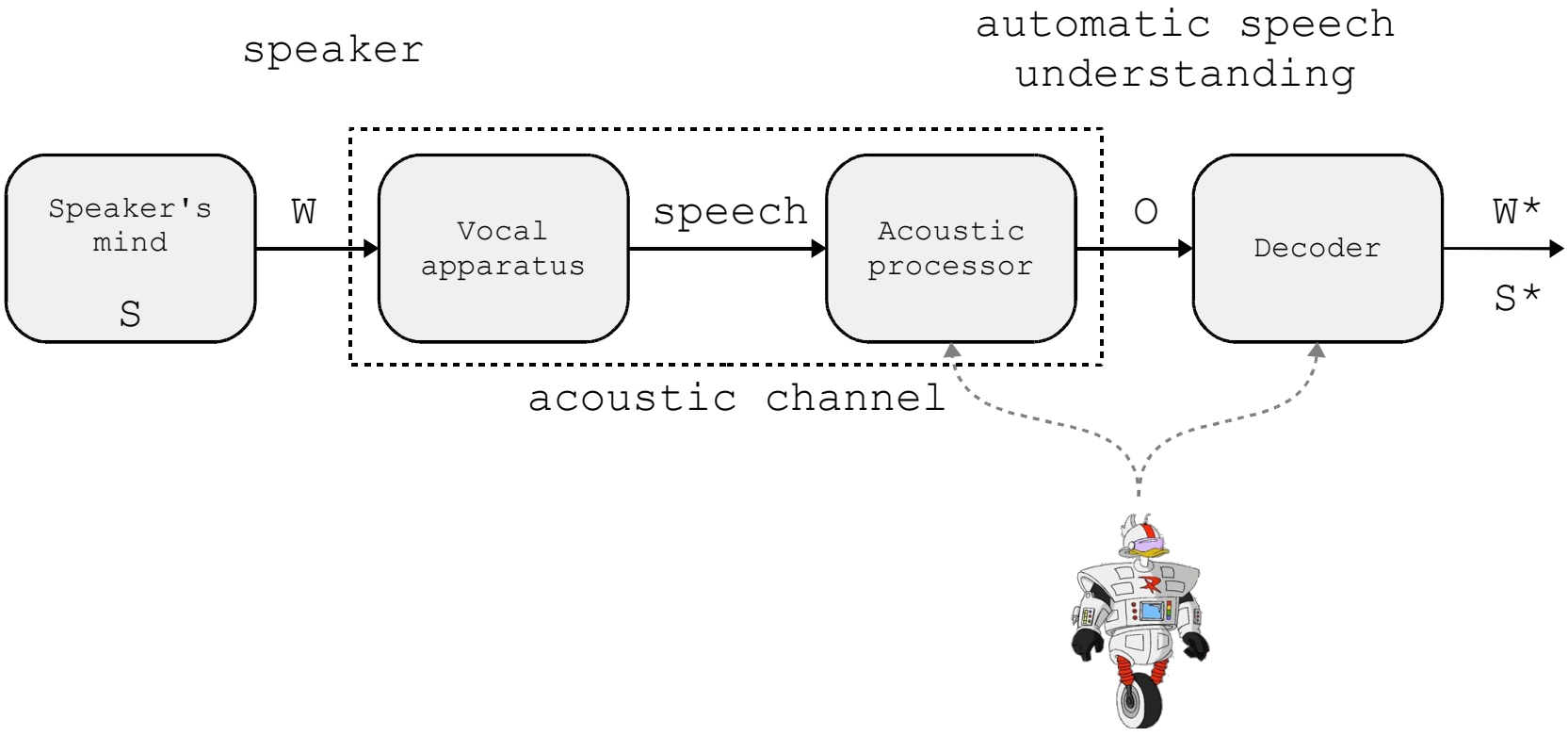
Based on context free grammars

- TINA from Massachusetts Institute of Technology
- PHOENIX from Carnegie Mellon University
- GEMINI from SRI International

Based on statistical models

- CHRONUS (Markov models) from AT&T
- Hidden Understanding Model (HUM) from BBN Technologies
- Hidden Vector State (HVS) parser from The University of Cambridge

The noisy channel



Concept sequence decoding

Concept sequence decoding

$$S^* = \underset{S}{\text{argmax}}\, P(W|S)P(S)$$

- $P(S)$ is the semantic model,
- $P(W|S)$ is the lexical model.

Flat-concept parser

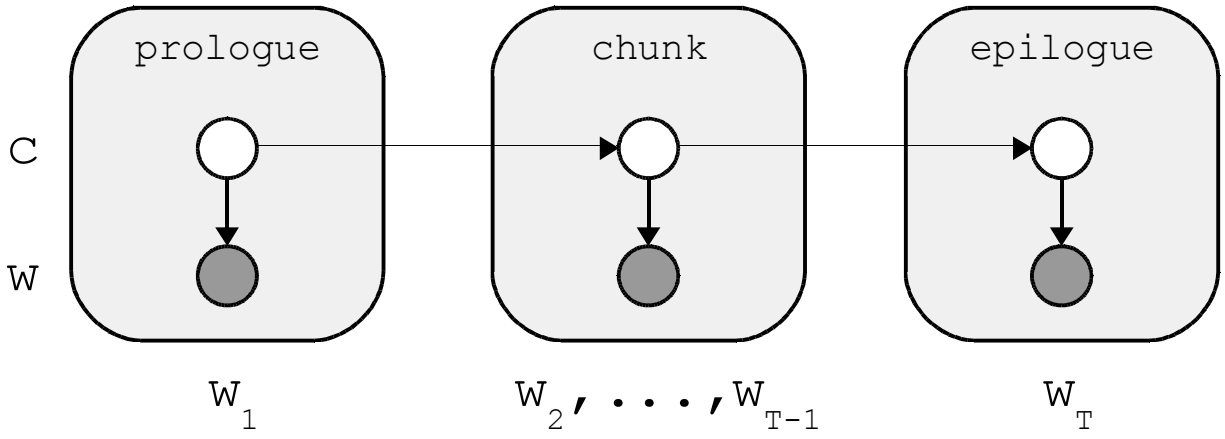


Figure: The graphical model of the FC model.

Flat-concept parser

Implementation

The semantic model is

$$P(S) = \prod_{t=1}^{T} P(c_t|c_{t-1})$$

The lexical model is

$$P(W|S) = \prod_{t=1}^{T} P(w_t|c_t)$$

The Hidden Vector State parser

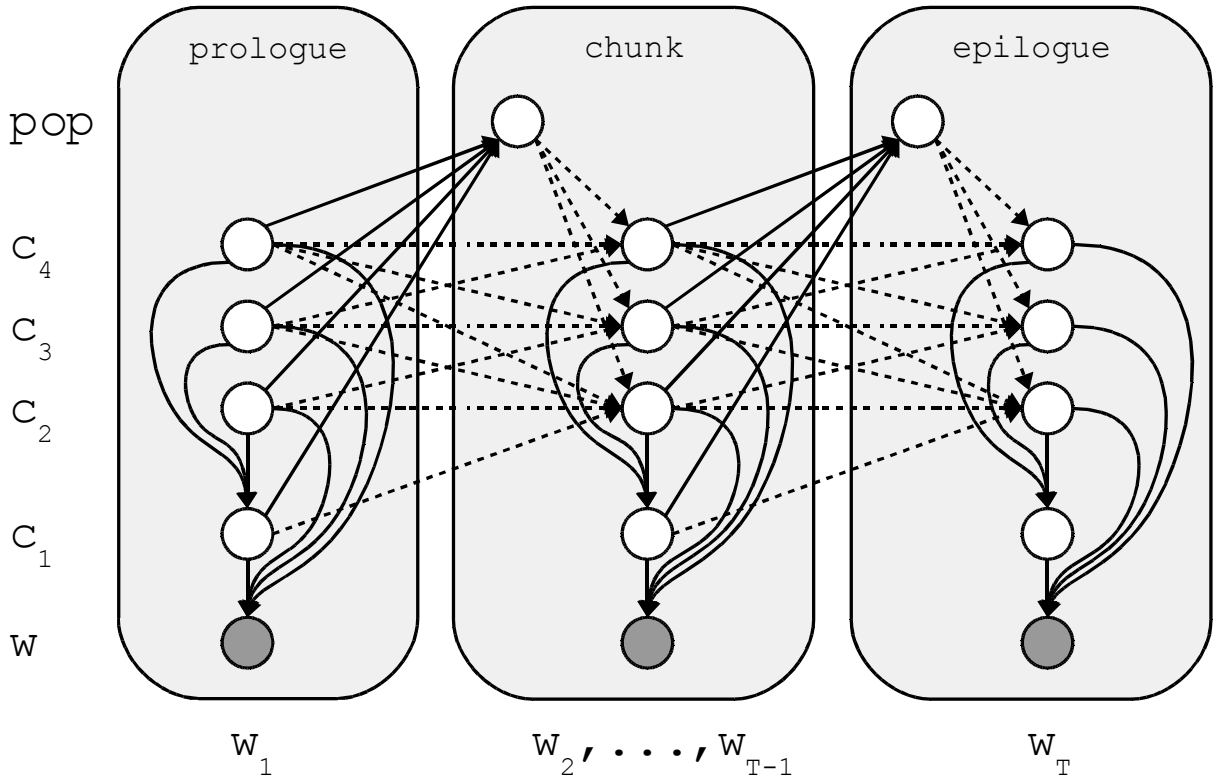


Figure: The graphical model of the HVS parser.

The Hidden Vector State parser

Implementation

The semantic model is

$$P(S) = \prod_{t=1}^{T} P(pop_t | c_{t-1}[1,4]) P(c_t[1] | c_t[2,4])$$

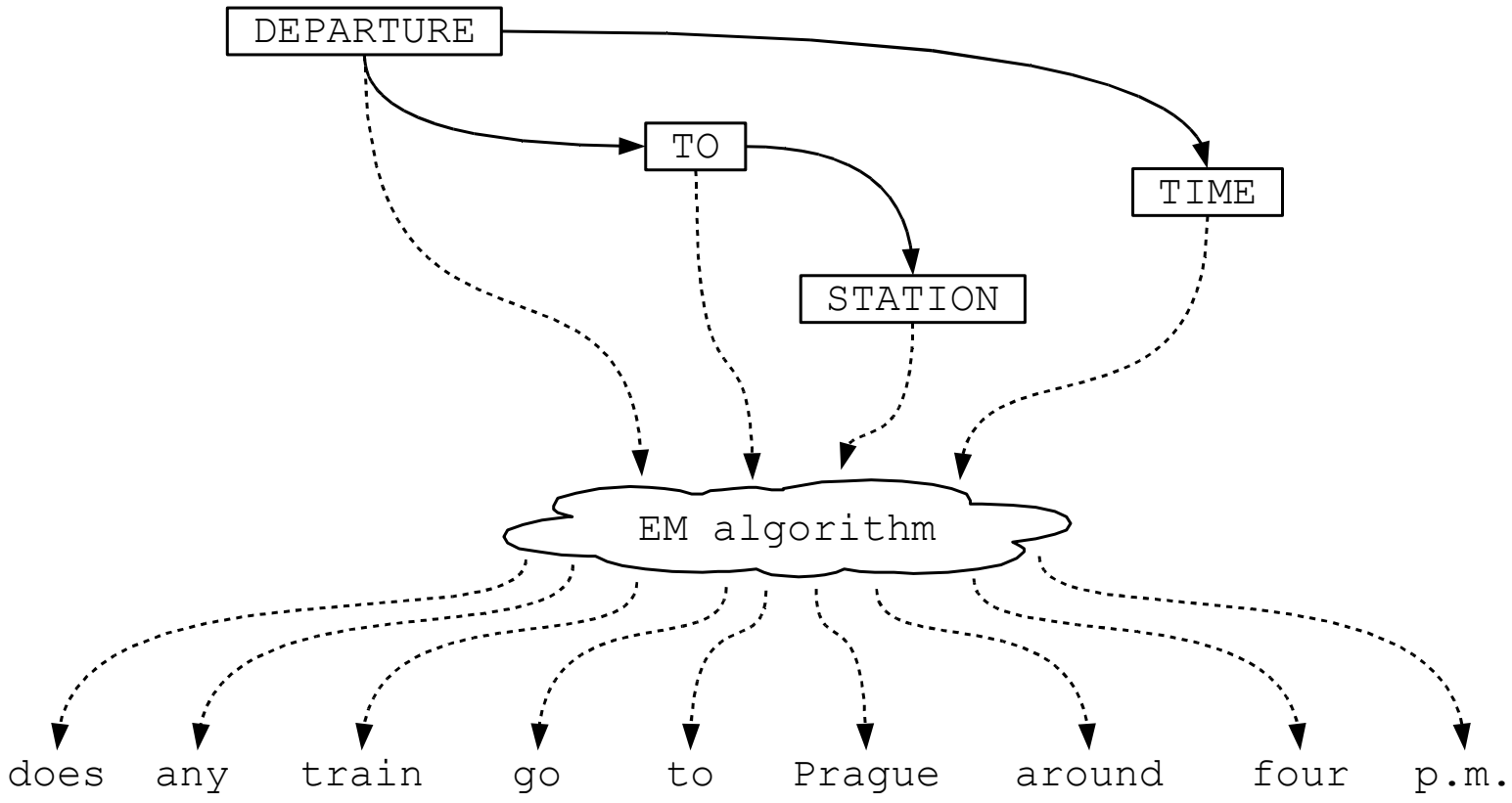
The lexical model is

$$P(W|S) = \prod_{t=1}^{T} P(w_t | c_t[1,4])$$

HVS parser - training data

Training data

does any train go to Prague around four p.m.
DEPARTURE(TO(STATION), TIME)



HVS parser - output of decoding

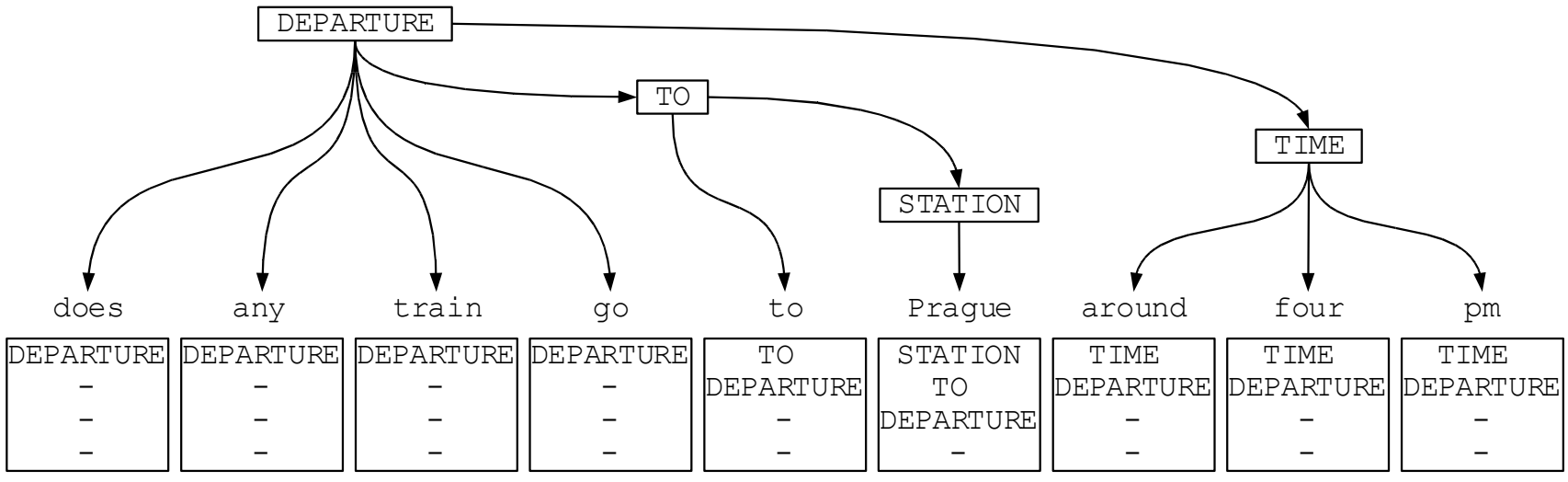


Figure: A full semantic parse tree with the corresponding stack sequence.

Outline

Introduction

The baseline
- Semantic data
- Difference in data
- Performance of the HVS parser

Negative examples

Left-right-branching parsing

Semantic parser input parametrization

HVS parser & GMTK

Conclusion

Semantic and dialogue act example dialogue

Semantics	Utterance (Literal English translation)
GREETING	the information please
GREETING	hello
DEPARTURE(TIME, TRAIN_TYPE, TO(STATION))	I have a question how can I go today by regional train to Starýho Plzence
OTHER_INFO	well we do not have many connections here
TIME, TIME	now one goes at eight sixteen if you catch it after that only at eleven ten
TIME	at eleven ten
ACCEPT(TIME, FROM(STATION))	it is not so bad at eleven ten from Hlavního yeah
TRAIN_TYPE	is it possible to take a stroller with me
ACCEPT	of course madam be sure that you can
TRAIN_TYPE	so there are the new cars

Size, cardinality and reliability

Size of the corpus

- 1,109 dialogues
- 17,900 utterances
- 2,872 unique words

Cardinality of dimensions

- SPEECH-ACT: 15 speech-acts.
- SEMANTIC: 35 concepts.

Reliability of SEMANTIC dimension

- stability: about 0.90
- reproducibility: about 0.81

Difference between CU-ATIS and UWB-HHTT semantic data

	CU-ATIS data	UWB-HHTT data
order of semantic concepts	not defined	defined
lexical classes	defined	not defined
performance evaluation	based on extracted slot values	based only on semantics

Error criteria

Semantic Accuracy

The reference and the hypothesis annotations are considered equal only if they exactly match each other.

Example

REF: ARRIVAL(TIME, FROM(STATION))

HYP1: ARRIVAL(TIME, TO(STATION))
HYP2: DEPARTURE(TRAIN_TYPE)

Error criteria

Concept Accuracy

The concept accuracy measures the minimum number of substitutions, deletions, and insertions required to transform one tree into another.

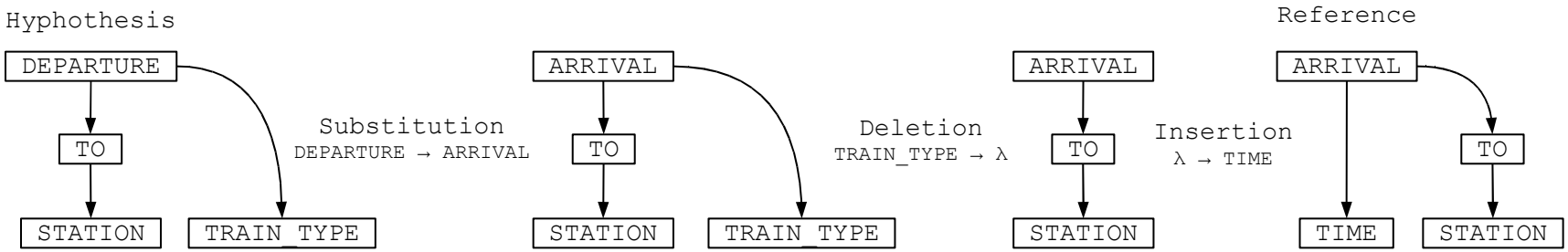


Figure: The tree edit distance operations' sequence.

The lower-bound baseline and the upper-bound ceiling

The lower-bound baseline

- SPEECH-ACT: 26%
- SEMANTIC: 17% SAcc, 26% CAcc

The upper-bound ceiling

- SPEECH-ACT: 95%
- SEMANTIC: 83% SAcc, 91% CAcc

Performance

	Test data		Development data	
	SAcc	CAcc	SAcc	CAcc
the lower-bound baseline	17.3	26.0	-	-
the tuned HVS parser	47.9	63.2	50.7	64.3
the upper-bound ceiling	85.0	91.0	-	-

Table: Performance of the tuned HVS parser.

Outline

Introduction

The baseline

Negative examples
- Positive examples
- Negative examples
- Extraction of negative examples
- Application of negative examples
- Performance

Left-right-branching parsing

Semantic parser input parametrization

HVS parser & GMTK

Positive examples

Positive examples in the corpus

Utterance	Semantics
does any train go to Prague around four p.m.	DEPARTURE(FROM(STATION), TIME)
what is the price of a ticket to Cambridge around six a.m.	PRICE(TO(STATION), TIME)
if you want a direct train than it departures around nine p.m.	DEPARTURE(TRAIN_TYPE, TIME)
. . .	. . .

Positive example not labeled in the corpus (word level)

Word (words)	Concept (Semantics)
Prague	STATION
around four p.m.	TIME
Cambridge	STATION
around six a.m.	TIME
direct	TRAIN_TYPE
around nine p.m.	TIME
. . .	. . .

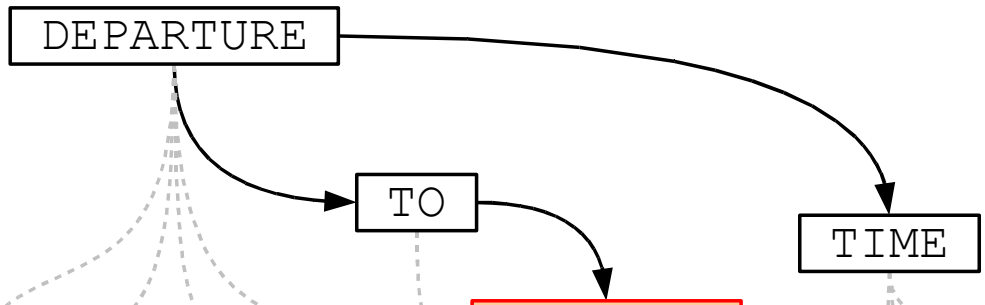
Negative examples

Negative examples not labeled in the corpus (word level)

Word (words)	Concept (Semantics)
if you want a direct train than it departures around nine p.m.	STATION
does any train go to Prague around four p.m.	TRAIN_TYPE
what is the price of a ticket to Cambridge around six a.m.	TRAIN_TYPE
. . .	. . .

The corpus

Utterance	Semantics
does any train go to Prague around four p.m.	DEPARTURE(FROM(STATION), TIME)
what is the price of a ticket to Cambridge around six a.m.	PRICE(TO(STATION), TIME)
if you want a direct train than it departures around nine p.m.	DEPARTURE(TRAIN_TYPE, TIME)
. . .	. . .



Extraction of negative examples

Used concepts

AMOUNT, LENGTH, NUMBER, STATION, TIME and TRAIN_TYPE

Potential problems

- the lexical realization of the concepts AMOUNT, LENGTH, NUMBER is similar
- incorrectly annotated utterance

Utterance	Semantics
does any train go to Prague around four p.m.	DEPARTURE(TRAIN_TYPE, TIME)

- too general concept in the annotation

Word (words)	Concept (Semantics)
weather is pleasant today in Prague	OTHER_INFO
. . .	. . .

Selection of proper utterances

Dominating concepts

ACCEPT, ARRIVAL, DELAY, DEPARTURE, DISTANCE, DURATION, PLATFORM, PRICE, and REJECT

Example

Utterance	Semantics
what is the price of a ticket to Cambridge	PRICE(TO(STATION))
it will arrive at the second platform	ARRIVAL(PLATFORM(NUMBER))
...	...

Application of negative examples

Modification of the lexical model

$$x(w, c[1,4]) = \begin{cases} \epsilon & \text{if } (w, c[1]) \text{ is a negative example,} \\ 1/|V| & \text{otherwise} \end{cases}$$

$$P(w|c[1,4]) = \frac{x(w, c[1,4])}{\sum_{\overline{w} \in V} x(\overline{w}, c[1,4])} \quad \forall c[1,4] \quad \forall w \in V$$

where ϵ is reasonably small positive value and V is a word lexicon.

Performance

	Test data			Development data		
	SAcc	CAcc	p-value	SAcc	CAcc	p-value
baseline	47.9	63.2		50.7	64.3	
with neg. examples	50.4	64.9	< 0.01	52.8	67.0	< 0.01

Table: Performance comparison of the HVS parser using negative examples.

Outline

Introduction

The baseline

Negative examples

Left-right-branching parsing
- Motivation
- The HVS parser with probabilistic pushing
- The left-right-branching HVS
- Performance

Semantic parser input parametrization

HVS parser & GMTK

The left-right-branching HVS parser

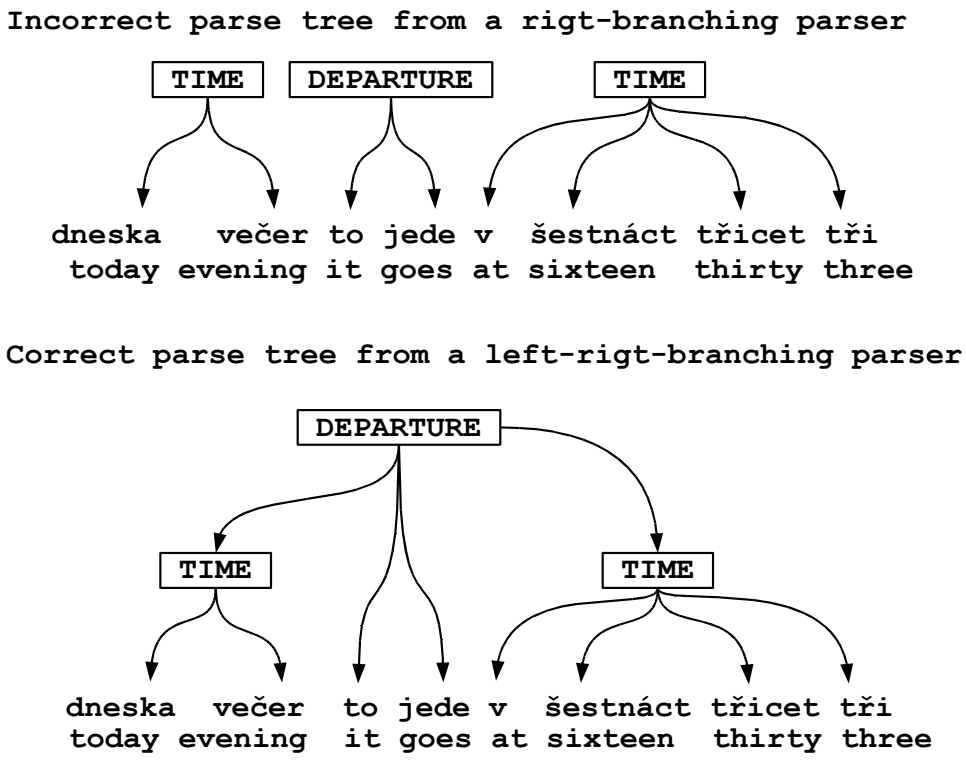


Figure: An incorrect parse tree from a right-branching parser and a correct parse tree from a left-right-branching parser.

Recapitulation - The Hidden Vector State parser

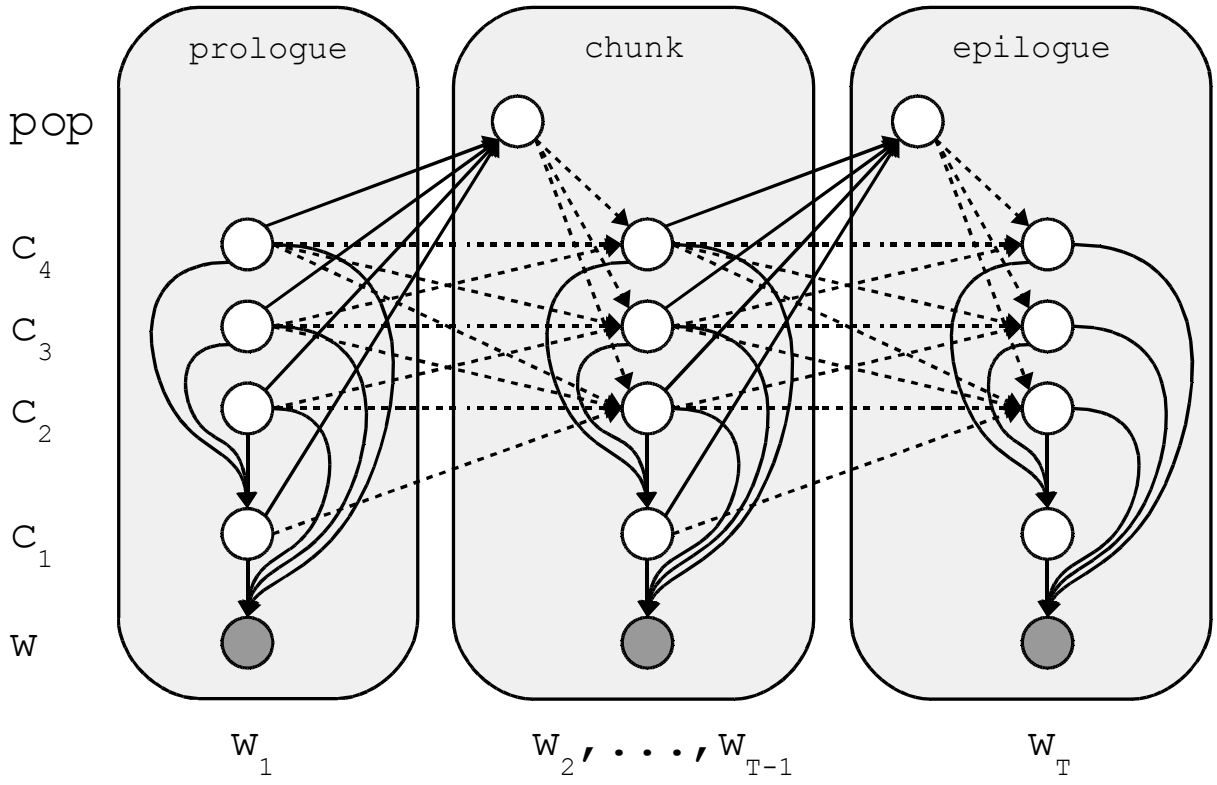


Figure: The graphical model of the HVS parser.

The HVS parser with probabilistic pushing

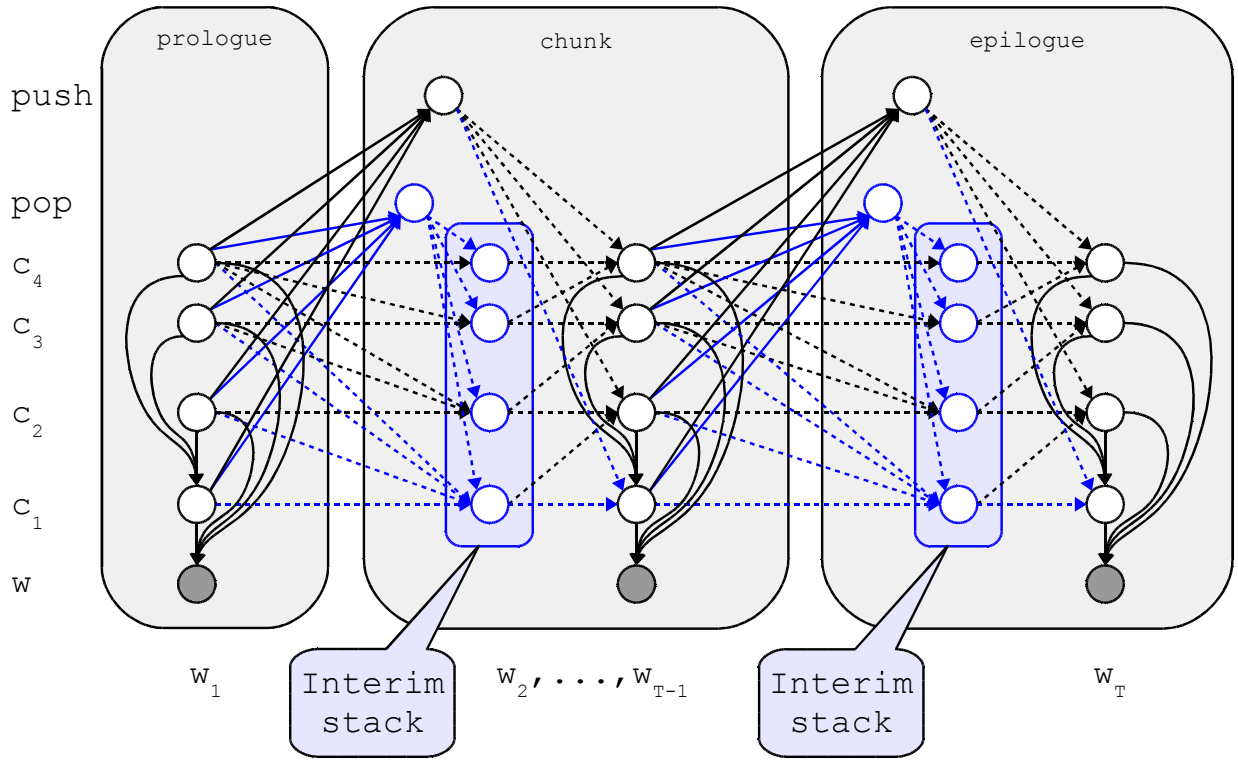


Figure: The graphical model of the HVS parser with probabilistic pushing (HVS-PP).

The left-right-branching HVS

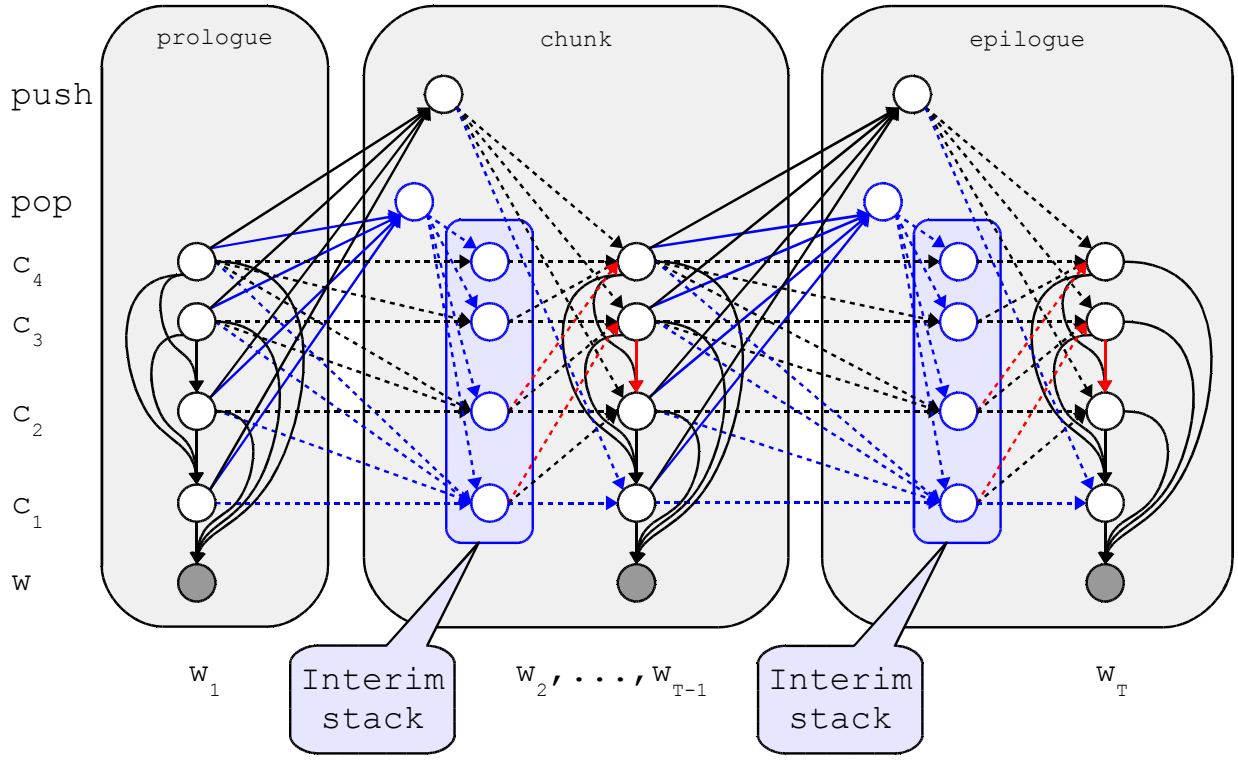


Figure: The graphical model of the right-branching HVS parser limited to insert at most two new concepts (LRB-HVS).

Mathematical representation

The HVS parser with probabilistic pushing

$$P(S) = \prod_{t=1}^{T} P(pop_t | c_{t-1}[1, \ldots 4]) P(push_t | c_{t-1}[1, \ldots 4]) \cdot \begin{cases} 1 & \text{if } push_t = 0 \\ P(c_t[1] | c_t[2, \ldots 4]) & \text{if } push_t = 1 \end{cases}$$

The left-right-branching HVS

$$P(S) = \prod_{t=1}^{T} P(pop_t | c_{t-1}[1, \ldots 4]) P(push_t | c_{t-1}[1, \ldots 4]) \cdot \cdot \begin{cases} 1 & \text{if } push_t = 0 \\ P(c_t[1] | c_t[2, \ldots 4]) & \text{if } push_t = 1 \\ P(c_t[1] | c_t[2, \ldots 4]) P(c_t[2] | c_t[3, 4]) & \text{if } push_t = 2 \end{cases}$$

Performance

The baseline is the parser using negative examples.

	Test data			Development data		
	SAcc	CAcc	p-value	SAcc	CAcc	p-value
baseline	50.4	64.9		52.8	67.0	
HVS-PP	54.1	67.2	< 0.01	56.6	68.4	< 0.01
LRB-HVS	58.3	69.3	< 0.01	60.1	70.6	< 0.01

Table: The performance of the HVS-PP and LRB-HVS parsers. The parsers were evaluated on the test and the development data.

Outline

Introduction

The baseline

Negative examples

Left-right-branching parsing

Semantic parser input parametrization
- Input feature vector
- Performance

HVS parser & GMTK

Conclusion

Input feature vector

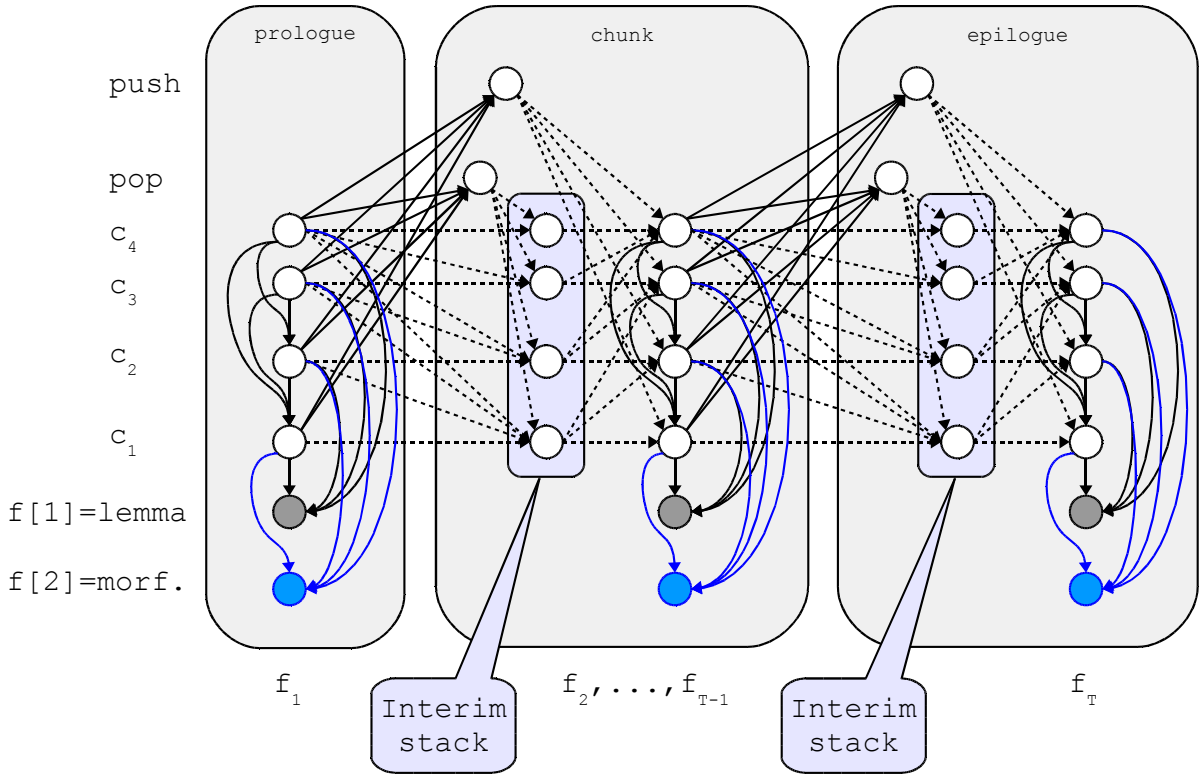


Figure: The graphical model of the HVS parser with the input feature vector composed of a lemma and a morphological tag (HVS-IFV).

Mathematical representation

Input feature vector

$$
\begin{aligned}
P(F|S) &= \prod_{t=1}^{T} P(f_t \mid c_t) \\
&= \prod_{t=1}^{T} P(f_t[1], f_t[2], \ldots f_t[N] \mid c_t) \\
&\approx \prod_{t=1}^{T} \prod_{i=1}^{N} P(f_t[i] \mid c_t)
\end{aligned}
$$

Scaling of the semantic model

$$
S^* = \arg\max_S P(F|S) P^{\lambda}(S)
$$

Performance

The baseline is the LRB-HVS parser.

	Test data			Development data		
input feature vector	SAcc	CAcc	p-value	SAcc	CAcc	p-value
baseline (Words)	58.3	69.3		60.1	70.6	
Lemmas	58.4	69.8		60.5	71.3	
Lemmas+Morphological tags	63.1	73.8	< 0.01	65.4	75.7	< 0.01

Table: The performance of the HVS parser using the input feature vector in the lexical model. The parsers were evaluated on the test and the development data.

Outline

Introduction

The baseline

Negative examples

Left-right-branching parsing

Semantic parser input parametrization

HVS parser & GMTK
- GMTK

Conclusion

GMTK

The Graphical Models Toolkit (GMTK)

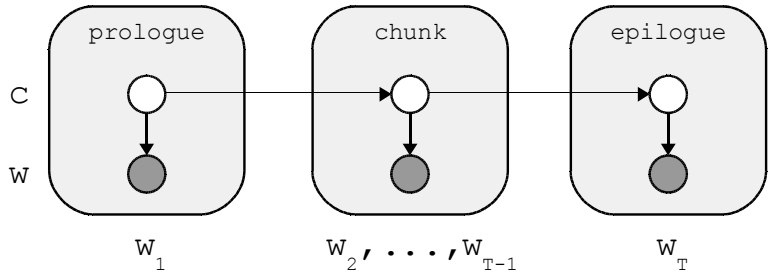
- Jeff Bilmes, University of Washington, Dept. of EE
- Geoffrey Zweig, Microsoft, Speech Research Group
- `http://ssli.ee.washington.edu/~bilmes/gmtk/`
- `http://ssli.ee.washington.edu/~bilmes/gmtk/doc.pdf`

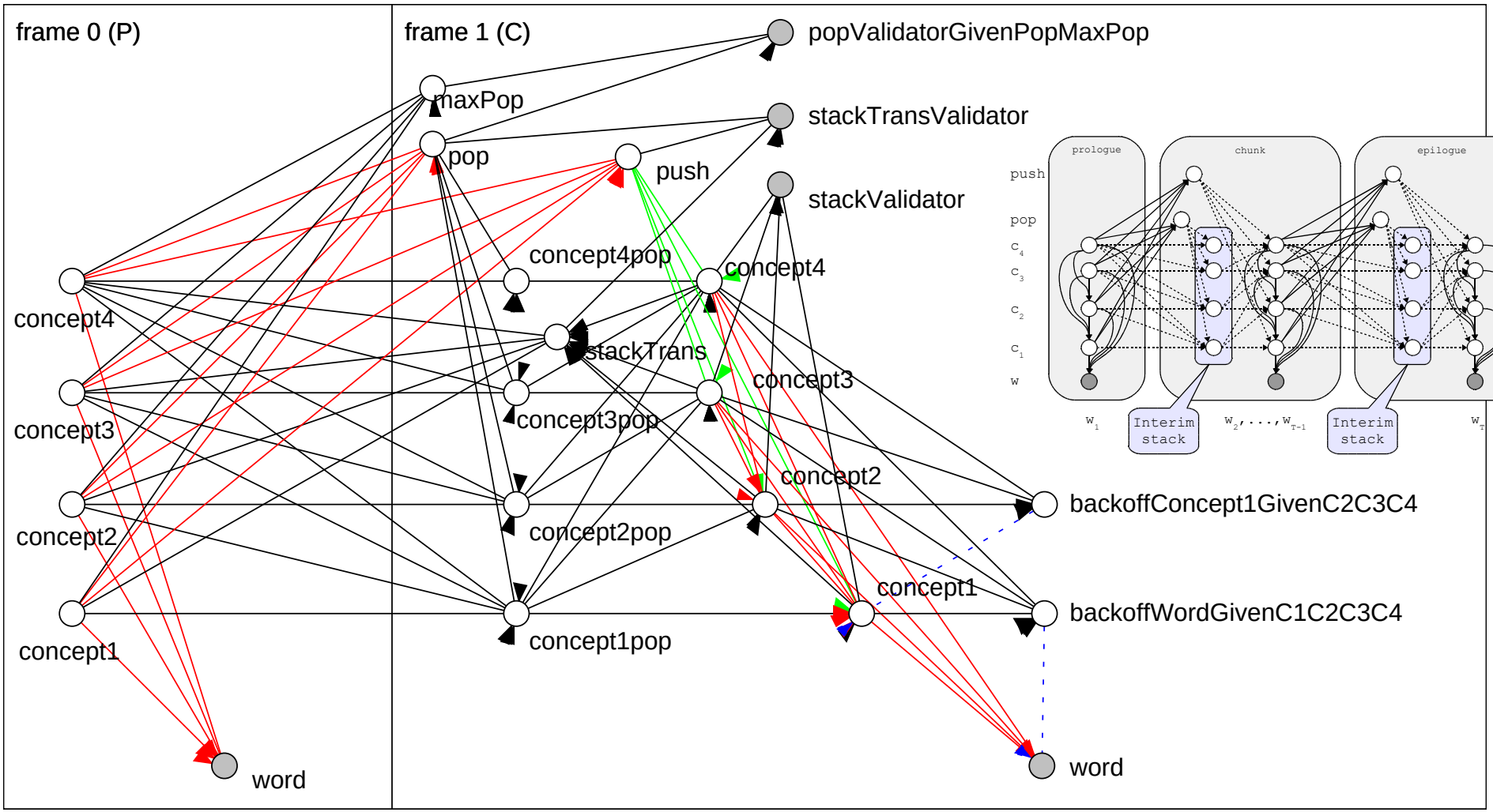
Main tools used from GMTK

- Preparing input files: gmtkTriangulate, gmtkDTindex
- Training: gmtkEMtrain(New)
- Decoding: gmtkViterbi(New)

HMM Tagger in GMTK

```
#include "commonParams"

frame : 0 {
  variable : concept {
    type: discrete hidden cardinality CONCEPT_CARD ;
    switchingparents : nil;
    conditionalparents: nil using DeterministicCPT("conceptZero");
  }

  variable : word  {
    type: discrete observed 0:0 cardinality WORD_CARD ;
    switchingparents : nil;
    conditionalparents: concept(0) using SparseCPT("wordGivenC1");
  }
}

frame : 1 {
  variable : concept {
    type: discrete hidden cardinality CONCEPT_CARD;
    switchingparents : nil;
    conditionalparents: concept(-1) using DenseCPT("conceptGivenC_1");
  }

  variable : word  {
    type: discrete observed 0:0 cardinality WORD_CARD;
    conditionalparents: concept(0) using DenseCPT("wordGivenC1")
  }
}

chunk 1:1
```




frame 0 (P)
frame 1 (C)
popValidatorGivenPopMaxPop
maxPop
stackTransValidator
pop
push
stackValidator
concept4pop
concept4
concept4
stackTrans
concept3
concept3
concept3pop
concept2
concept2
concept2pop
backoffConcept1GivenC2C3C4
concept1
backoffWordGivenC1C2C3C4
concept1
concept1pop
word
word
prologue
chunk
epilogue
push
pop
c4
c3
c2
c1
w
w1
Interim stack
w2,...,wT-1
Interim stack
wT

Backoff word generation

```
variable : backoffWordGivenC1C2C3C4 {
    type: discrete hidden cardinality BACKOFF_C1C2C3C4_CARD;
    switchingparents: nil;
    conditionalparents:
      concept1(0),concept2(0),concept3(0),concept4(0) using DeterministicCPT("backoffC1C2C3C4");
  }

variable : word  {
    type: discrete observed 0:0 cardinality WORD_CARD;
    switchingparents: backoffWordGivenC1C2C3C4(0) using mapping("copy");
    conditionalparents:
        concept1(0),concept2(0),concept3(0),concept4(0) using SparseCPT("wordGivenC1C2C3C4")
      | concept1(0),concept2(0),concept3(0)             using SparseCPT("wordGivenC1C2C3")
      | concept1(0),concept2(0)                         using SparseCPT("wordGivenC1C2")
      | concept1(0)                                     using  DenseCPT("wordGivenC1")
      | nil                                             using  DenseCPT("wordUnigram");
}
```

Stack transition validator

```
1 % a DT that evaluates to one when there is "transition"
stackTransGivenC1C1_1C2C2_1C3C3_1C4C4_1
8 % number of parents
-1  {(!((p0==p1) && (p2==p3) && (p4==p5) && (p6==p7)))}

2 % a DT that validate pop, push, and stack transition
%   if there is no transition, the pop and push RV should be equal to 0
stackTransValidatorGivenStPopPush
3 % number of parents
0 2 0 default
  -1  {(p1==0)&&(p2==0)}  % the pop and push should be equal to 0
  -1  {p1||p2}    % the pop or push should be different to 0

-------------------------------------------------------------------------------

variable : stackTrans {
    type: discrete hidden cardinality STACK_TRANS_CARD;
    switchingparents: nil;
    conditionalparents: concept1(0),concept1(-1),concept2(0),concept2(-1),concept3(0),concept3(-1),concept4(0),concept4(-1)
      using DeterministicCPT("stackTransGivenC1C1_1C2C2_1C3C3_1C4C4_1");
}

% make sure that in case no change on the stack, it will use pop==push==0
variable : stackTransValidator {
    type: discrete observed value 1 cardinality 2;
    switchingparents: nil;
    conditionalparents: stackTrans(0), pop(0), push(0)
      using DeterministicCPT("stackTransValidatorGivenStPopPush");
}
```

Outline

Introduction

The baseline

Negative examples

Left-right-branching parsing

Semantic parser input parametrization

HVS parser & GMTK

Conclusion

- Performance
- Questions?

Performance overview

	Test data			Development data		
	SAcc	CAcc	p-value	SAcc	CAcc	p-value
the lower-bound baseline	17.3	26.0				
baseline HVS parser	47.9	63.2		50.7	64.3	
neg. examples	50.4	64.9	< 0.01	52.8	67.0	< 0.01
LRB-HVS	58.3	69.3	< 0.01	60.1	70.6	< 0.01
HVS-IFV	63.1	73.8	< 0.01	65.4	75.7	< 0.01
the upper-bound ceiling	85.0	91.0				

Table: The parsers were evaluated on the test and the development data.

Questions?

Thank you!

And also

Many thanks to my colleague Jan Švec from UWB for his work on the IFV part of the parser.