Concept-to-text Generation via Discriminative Reranking

Ioannis Konstas and Mirella Lapata

School of Informatics
Institute for Language, Cognition and Computation
University of Edinburgh

ACL 2012, Jeju Island

Introduction

Concept-to-text generation refers to the task of automatically producing textual output from nonlinguistic input (Reiter and Dale, 2000)

Introduction

Concept-to-text generation refers to the task of automatically producing textual output from nonlinguistic input (Reiter and Dale, 2000)

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

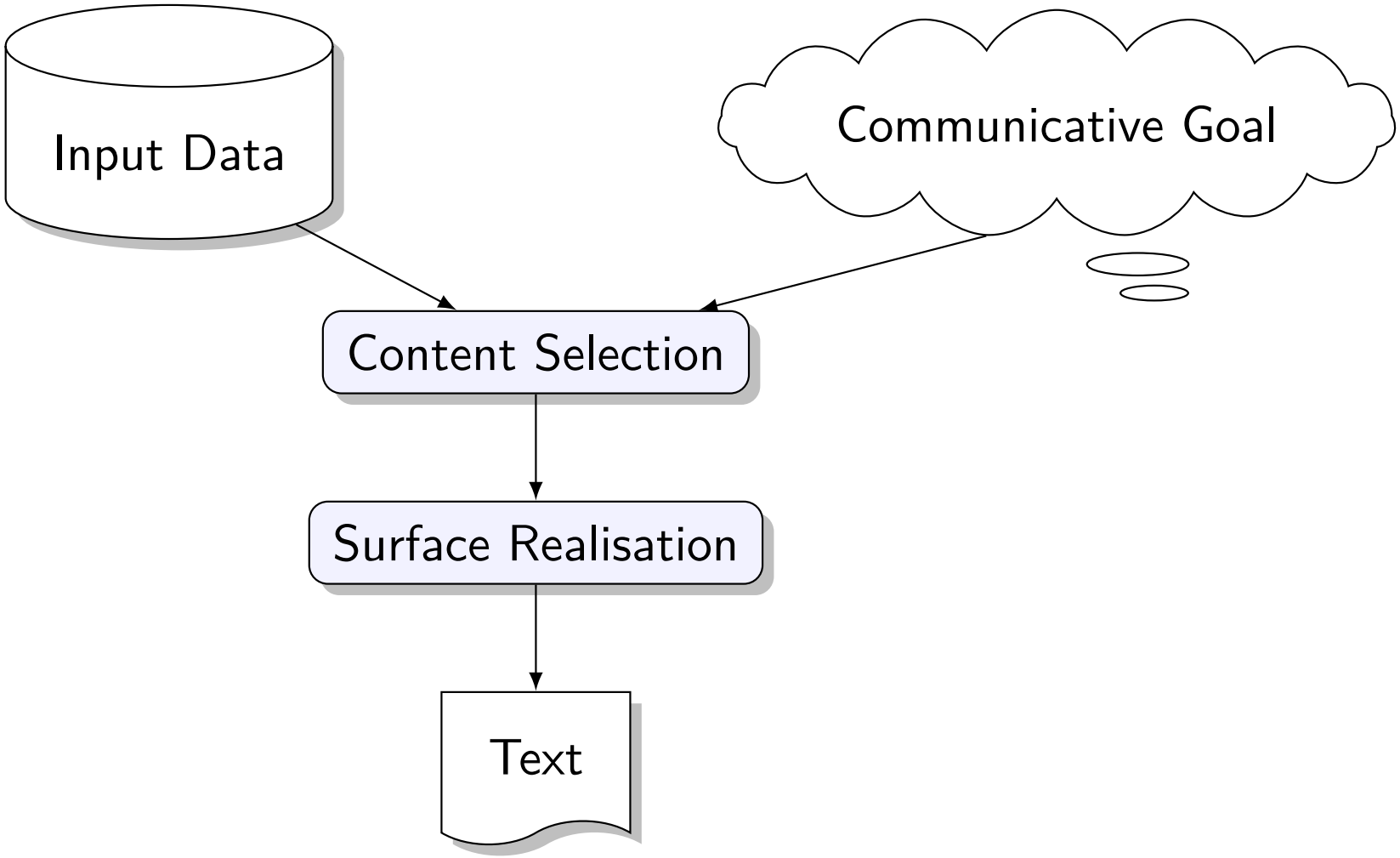
Month	
month	**dep/ar**
july	departure

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

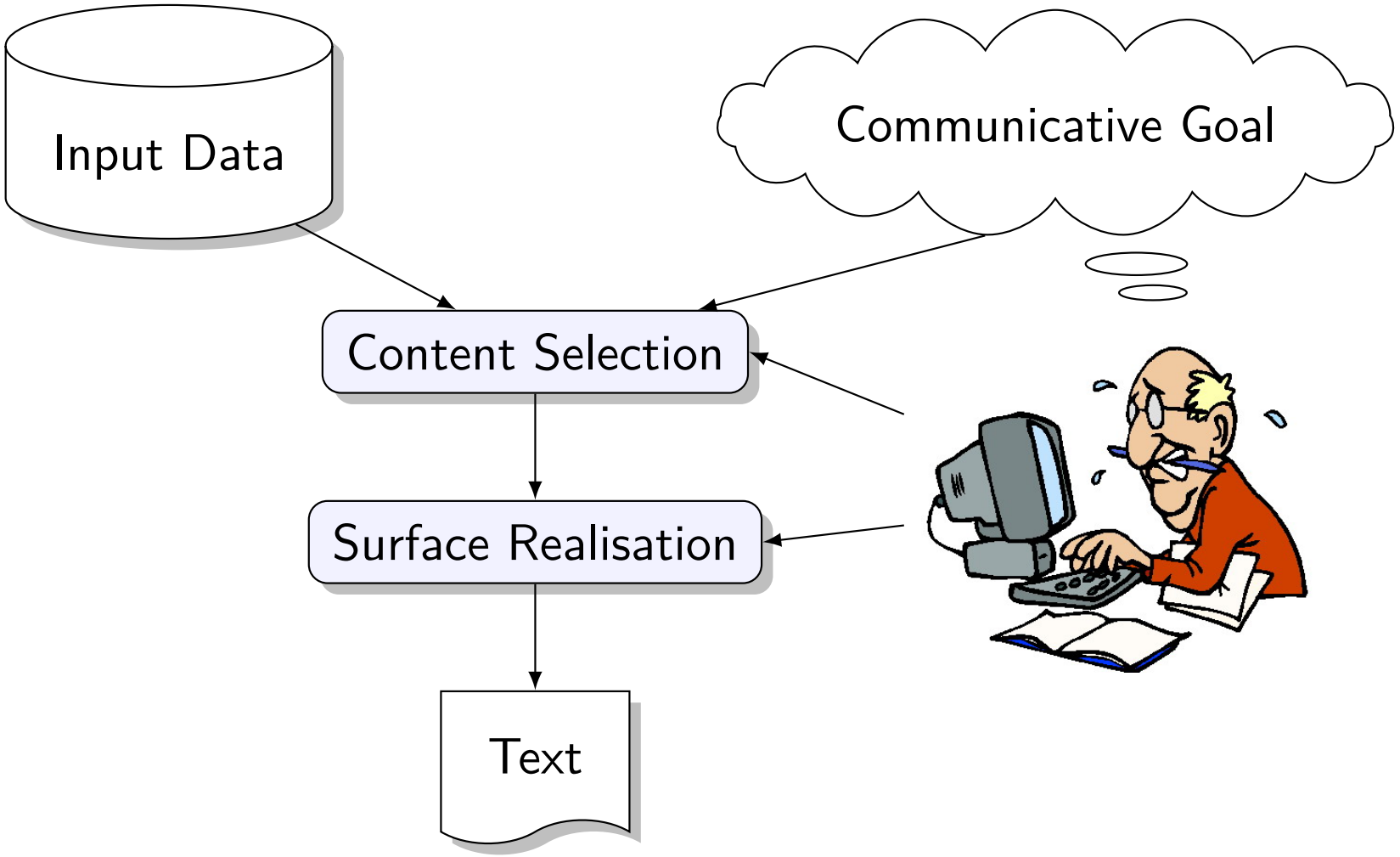
Search	
type	**what**
query	flight

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Traditional NLG Pipeline



Traditional NLG Pipeline



Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

Search	
type	**what**
query	flight

Our Approach

Flight	
from	**to**
edinburgh	jeju

Day Number	
number	**dep/ar**
7	departure

Month	
month	**dep/ar**
july	departure

Condition		
arg1	**arg2**	**type**
arrival_time	1600	$<$

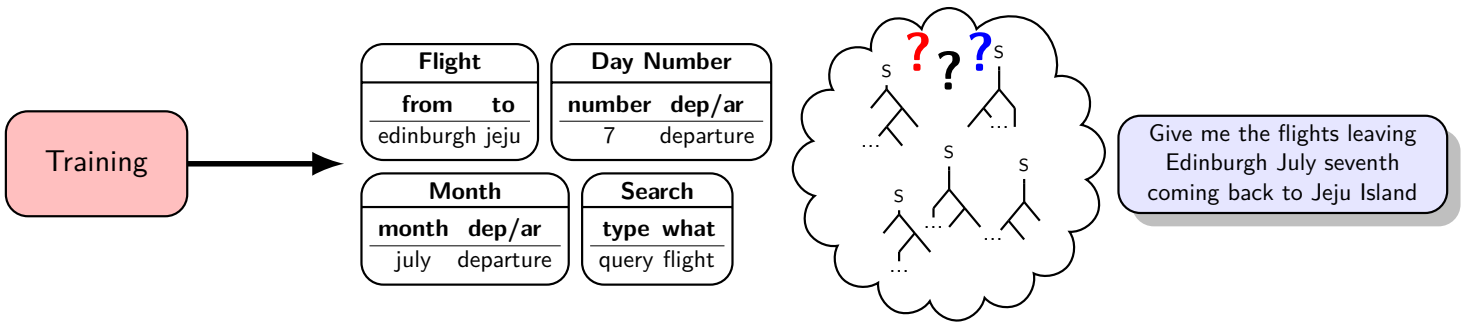
Search	
type	**what**
query	flight

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm

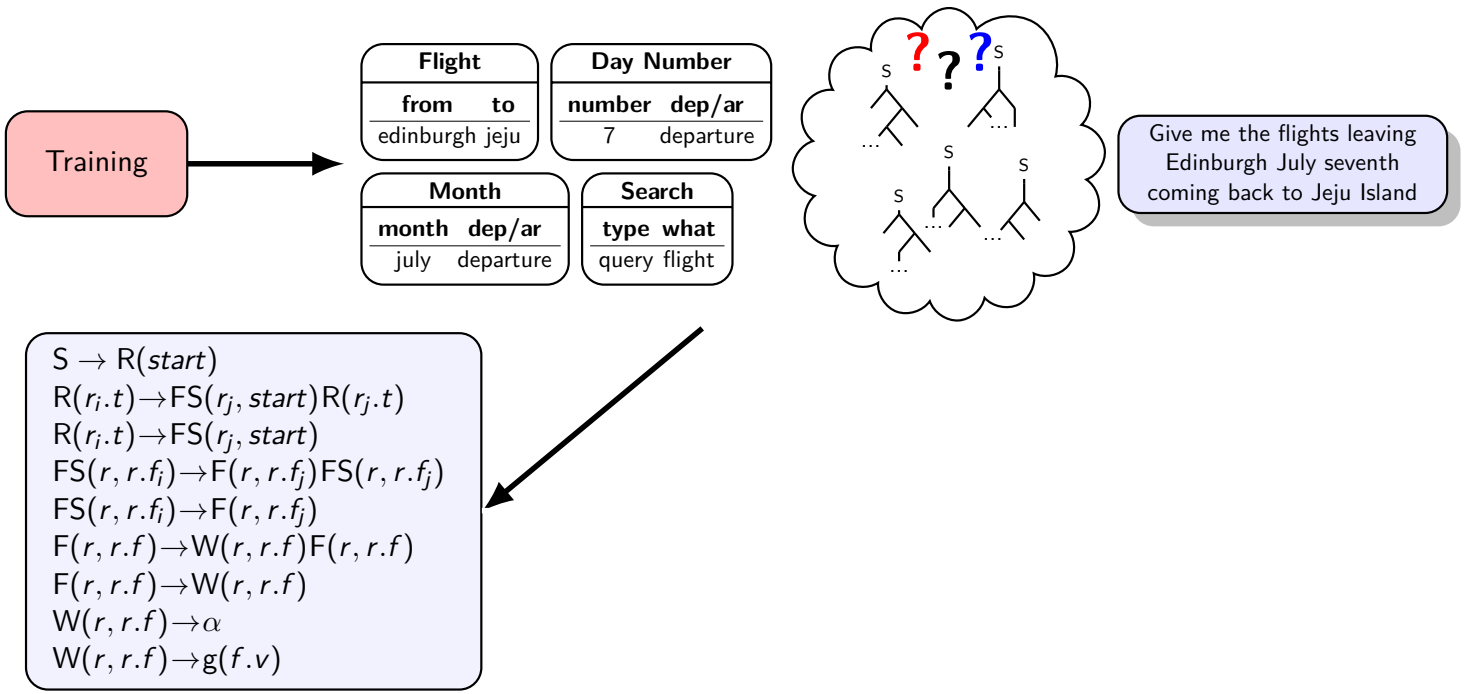
Please show me the flights from Edinburgh on July seventh to Jeju Island before 16:00

Joint Discriminative Reranking with Hypergraphs

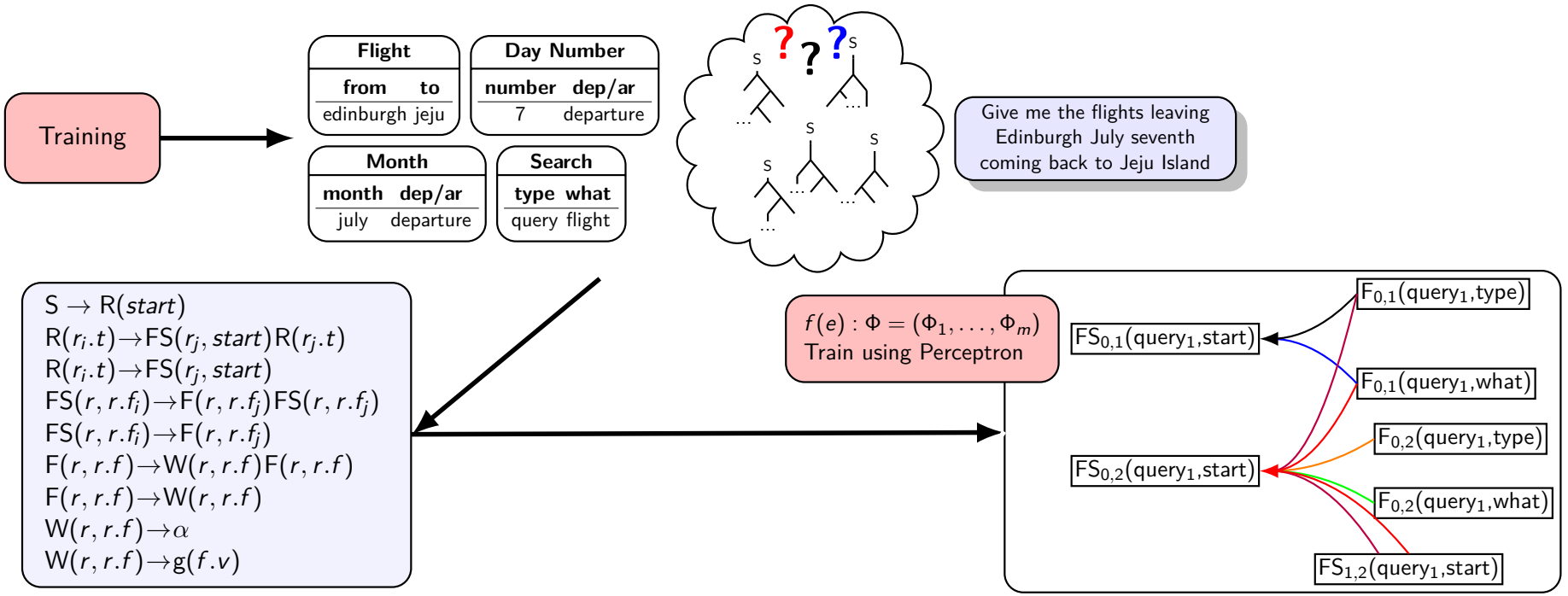
Joint Discriminative Reranking with Hypergraphs



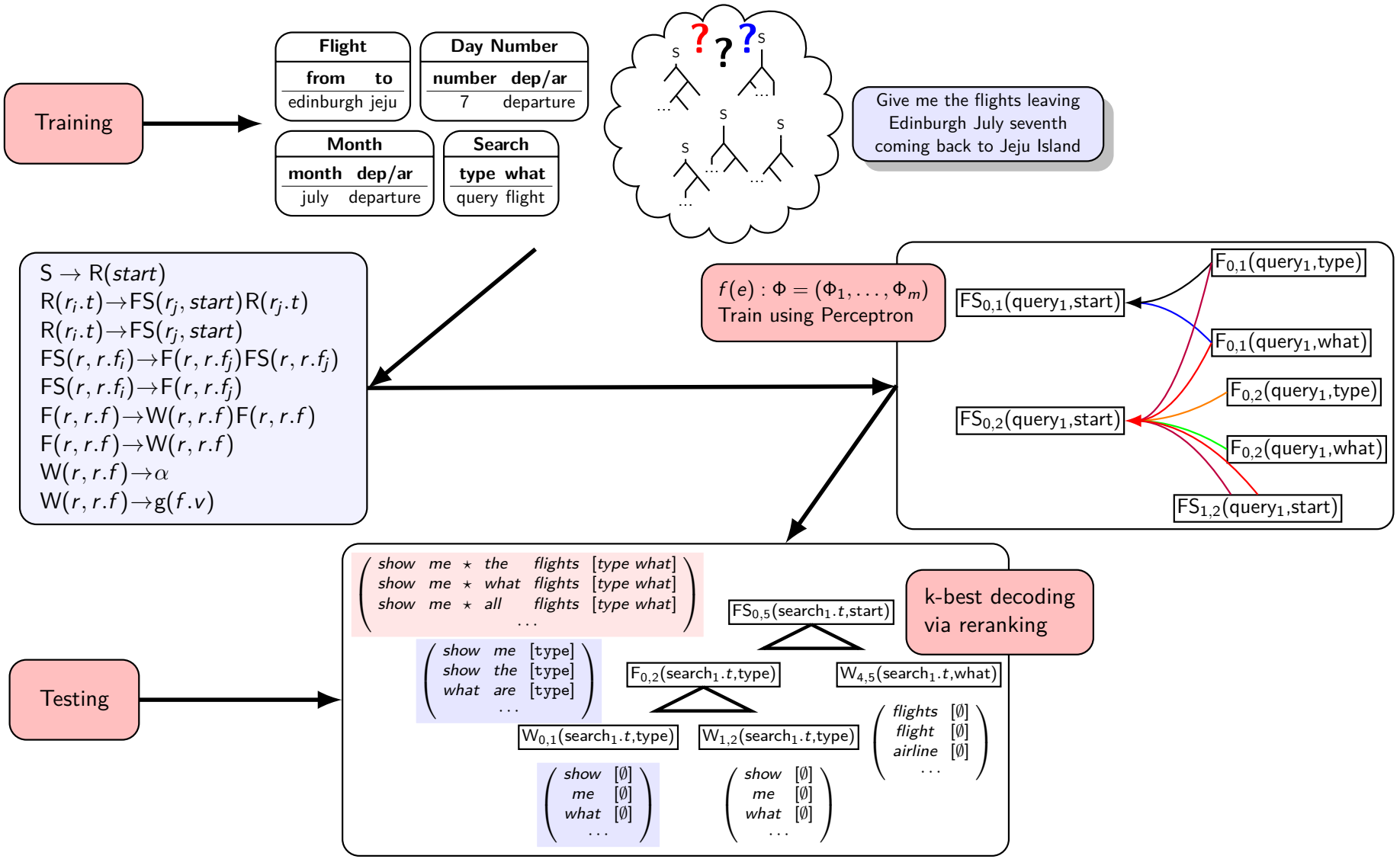
Joint Discriminative Reranking with Hypergraphs



Joint Discriminative Reranking with Hypergraphs



Joint Discriminative Reranking with Hypergraphs



Related Work

Angeli et al., 2010

- Unified content selection and surface realisation
- Obtain alignments from Liang et al. (2009)
- Sequence of discriminative (log-linear) local decisions (records - fields - templates)

Related Work

Angeli et al., 2010

- Unified content selection and surface realisation
- Obtain alignments from Liang et al. (2009)
- Sequence of discriminative (log-linear) local decisions (records - fields - templates)

Our approach

- **Joint** model allows for more global decisions
- **Hypergraphs** are a compact representation and allow for efficient inference (k-best decoding via cube pruning)
- **Discriminative reranking** reranks k-best trees at all internal nodes

Input

- Input: database records **d**
- Output: words **w** corresponding to some records of **d**
- Each record $r \in \mathbf{d}$ has a type $r.t$ and fields f
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

Day	
day of week	**dep/ar/ret**
sunday	departure
thursday	return

leaving on sunday

Input

- Input: database records **d**
- Output: words **w** corresponding to some records of **d**
- Each record $r \in \mathbf{d}$ has a type $r.t$ and fields f
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

Day

day of week	dep/ar/ret
sunday	departure
thursday	return

leaving on sunday

Input

- Input: database records **d**
- Output: words **w** corresponding to some records of **d**
- Each record $r \in \mathbf{d}$ has a type $r.t$ and fields f
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

Day	
day of week	**dep/ar/ret**
sunday	departure
thursday	return

leaving on sunday

Input

- Input: database records **d**
- Output: words **w** corresponding to some records of **d**
- Each record $r \in \mathbf{d}$ has a type $r.t$ and fields f
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

Day	
day of week	**dep/ar/ret**
sunday	departure
thursday	return

leaving on sunday

Input

- Input: database records **d**
- Output: words **w** corresponding to some records of **d**
- Each record $r \in \mathbf{d}$ has a type $r.t$ and fields f
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

Day	
day of week	**dep/ar/ret**
sunday	departure
thursday	return

leaving on sunday

Grammar Definition

1. $S \rightarrow R(start)$
2. $R(r_i.t) \rightarrow FS(r_j, start)R(r_j.t)$
3. $R(r_i.t) \rightarrow FS(r_j, start)$
4. $FS(r, r.f_i) \rightarrow F(r, r.f_j)FS(r, r.f_j)$
5. $FS(r, r.f_i) \rightarrow F(r, r.f_j)$
6. $F(r, r.f) \rightarrow W(r, r.f)F(r, r.f)$
7. $F(r, r.f) \rightarrow W(r, r.f)$
8. $W(r, r.f) \rightarrow \alpha$
9. $W(r, r.f) \rightarrow g(f.v)$

Grammar Definition

$\mathrm{R}(search_1.t) \rightarrow \mathrm{FS}(flight_1, start)\mathrm{R}(flight_1.t)$

1. $\mathrm{S} \rightarrow \mathrm{R}(start)$
2. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)\mathrm{R}(r_j.t)$
3. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)$
4. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)\mathrm{FS}(r, r.f_j)$
5. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)$
6. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)\mathrm{F}(r, r.f)$
7. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)$
8. $\mathrm{W}(r, r.f) \rightarrow \alpha$
9. $\mathrm{W}(r, r.f) \rightarrow \mathrm{g}(f.v)$

Grammar Definition

$\mathrm{FS}(\mathit{flight}_1, \mathit{from}) \rightarrow \mathrm{F}(\mathit{flight}_1, \mathit{to})\mathrm{FS}(\mathit{flight}_1, \mathit{to})$

1. $\mathrm{S} \rightarrow \mathrm{R}(\mathit{start})$
2. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\mathrm{R}(r_j.t)$
3. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})$
4. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)\mathrm{FS}(r, r.f_j)$
5. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)$
6. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)\mathrm{F}(r, r.f)$
7. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)$
8. $\mathrm{W}(r, r.f) \rightarrow \alpha$
9. $\mathrm{W}(r, r.f) \rightarrow \mathrm{g}(f.v)$

Grammar Definition

$$\mathrm{F}(search_1, what) \rightarrow \mathrm{W}(search_1, what)\mathrm{F}(search_1, what)$$

1. $\mathrm{S} \rightarrow \mathrm{R}(start)$
2. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)\mathrm{R}(r_j.t)$
3. $\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)$
4. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)\mathrm{FS}(r, r.f_j)$
5. $\mathrm{FS}(r, r.f_i) \rightarrow \mathrm{F}(r, r.f_j)$
6. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)\mathrm{F}(r, r.f)$
7. $\mathrm{F}(r, r.f) \rightarrow \mathrm{W}(r, r.f)$
8. $\mathrm{W}(r, r.f) \rightarrow \alpha$
9. $\mathrm{W}(r, r.f) \rightarrow \mathrm{g}(f.v)$

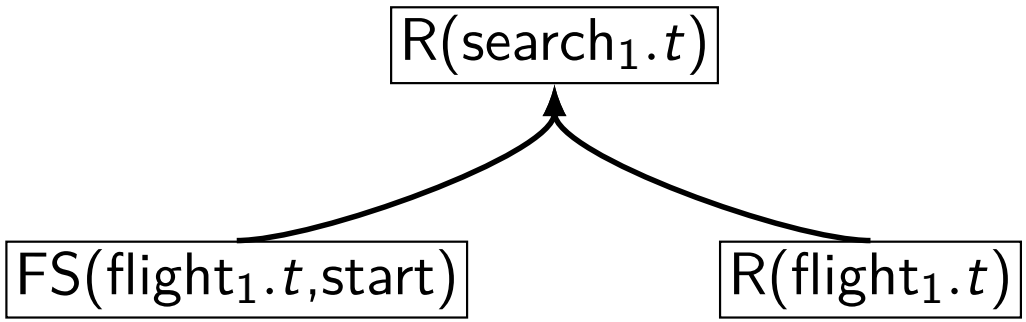
Grammar Definition

$W(search_1, type) \rightarrow \text{show } [type.v = \text{`query'}]$

1. $S \rightarrow R(start)$
2. $R(r_i.t) \rightarrow FS(r_j, start)R(r_j.t)$
3. $R(r_i.t) \rightarrow FS(r_j, start)$
4. $FS(r, r.f_i) \rightarrow F(r, r.f_j)FS(r, r.f_j)$
5. $FS(r, r.f_i) \rightarrow F(r, r.f_j)$
6. $F(r, r.f) \rightarrow W(r, r.f)F(r, r.f)$
7. $F(r, r.f) \rightarrow W(r, r.f)$
8. $W(r, r.f) \rightarrow \alpha$
9. $W(r, r.f) \rightarrow g(f.v)$

Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$

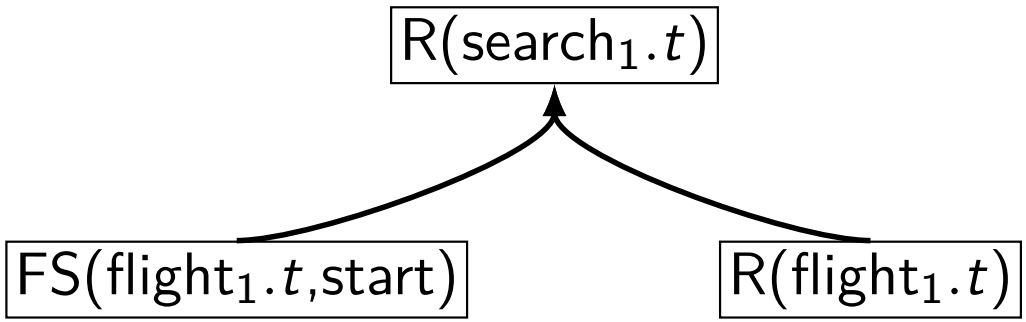


$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\,\mathrm{R}(r_j.t)$$

Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$

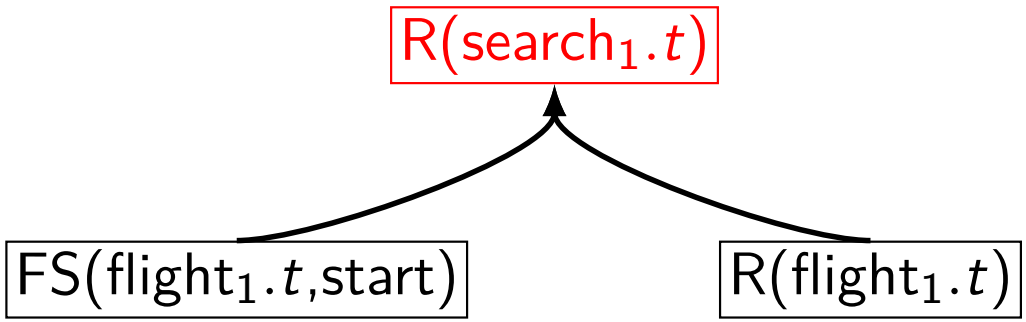


$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\,\mathrm{R}(r_j.t)$$

Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$

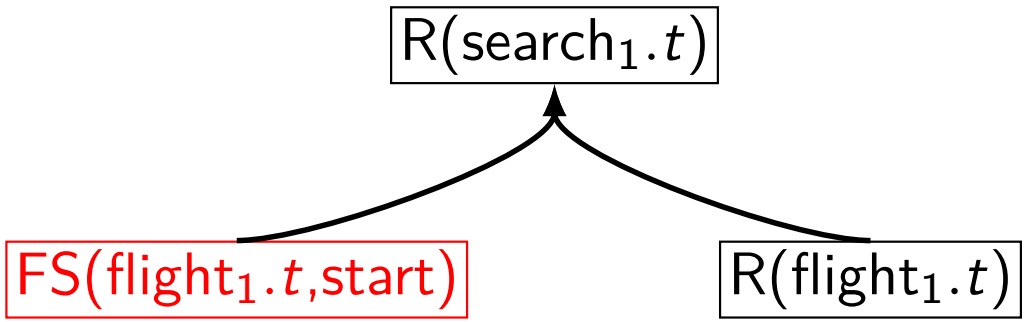


$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\,\mathrm{R}(r_j.t)$$

Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$

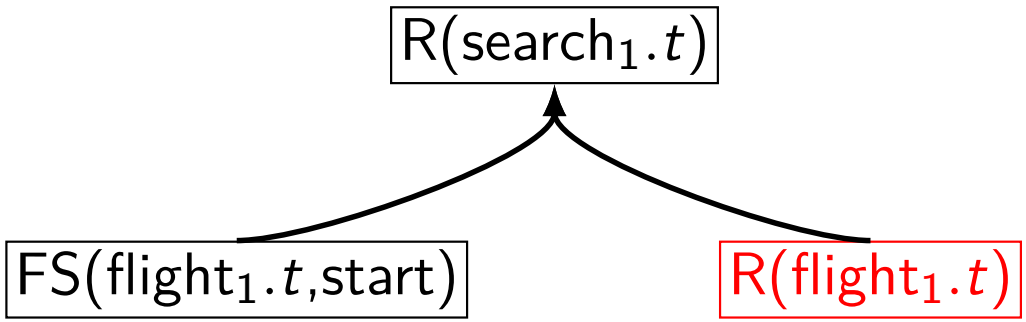


$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\,\mathrm{R}(r_j.t)$$

Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$

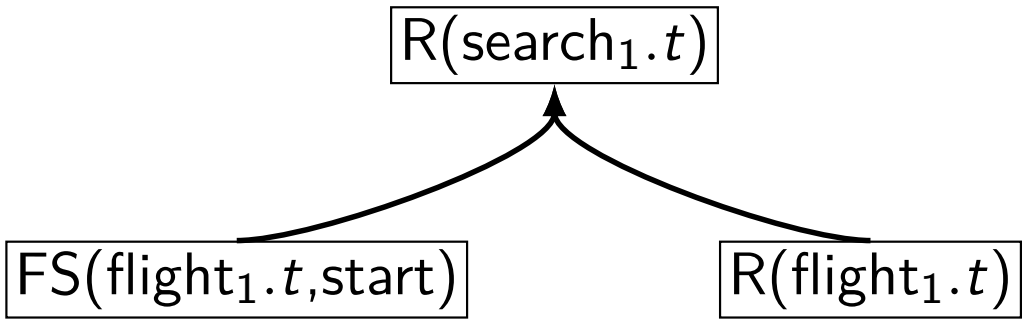


$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\, \mathrm{R}(r_j.t)$$

Hypergraph Construction

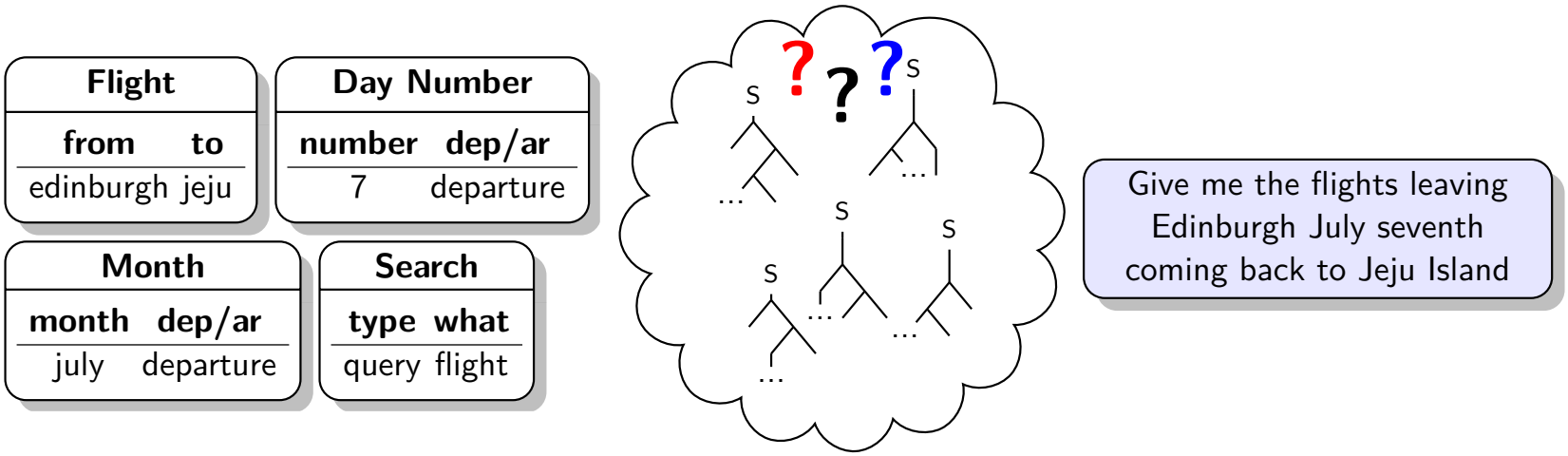
Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, \mathbf{R} \rangle$



$$f(e) = f(\mathrm{FS}_{5,7}(\mathrm{flight}_1.t, \mathit{start})) \otimes f(\mathrm{R}_{7,9}(\mathrm{flight}_1.t)) \otimes w(\mathrm{R}(\mathrm{search}_1.t) \rightarrow \mathrm{FS}(\mathrm{flight}_1, \mathit{start})\ \mathrm{R}(\mathrm{flight}_1.t))$$

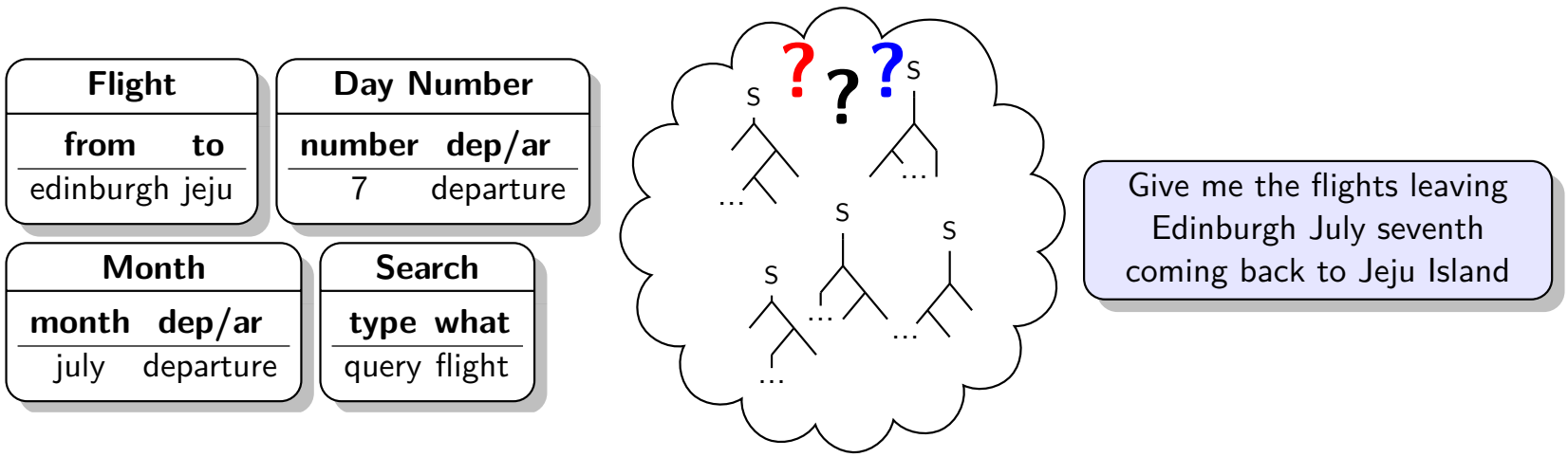
$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, \mathit{start})\,\mathrm{R}(r_j.t)$$

Hypergraph Reranking



Hidden correspondence **h** between database **d** and words **w**

Hypergraph Reranking

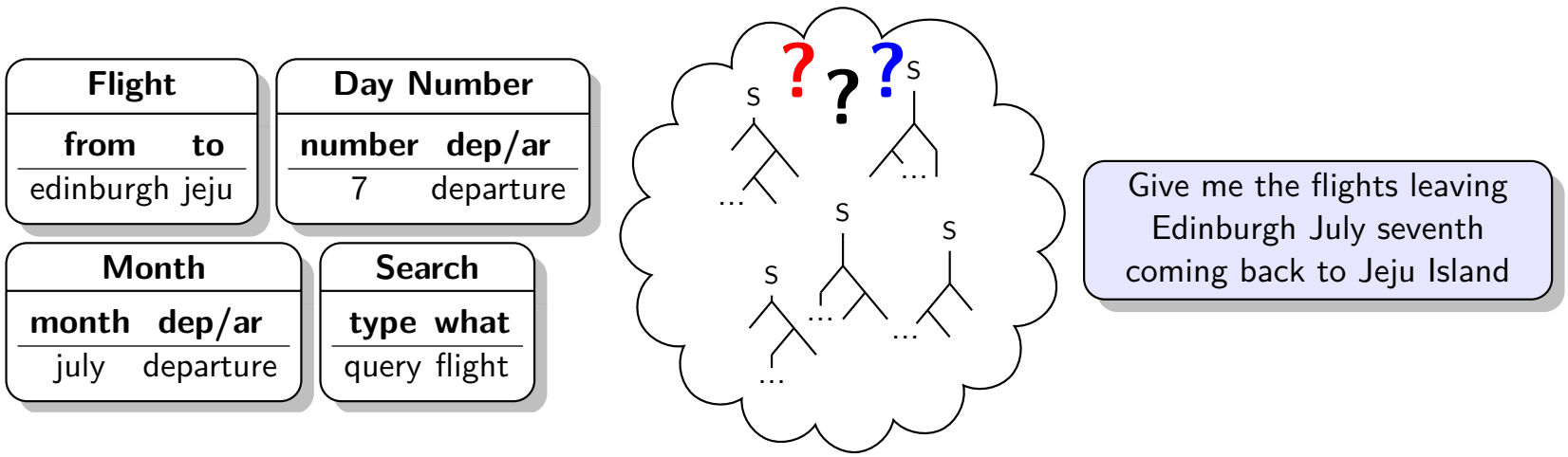


Hidden correspondence **h** between database **d** and words **w**

$$(\widehat{\mathbf{w}}, \widehat{\mathbf{h}}) = \arg\max_{w,h} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})$$

- $\Phi = (\Phi_1, \ldots, \Phi_m)$: high dimensional feature representation
- α : weight vector
- Learn α with averaged structured perceptron (Collins, 2002)

Hypergraph Reranking

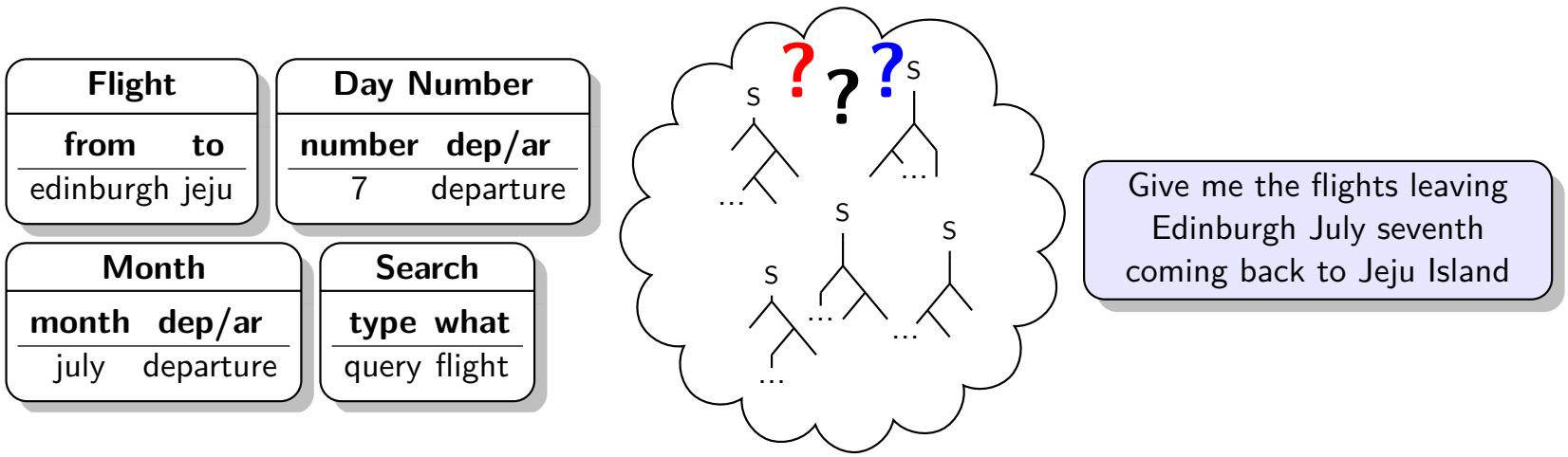


Hidden correspondence **h** between database **d** and words **w**

$$(\widehat{\mathbf{w}}, \widehat{\mathbf{h}}) = \arg\max_{w,h} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})$$

- $\Phi = (\Phi_1, \ldots, \Phi_m)$: high dimensional feature representation
- α : weight vector
- Learn α with averaged structured perceptron (Collins, 2002)

Hypergraph Reranking



Hidden correspondence **h** between database **d** and words **w**

$$(\widehat{\mathbf{w}}, \widehat{\mathbf{h}}) = \arg\max_{w,h} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})$$

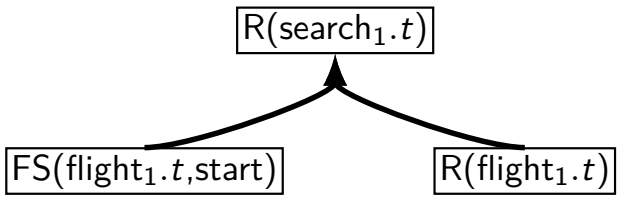
- $\Phi = (\Phi_1, \ldots, \Phi_m)$: high dimensional feature representation
- α : weight vector
- Learn α with averaged structured perceptron (Collins, 2002)

Baseline Features

- Baseline Model Feature (local) : Log score of unsupervised generative decoder (Konstas and Lapata, 2012a)

Baseline Features

- Baseline Model Feature (local) : Log score of unsupervised generative decoder (Konstas and Lapata, 2012a)
- Alignment Features (local) : Count of each PCFG rule



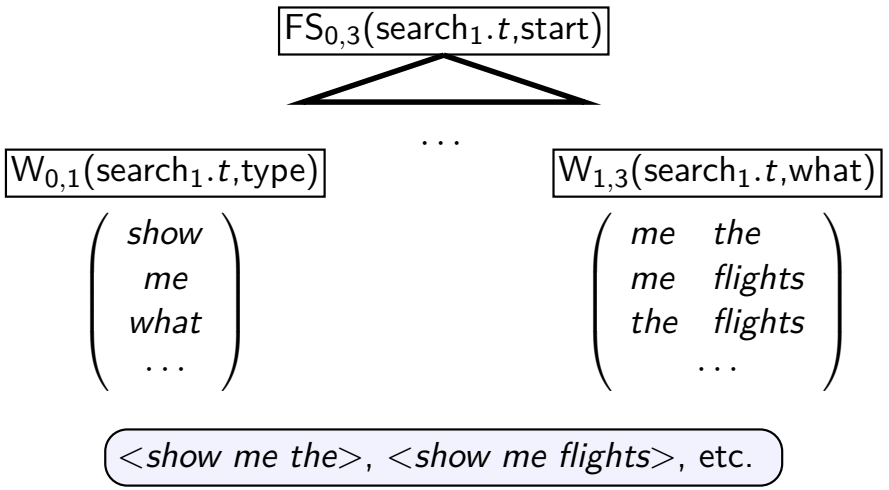
$$\mathrm{R}(r_i.t) \rightarrow \mathrm{FS}(r_j, start)\mathrm{R}(r_j.t)$$

Lexical Features

- Word Bigrams/Trigrams (non-local)
- Number of Words per Field (local)
- Consecutive Word/Bigram/Trigram (non-local)

Lexical Features

- Word Bigrams/Trigrams (non-local)
- Number of Words per Field (local)
- Consecutive Word/Bigram/Trigram (non-local)

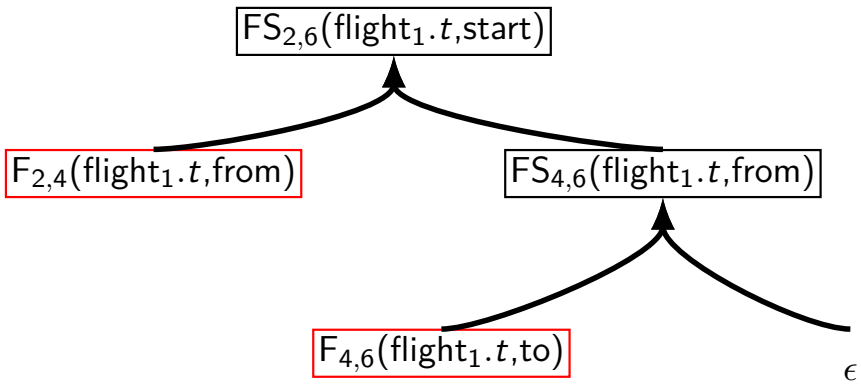


Structural Features

- Field Bigrams/Trigrams (non-local)
- Number of Fields per Record (local)
- Fields with no Value (local)

Structural Features

- Field Bigrams/Trigrams (non-local)
- Number of Fields per Record (local)
- Fields with no Value (local)



<*from to*> | *flight*

k-best Decoding

- Bottom-up Viterbi search

k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)

k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)
- Score of **j**-th derivation: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(< e, \mathbf{j} >)$

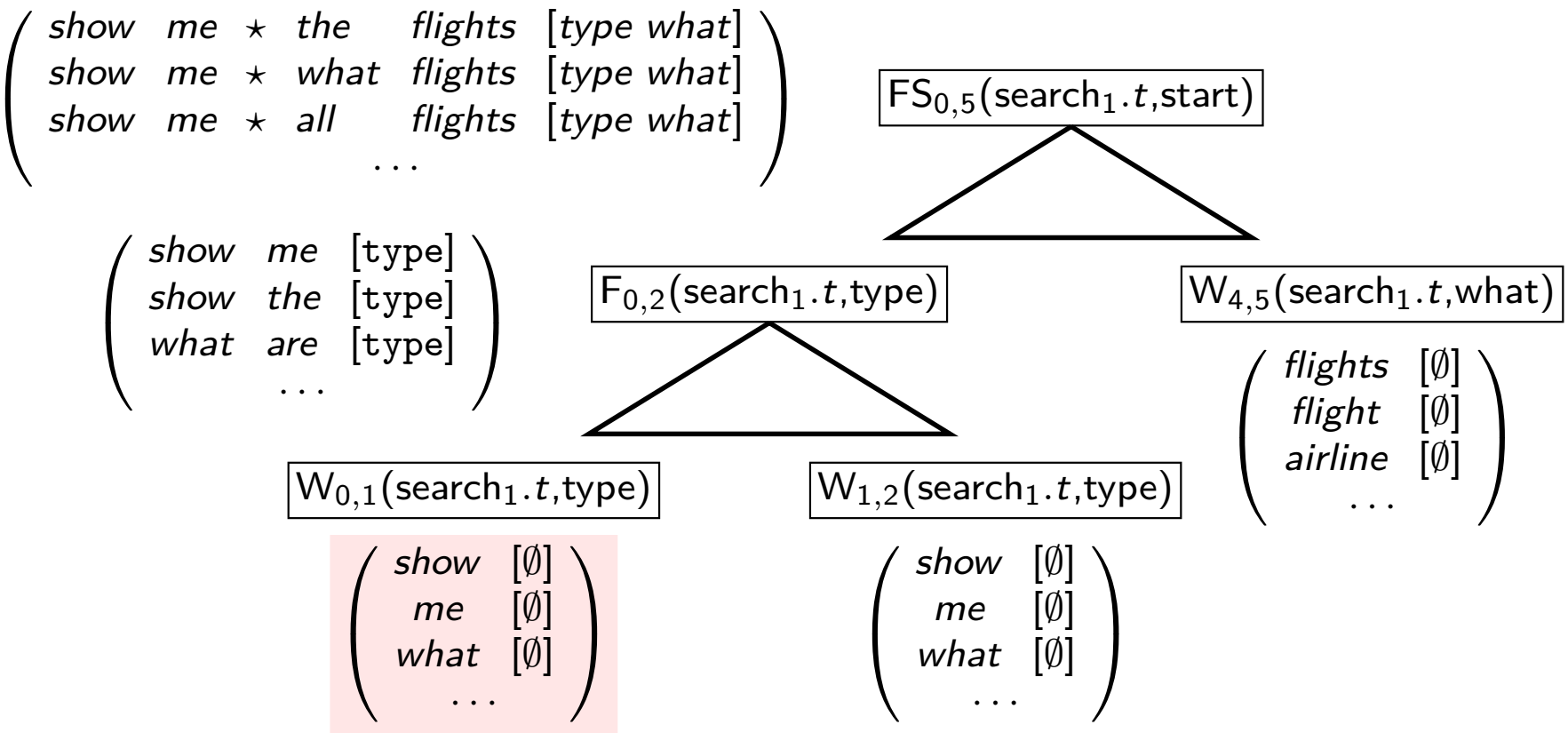
k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)
- Score of **j**-th derivation: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(< e, \mathbf{j} >)$
- Nodes in hypergraph augmented with lexical and structural sub-strings (Huang and Chiang, 2007)
  e.g. $R_{2,8}(flight_1.t)$<one way⋆to Seoul; direction from⋆stop-over to>

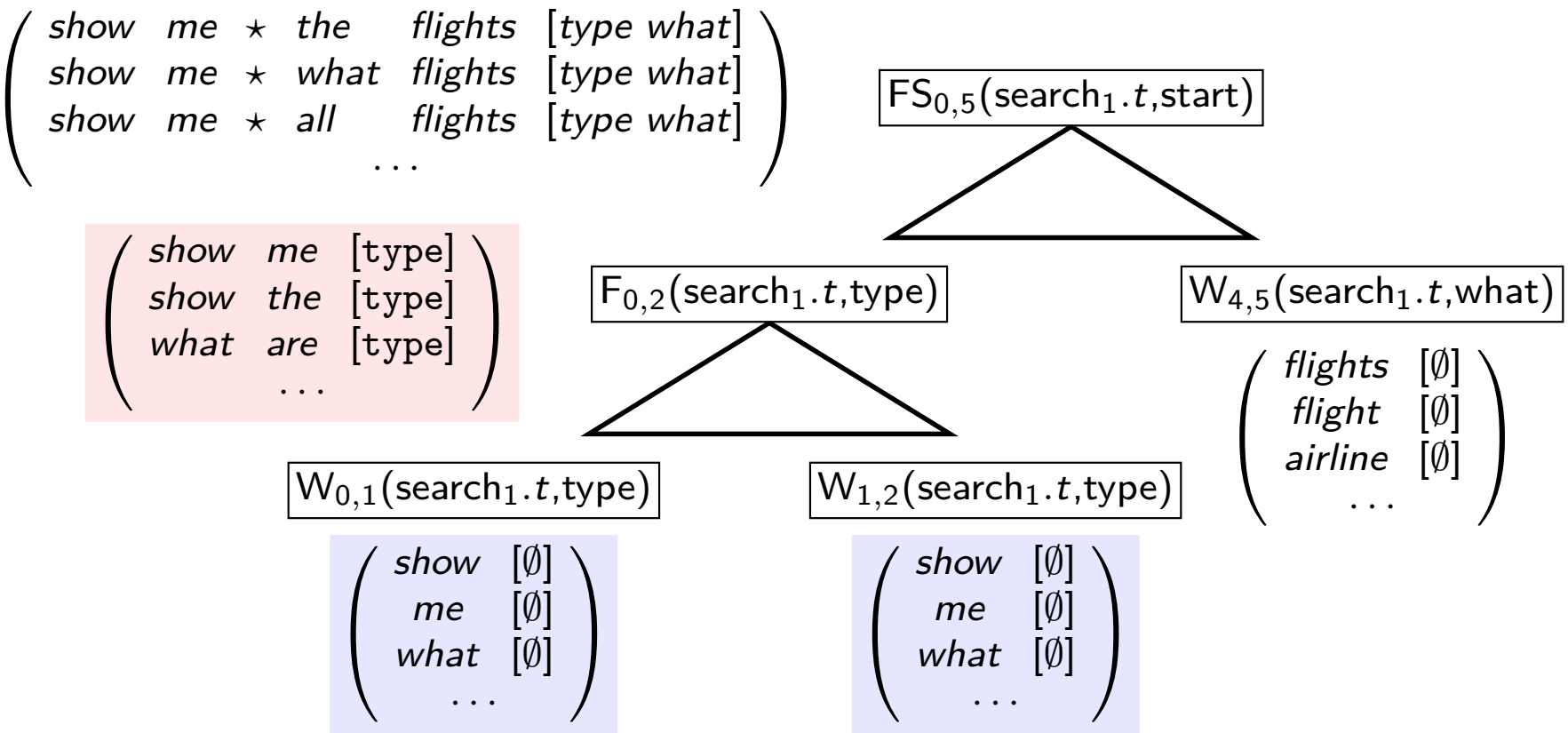
k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)
- Score of **j**-th derivation: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(< e, \mathbf{j} >)$
- Nodes in hypergraph augmented with lexical and structural sub-strings (Huang and Chiang, 2007)
  e.g. $R_{2,8}(flight_1.t)$<one way⋆to Seoul; direction from⋆stop-over to>
- Root node: most grammatical and semantically correct derivation

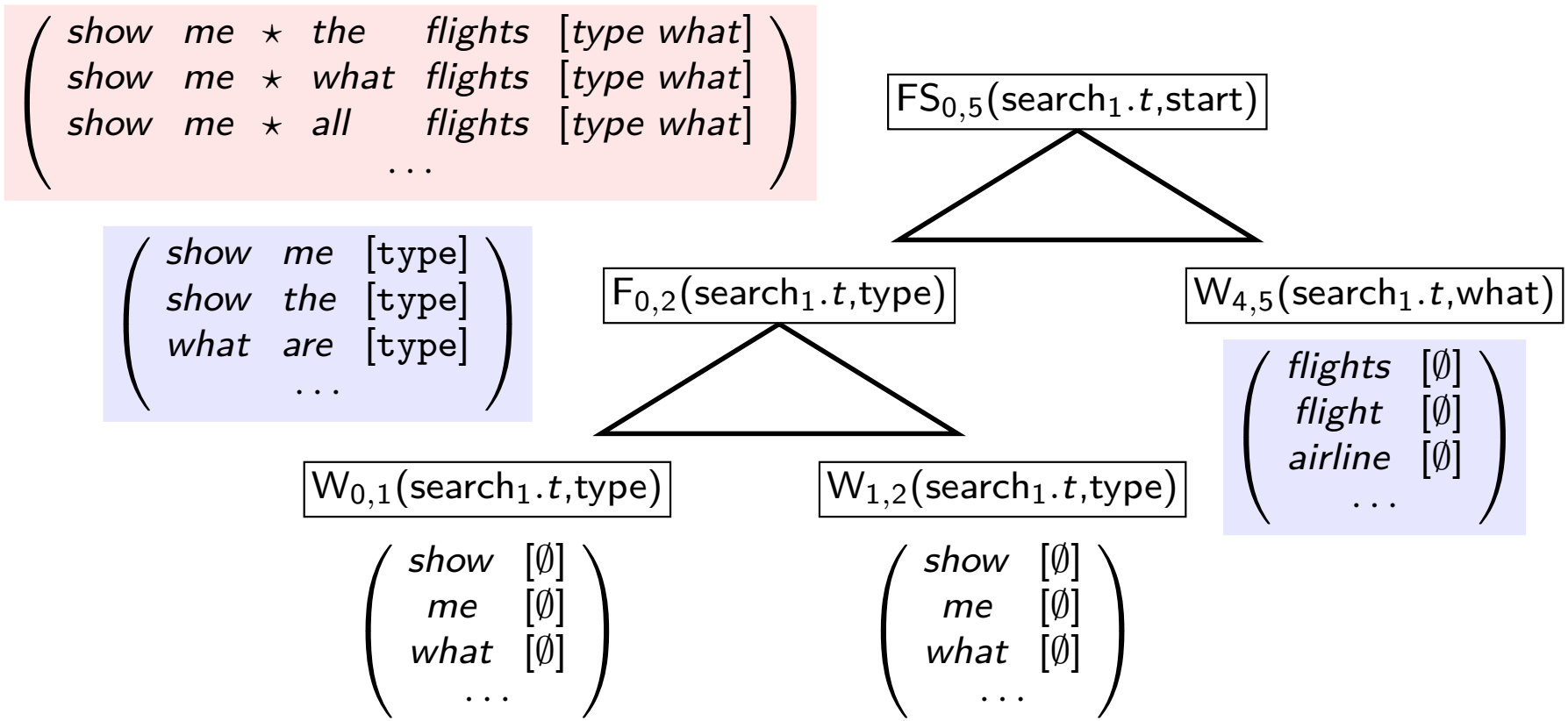
k-best Decoding



k-best Decoding



k-best Decoding



Experimental Setup

Data

- ATIS : mapping from λ-version (Zettlemoyer and Collins, 2007)
- Model parameters estimated on dev set (k-best, n-grams)

Experimental Setup

Data

- ATIS : mapping from λ-version (Zettlemoyer and Collins, 2007)
- Model parameters estimated on dev set (k-best, n-grams)

Evaluation

- Automatic evaluation: BLEU-4, METEOR
- Human evaluation (MTurk): fluency, semantic correctness

Experimental Setup

Data

- ATIS : mapping from λ-version (Zettlemoyer and Collins, 2007)
- Model parameters estimated on dev set (k-best, n-grams)

Evaluation

- Automatic evaluation: BLEU-4, METEOR
- Human evaluation (MTurk): fluency, semantic correctness

System Comparison

- Baseline: 1-BEST+BASE+ALIGN
- k-best (+Lexical): k-BEST+BASE+ALIGN+LEX
- k-best (+Structural): k-BEST+BASE+ALIGN+LEX+STR
- Angeli et al. (2010)

Results: Automatic Evaluation

System	BLEU	METEOR
1-BEST+BASE+ALIGN	21.93	34.01
k-BEST+BASE+ALIGN+LEX	28.66	45.18
k-BEST+BASE+ALIGN+LEX+STR	30.62	46.07
ANGELI	26.77	42.41

Results: Automatic Evaluation

System	BLEU	METEOR
1-BEST+BASE+ALIGN	21.93	34.01
k-BEST+BASE+ALIGN+LEX	28.66	45.18
k-BEST+BASE+ALIGN+LEX+STR	30.62	46.07
ANGELI	26.77	42.41

Results: Human Evaluation

System	Fluency	SemCor
1-Best	2.70	3.05
k-Best	4.02	4.04
Angeli	3.74	3.17
Human	4.18	4.02

Results: Human Evaluation

System	Fluency	SemCor
1-Best	2.70	3.05
k-Best	4.02	4.04
Angeli	3.74	3.17
Human	4.18	4.02

- *k*-Best significantly better than 1-best and Angeli ($\alpha < 0.01$)
- *k*-Best and Human are not significantly different

Output

Flight		Time		Day		Search	
from	**to**	**when**	**dep/ar**	**day**	**dep/ar**	**type**	**what**
phoenix	milwaukee	evening	departure	wednesday	departure	query	flight

1-BEST: On Wednesday evening from from Phoenix to Milwaukee on Wednesday evening

k-BEST: **Please list the flights from Phoenix to Milwaukee on Wednesday evening**

ANGELI: Show me the flights from Phoenix to Milwaukee on Wednesday evening flights from Phoenix to Milwaukee

HUMAN: Give me the flights from Phoenix to Milwaukee on Wednesday evening

Conclusions

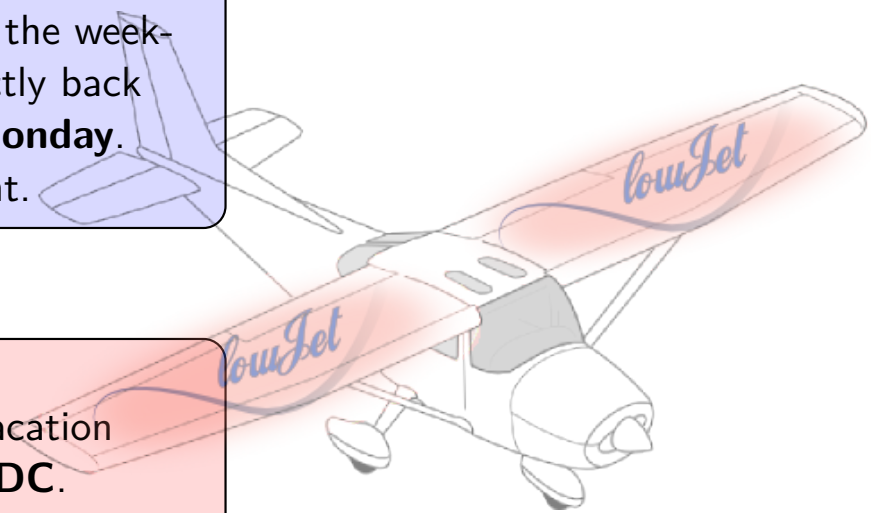
- Generation as parsing problem using the hypergraph framework
- Discriminative reranking using the structured perceptron
- Introduced local and non-local features
- Performance better than state-of-the-art
- Future work: dependency relations, discourse

Demo

Flight Booking System - lowJet

Scenario 1
Imagine you are in **Los Angeles** for the weekend and you would like to fly directly back home to **Denver** the following **Monday**.
Try to find the **earliest** flight.

Scenario 2
You have won a free **one-way** vacation from **Miami** to **Washington DC**.
Book the **cheapest** tickets for next **Thursday** morning.
Alternatively, just find the **earliest** itinerary.



Thank you

Questions ?

Oracle Derivation

Oracle derivation $(\mathbf{w}^*, \mathbf{h}^+)$

- Use the existing decoder but observe the training text.
- $\mathbf{w}^*$: gold standard text
- $\mathbf{h}^+$: best latent configuration

Human Evaluation (Mturk)

Table 3

Category	Fields - Values	
 Flight Info	**from:** milwaukee	**to:** phoenix
 Query	**type:** show	**what:** flight
 Day	**day:** saturday	**dep/ar/ret:** departure

Translation 3

SHOW ME THE FLIGHTS BETWEEN MILWAUKEE AND PHOENIX ON SATURDAY

Fluency ○ 1 ○ 2 ○ 3 ○ 4 ○ 5

Semantic Correctness ○ 1 ○ 2 ○ 3 ○ 4 ○ 5

Determining Text Length

- Train a linear regression model
- Idea: The more records and fields that have values in the database $\rightarrow$ the more facts need to be uttered
- Input to the model: Flattened version of the database input, i.e. each feature is a record-field pair
- Feature values: Values vs Counts of Fields