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

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

Traditional NLG Pipeline

1. Input Data
2. Communicative Goal
3. Content Selection
4. Surface Realisation
5. Text

Konstas, Lapata (ILCC)  
ACL 2012, Jeju Island
Traditional NLG Pipeline

Input Data

Content Selection

Surface Realisation

Text

Communicative Goal
Our Approach

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to</td>
<td>number dep/ar</td>
<td>month dep/ar</td>
</tr>
<tr>
<td>edinburgh jeju</td>
<td>7 departure</td>
<td>july departure</td>
</tr>
</tbody>
</table>

Condition

<table>
<thead>
<tr>
<th>arg1 arg2 type</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrival_time 1600 &lt;</td>
</tr>
</tbody>
</table>

Search

<table>
<thead>
<tr>
<th>type what query flight</th>
</tr>
</thead>
</table>

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
<th>Condition</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to edinburgh jeju</td>
<td>number dep/ar</td>
<td>month dep/ar</td>
<td>arg1 arg2 type</td>
<td>type what</td>
</tr>
<tr>
<td></td>
<td>7 departure</td>
<td>july departure</td>
<td>arrival_time 1600</td>
<td>query flight</td>
</tr>
</tbody>
</table>

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

#### Flight
- **from**: Edinburgh
- **to**: 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.
Our Approach

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>number</td>
<td>month</td>
</tr>
<tr>
<td>edinburgh</td>
<td>dep/ar</td>
<td>dep/ar</td>
</tr>
<tr>
<td>to</td>
<td>7</td>
<td>july</td>
</tr>
<tr>
<td>jeju</td>
<td>departure</td>
<td>departure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>arg1</td>
<td>type</td>
</tr>
<tr>
<td>arg2</td>
<td>what</td>
</tr>
<tr>
<td>type</td>
<td>query</td>
</tr>
<tr>
<td>arrival_time 1600</td>
<td>flight</td>
</tr>
</tbody>
</table>
Give me the flights leaving Edinburgh July seventh coming back to Jeju Island before 4pm.
### Flight

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>number</td>
<td>month</td>
</tr>
<tr>
<td>edinburgh</td>
<td>7</td>
<td>july</td>
</tr>
<tr>
<td>to</td>
<td>dep/ar</td>
<td>dep/ar</td>
</tr>
<tr>
<td>jeju</td>
<td>departure</td>
<td>departure</td>
</tr>
</tbody>
</table>

### Condition

<table>
<thead>
<tr>
<th>Condition</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>arg1</td>
<td>type</td>
</tr>
<tr>
<td>arg2</td>
<td>what</td>
</tr>
<tr>
<td>type</td>
<td>flight</td>
</tr>
<tr>
<td>arrival_time 1600</td>
<td>&lt;</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>from Edinburgh to Jeju</td>
<td>number 7</td>
<td>month july</td>
</tr>
<tr>
<td>Departure</td>
<td>Departure</td>
<td>Departure</td>
</tr>
</tbody>
</table>

Condition

<table>
<thead>
<tr>
<th>arg1</th>
<th>arg2</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrival_time 1600</td>
<td>&lt;</td>
<td></td>
</tr>
</tbody>
</table>

Search

<table>
<thead>
<tr>
<th>type</th>
<th>what</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>flight</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
<th>Month</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>number</td>
<td>month</td>
</tr>
<tr>
<td>edinburgh</td>
<td>dep/ar</td>
<td>dep/ar</td>
</tr>
<tr>
<td>to</td>
<td>7</td>
<td>july</td>
</tr>
<tr>
<td>jeju</td>
<td>departure</td>
<td>departure</td>
</tr>
</tbody>
</table>

**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

**Training**

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to</td>
<td>number dep/ar</td>
</tr>
<tr>
<td>edinburgh jeju</td>
<td>7 departure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>month dep/ar</td>
<td>type what</td>
</tr>
<tr>
<td>july departure</td>
<td>query flight</td>
</tr>
</tbody>
</table>

---

*Give me the flights leaving Edinburgh July seventh coming back to Jeju Island*

---

**Concept-to-text Generation**

[ACL 2012, Jeju Island]
Joint Discriminative Reranking with Hypergraphs

Training

S \rightarrow R(\text{start})
R(r_j, t) \rightarrow FS(r_j, \text{start}) R(r_j, t)
R(r_j, t) \rightarrow FS(r_j, \text{start})
FS(r, r.f_i) \rightarrow F(r, r.f_i) FS(r, r.f_i)
FS(r, r.f_i) \rightarrow F(r, r.f_i)
F(r, r.f) \rightarrow W(r, r.f) F(r, r.f)
F(r, r.f) \rightarrow W(r, r.f)
W(r, r.f) \rightarrow \alpha_k
W(r, r.f) \rightarrow g(f, v)

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island
Joint Discriminative Reranking with Hypergraphs

Training

S → R(start)
R(rj.t)→FS(rj, start)R(rj.t)
R(rj.t)→FS(rj, start)
FS(r, r.fj)→F(r, r.fj)FS(r, r.fj)
FS(r, r.fj)→F(r, r.fj)
F(r, r.f)→W(r, r.f)F(r, r.f)
F(r, r.f)→W(r, r.f)
W(r, r.f)→α
W(r, r.f)→g(f.v)

Month

Month dep/ar
july departure

Search

type what
query flight

Day Number

number dep/ar
7 departure

Flight

from to
edinburgh jeju

f(e): Φ = (Φ₁, ..., Φ_m)
Train using Perceptron

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

f₀₁(query₁.type)
f₀₁(query₁.what)
f₀₂(query₁.type)
f₀₂(query₁.what)
FS₁,₂(query₁.start)
FS₀,₁(query₁.start)
FS₀,₂(query₁.start)
Joint Discriminative Reranking with Hypergraphs

**Training**

- **Flight**: from to
  - Edinburgh Jeju
- **Day Number**: number dep/ar
  - 7 departure
- **Month**: month dep/ar
  - July departure
- **Search**: type what
  - query flight

**Search**

- $S \rightarrow R(start)$
- $R(rj,t) \rightarrow FS(rj,start) R(rj,t)$
- $R(rj,t) \rightarrow FS(rj,start)$
- $FS(r,r.fj) \rightarrow F(r,r.fj) FS(r,r.fj)$
- $FS(r,r.fj) \rightarrow F(r,r.fj)$
- $F(r,r.f) \rightarrow W(r,r.f) F(r,r.f)$
- $F(r,r.f) \rightarrow W(r,r.f)$
- $W(r,r.f) \rightarrow \alpha$
- $W(r,r.f) \rightarrow g(f.v)$

**Testing**

- $f(e): \Phi = (\Phi_1, \ldots, \Phi_m)$
- **train using Perceptron**

**k-best decoding via reranking**

**Example Queries**

- Show me all flights
- Show me what flights
- Show me the flights

**Concept-to-text Generation**

Konstas, Lapata (ILCC)
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)
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 $r \in d$ has a type $r.t$ and fields $f$
- Fields have values $f.v$ and types $f.t$ (integer, categorical)

<table>
<thead>
<tr>
<th>Day</th>
<th>day of week</th>
<th>dep/ar/ret</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sunday</td>
<td>departure</td>
</tr>
<tr>
<td></td>
<td>thursday</td>
<td>return</td>
</tr>
</tbody>
</table>

leaving on sunday
**Input**

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

<table>
<thead>
<tr>
<th>Day</th>
<th>dep/ar/ret</th>
</tr>
</thead>
<tbody>
<tr>
<td>day of week</td>
<td></td>
</tr>
<tr>
<td>sunday</td>
<td>departure</td>
</tr>
<tr>
<td>thursday</td>
<td>return</td>
</tr>
</tbody>
</table>

leaving on sunday
Input

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

<table>
<thead>
<tr>
<th>Day</th>
<th>dep/ar/ret</th>
</tr>
</thead>
<tbody>
<tr>
<td>day of week</td>
<td></td>
</tr>
<tr>
<td>sunday</td>
<td>departure</td>
</tr>
<tr>
<td>thursday</td>
<td>return</td>
</tr>
</tbody>
</table>

leaving on sunday
Input: database records \( d \)

Output: words \( w \) corresponding to some records of \( d \)

Each record \( r \in d \) has a type \( r.t \) and fields \( f \)

Fields have values \( f.v \) and types \( f.t \) (integer, categorical)

<table>
<thead>
<tr>
<th>Day</th>
<th>dep/ar/ret</th>
</tr>
</thead>
<tbody>
<tr>
<td>day of week</td>
<td>departure</td>
</tr>
<tr>
<td>sunday</td>
<td>return</td>
</tr>
</tbody>
</table>

leaving on sunday
Input:

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

<table>
<thead>
<tr>
<th>Day</th>
<th>dep/ar/ret</th>
</tr>
</thead>
<tbody>
<tr>
<td>day of week</td>
<td></td>
</tr>
<tr>
<td>sunday</td>
<td>departure</td>
</tr>
<tr>
<td>thursday</td>
<td>return</td>
</tr>
</tbody>
</table>

leaving on sunday
Grammar Definition

1. S → R(start)
2. R(r_i . t) → FS(r_j , start) R(r_j . t)
3. R(r_i . t) → FS(r_j , start)
4. FS(r , r.f_i) → F(r , r.f_j) FS(r , r.f_j)
5. FS(r , r.f_i) → F(r , r.f_j)
6. F(r , r.f) → W(r , r.f) F(r , r.f)
7. F(r , r.f) → W(r , r.f)
8. W(r , r.f) → α
9. W(r , r.f) → g(f . v)
Grammar Definition

\[ R(search_1.t) \rightarrow FS(flight_1, start)R(flight_1.t) \]

1. \[ S \rightarrow R(start) \]
2. \[ R(r_j.t) \rightarrow FS(r_j, start)R(r_j.t) \]
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

\[ FS(flight_1, from) \rightarrow F(flight_1, to)FS(flight_1, to) \]

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

\[ F(\text{search}_1, \text{what}) \rightarrow W(\text{search}_1, \text{what})F(\text{search}_1, \text{what}) \]

1. \( S \rightarrow R(\text{start}) \)
2. \( R(\text{r}_i, \text{t}) \rightarrow FS(\text{r}_j, \text{start})R(\text{r}_j, \text{t}) \)
3. \( R(\text{r}_i, \text{t}) \rightarrow FS(\text{r}_j, \text{start}) \)
4. \( FS(\text{r}, \text{r}.f_i) \rightarrow F(\text{r}, \text{r}.f_j)FS(\text{r}, \text{r}.f_j) \)
5. \( FS(\text{r}, \text{r}.f_i) \rightarrow F(\text{r}, \text{r}.f_j) \)
6. \( F(\text{r}, \text{r}.f) \rightarrow W(\text{r}, \text{r}.f)F(\text{r}, \text{r}.f) \)
7. \( F(\text{r}, \text{r}.f) \rightarrow W(\text{r}, \text{r}.f) \)
8. \( W(\text{r}, \text{r}.f) \rightarrow \alpha \)
9. \( W(\text{r}, \text{r}.f) \rightarrow g(\text{f}.v) \)
Grammar Definition

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

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. \( W(r, r.f) \rightarrow g(f.v) \)
Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, R \rangle$

$$f(e) = f(FS_{5,7}(flight_1.t, start)) \otimes f(R_{7,9}(flight_1.t)) \otimes w(R(search_1.t) \rightarrow FS(flight_1, start) \ R(flight_1.t))$$

$$R(r_i.t) \rightarrow FS(r_j, start)R(r_j.t)$$
Hypergraph Construction

Map standard weighted CYK algorithm to hypergraph \( H : \langle N, E, t, R \rangle \)

\[
f(e) = f(FS_{5,7}(flight_1.t, start)) \otimes f(R_{7,9}(flight_1.t)) \otimes w(R(search_1.t) \rightarrow FS(flight_1, start) R(flight_1.t))
\]

\[
R(r_i.t) \rightarrow FS(r_j, start)R(r_j.t)
\]
Map standard weighted CYK algorithm to hypergraph $H: \langle N, E, t, R \rangle$

$$f(e) = f(FS_{5,7}(flight_1.t, start)) \otimes f(R_{7,9}(flight_1.t)) \otimes w(R(search_1.t) \rightarrow FS(flight_1, start) \ R(flight_1.t))$$

$$R(r_j.t) \rightarrow FS(r_j, start) R(r_j.t)$$
Map standard weighted CYK algorithm to hypergraph $H : \langle N, E, t, R \rangle$

\[
\begin{align*}
  f(e) &= f(FS_{5,7}(flight_1.t, start)) \otimes f(R_{7,9}(flight_1.t)) \otimes w(R(search_1.t) \rightarrow FS(flight_1, start) R(flight_1.t)) \\
  R(r_i.t) &\rightarrow FS(r_j, start) R(r_j.t)
\end{align*}
\]
Hypergraph Construction

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

$$f(e) = f(FS_{5,7}(flight_1.t, start)) \otimes f(R_{7,9}(flight_1.t)) \otimes w(R(search_1.t) \rightarrow FS(flight_1, start) \ R(flight_1.t))$$

$$R(r_i.t) \rightarrow FS(r_j, start) R(r_j.t)$$
Hypergraph Construction

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

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

$$\text{R}(r_i.t) \rightarrow \text{FS}(r_j, \text{start}) \text{R}(r_j.t)$$
Hypergraph Reranking

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to</td>
<td>number dep/ar</td>
</tr>
<tr>
<td>edinburgh jeju</td>
<td>7 departure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>month dep/ar</td>
<td>type what query flight</td>
</tr>
<tr>
<td>july departure</td>
<td></td>
</tr>
</tbody>
</table>

Hidden correspondence $h$ between database $d$ and words $w$
Hypergraph Reranking

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

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

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to</td>
<td>number dep/ar</td>
</tr>
<tr>
<td>edinburgh</td>
<td>7 departure</td>
</tr>
<tr>
<td>jeju</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>month dep/ar</td>
<td>type what</td>
</tr>
<tr>
<td>july</td>
<td>query flight</td>
</tr>
</tbody>
</table>

Give me the flights leaving Edinburgh July seventh coming back to Jeju Island
Hidden correspondence $h$ between database $d$ and words $w$

\[
(\hat{w}, \hat{h}) = \arg \max_{w,h} \alpha \cdot \Phi(d, w, h)
\]

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

<table>
<thead>
<tr>
<th>Flight</th>
<th>Day Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>number</td>
</tr>
<tr>
<td>edinburgh</td>
<td>dep/ar</td>
</tr>
<tr>
<td>to</td>
<td>7</td>
</tr>
<tr>
<td>jeju</td>
<td>departure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>month</td>
<td>type</td>
</tr>
<tr>
<td>july</td>
<td>what</td>
</tr>
<tr>
<td></td>
<td>query</td>
</tr>
<tr>
<td></td>
<td>flight</td>
</tr>
</tbody>
</table>

Hidden correspondence \( \mathbf{h} \) between database \( \mathbf{d} \) and words \( \mathbf{w} \)

\[
(\hat{\mathbf{w}}, \hat{\mathbf{h}}) = \arg \max_{\mathbf{w}, \mathbf{h}} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})
\]

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

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

Alignment Features (local) : Count of each PCFG rule
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)

\[
\begin{align*}
&\text{FS}_{0,3}(\text{search}_1.t, \text{start}) \\
&\text{W}_{0,1}(\text{search}_1.t, \text{type}) \\
&\quad \left( \begin{array}{c}
\text{show} \\
\text{me} \\
\text{what} \\
\cdots
\end{array} \right) \\
&\text{W}_{1,3}(\text{search}_1.t, \text{what}) \\
&\quad \left( \begin{array}{c}
\text{me} \\
\text{the} \\
\text{me flights} \\
\text{the flights} \\
\cdots
\end{array} \right)
\end{align*}
\]

\(<\text{show me the}>, <\text{show me flights}>, \text{etc.}\)
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)
k-best Decoding

- Bottom-up Viterbi search

\[
\text{Score of } j\text{-th derivation: } \alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(<e, j>)
\]

Nodes in hypergraph augmented with lexical and structural sub-strings (Huang and Chiang, 2007)

e.g. R2, 8 (flight 1, t) < one way ⋆ to Seoul; ⋆ stop-over to ⋆ stop-over>

Root node: most grammatical and semantically correct derivation
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: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(<e,j>)$
k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)
- Score of j-th derivation: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(<e,j>)$
- Nodes in hypergraph augmented with lexical and structural sub-strings (Huang and Chiang, 2007)
  e.g. $R_{2,8}(\text{flight}_1.t)<\text{one way to Seoul; direction from stop-over to}>$
k-best Decoding

- Bottom-up Viterbi search
- Keep k-best derivations at each node, cube pruning (Chiang, 2007)
- Score of j-th derivation: $\alpha \cdot \Phi_L(e) + \alpha \cdot \Phi_N(<e,j>)$
- Nodes in hypergraph augmented with lexical and structural sub-strings (Huang and Chiang, 2007)
  e.g. $R_{2,8}(flight_{1.t})<\text{one way}*, \text{to Seoul}; \text{direction from}*, \text{stop-over to}>$
- Root node: most grammatical and semantically correct derivation
k-best Decoding

\[
\begin{align*}
&\left(\text{show me the flights} \ [\text{type what}]\right) \\
&\left(\text{show me what flights} \ [\text{type what}]\right) \\
&\left(\text{show me all flights} \ [\text{type what}]\right) \\
&\ldots
\end{align*}
\]

\[
\begin{align*}
&\left(\text{show me} \ [\text{type}]\right) \\
&\left(\text{show the} \ [\text{type}]\right) \\
&\left(\text{what are} \ [\text{type}]\right) \\
&\ldots
\end{align*}
\]

\[
\begin{align*}
&F_{0,2}(\text{search}_1.t,\text{type}) \\
&F_{0,5}(\text{search}_1.t,\text{start}) \\
&W_{0,1}(\text{search}_1.t,\text{type}) \\
&W_{1,2}(\text{search}_1.t,\text{type}) \\
&W_{4,5}(\text{search}_1.t,\text{what})
\end{align*}
\]
### k-best Decoding

\[
\begin{align*}
\text{show me } & \ast \text{ the flights } [\text{type what}] \\
\text{show me } & \ast \text{ what flights } [\text{type what}] \\
\text{show me } & \ast \text{ all flights } [\text{type what}] \\
\end{align*}
\]

\[
\begin{align*}
\text{FS}_{0,5}(\text{search}_1.t, \text{start}) \\
\text{F}_{0,2}(\text{search}_1.t, \text{type}) \\
\text{W}_{4,5}(\text{search}_1.t, \text{what}) \\
\end{align*}
\]

\[
\begin{align*}
\text{W}_{0,1}(\text{search}_1.t, \text{type}) \\
\text{W}_{1,2}(\text{search}_1.t, \text{type}) \\
\end{align*}
\]
k-best Decoding

\[
\begin{align*}
F_{0,5}(\text{search}_{1.t}, \text{start}) & \\
F_{0,2}(\text{search}_{1.t}, \text{type}) & \\
W_{0,1}(\text{search}_{1.t}, \text{type}) & \\
W_{1,2}(\text{search}_{1.t}, \text{type}) & \\
W_{4,5}(\text{search}_{1.t}, \text{what}) & \\
\end{align*}
\]
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 $\lambda$-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 \( \lambda \)-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

<table>
<thead>
<tr>
<th>System</th>
<th>BLEU</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-BEST + BASE + ALIGN</td>
<td>21.93</td>
<td>34.01</td>
</tr>
<tr>
<td>$k$-BEST + BASE + ALIGN + LEX</td>
<td>28.66</td>
<td>45.18</td>
</tr>
<tr>
<td>$k$-BEST + BASE + ALIGN + LEX + STR</td>
<td>30.62</td>
<td>46.07</td>
</tr>
<tr>
<td>ANGELI</td>
<td>26.77</td>
<td>42.41</td>
</tr>
</tbody>
</table>
### Results: Automatic Evaluation

<table>
<thead>
<tr>
<th>System</th>
<th>BLEU</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-best + base + align</td>
<td>21.93</td>
<td>34.01</td>
</tr>
<tr>
<td>k-best + base + align + lex</td>
<td>28.66</td>
<td>45.18</td>
</tr>
<tr>
<td>k-best + base + align + lex + str</td>
<td>30.62</td>
<td>46.07</td>
</tr>
<tr>
<td>Angeli</td>
<td>26.77</td>
<td>42.41</td>
</tr>
</tbody>
</table>
## Results: Human Evaluation

<table>
<thead>
<tr>
<th>System</th>
<th>Fluency</th>
<th>SemCor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-BEST</td>
<td>2.70</td>
<td>3.05</td>
</tr>
<tr>
<td>k-BEST</td>
<td>4.02</td>
<td>4.04</td>
</tr>
<tr>
<td>ANGELI</td>
<td>3.74</td>
<td>3.17</td>
</tr>
<tr>
<td>HUMAN</td>
<td>4.18</td>
<td>4.02</td>
</tr>
</tbody>
</table>

1-BEST and k-BEST are significantly better than 1-best and ANGELI ($\alpha < 0.01$). k-BEST and HUMAN are not significantly different.
Results: Human Evaluation

<table>
<thead>
<tr>
<th>System</th>
<th>Fluency</th>
<th>SemCor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-Best</td>
<td>2.70</td>
<td>3.05</td>
</tr>
<tr>
<td>k-Best</td>
<td>4.02</td>
<td>4.04</td>
</tr>
<tr>
<td>ANGELI</td>
<td>3.74</td>
<td>3.17</td>
</tr>
<tr>
<td>HUMAN</td>
<td>4.18</td>
<td>4.02</td>
</tr>
</tbody>
</table>

- **k-Best** significantly better than **1-Best** and **ANGELI** ($\alpha < 0.01$)
- **k-Best** and **HUMAN** are not significantly different
Results

Output

<table>
<thead>
<tr>
<th>Flight</th>
<th>Time</th>
<th>Day</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>when</td>
<td>day</td>
<td>type</td>
</tr>
<tr>
<td>phoenix</td>
<td>dep/ar</td>
<td>dep/ar</td>
<td>what</td>
</tr>
<tr>
<td>to</td>
<td>evening departure</td>
<td>Wednesday departure</td>
<td>query flight</td>
</tr>
</tbody>
</table>

1-Best: On Wednesday evening 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
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 \((w^*, h^+)\)

- Use the existing decoder but observe the training text.
- \(w^*\): gold standard text
- \(h^+\): best latent configuration
Human Evaluation (Mturk)

<table>
<thead>
<tr>
<th>Category</th>
<th>Fields - Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flight Info</td>
<td>from: milwaukee, to: phoenix</td>
</tr>
<tr>
<td>Query</td>
<td>type: show, what: flight</td>
</tr>
<tr>
<td>Day</td>
<td>day: saturday, dep/ar/ret: departure</td>
</tr>
</tbody>
</table>

Translation 3

SHOW ME THE FLIGHTS BETWEEN MILWAUKEE AND PHOENIX ON SATURDAY

Fluency

Semantic Correctness
Train a linear regression model

Idea: The more records and fields that have values in the database → 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