Recent findings in Statistical Transfer

Michael Wayne Goodman

2016-06-16

Recap

- My goal is to do SMT using semantic structure on both sides
- Premise:
  - Rule-based transfer is not easily scalable to many language pairs
  - But we can parse bitext corpora to get bisem corpora
  - So maybe we can learn to transfer for any language pair with decent grammar coverage?
- I assume resources to parse semantics and realize sentences
  - …and therefore focus only on sem-to-sem transfer
  - …but I still need sentences for BLEU scores
- Spoiler: no BLEU scores will be reported in this presentation

Recap

- Last year I discussed *MrsPaths*:
  - Lossy MRS trees without variables or other node identifiers
  - Could be used as a query language (like XPath is for XML)
  - …or as a hashable MRS representation (isomorphic MRS fragments are string-equivalent)
  - …and could be reified into (possibly multiple) MRSs
- Also I brought up the idea of a *headed walk*
  - `*/EQ` and `RSTR/H` links are inverted, regular arguments are unmodified
  - Makes each node the *semantic head** of its descendants

* this term is not well defined

Recap

- A simple transfer model learned subgraph alignments
  - basic frequencies
  - heuristic-based filtering
- A basic decoder naively constructed target MRSs using aligned subgraphs
  - Search-space issues, even with aggressive beam search
  - Resulting MRSs sometimes needed to be augmented (e.g. setting the `TENSE` property); sometimes still couldn't be used for generation

Developments

- Simplified MrsPaths to singly-rooted DAGs
  - re-introduced node-identifiers (what about string-equivalency?)
  - structurally similar to Abstract Meaning Representation (AMR), so why not use the same (PENMAN) notation?

E.g., from this:

```
_chase_v_1(
  :ARG1/NEQ>_dog_n_1<RSTR/H:udef_q &
  :ARG2/NEQ>_cat_n_1<RSTR/H:udef_q
)
```

To this:

```
(e2 / _chase_v_1
    :ARG1/NEQ (x4 / _dog_n_1
                  :RSTR/H-of (q4 / udef_q))
    :ARG2/NEQ (x6 / _cat_n_1
                  :RSTR/H-of (q6 / udef_q)))
```

Developments

```
(e2 / _chase_v_1
    :ARG1/NEQ (x4 / _dog_n_1
                  :RSTR/H-of (q4 / udef_q))
    :ARG2/NEQ (x6 / _cat_n_1
                  :RSTR/H-of (q6 / udef_q)))
```

- Why does it matter?
  - node identifiers allow for encoding of re-entrantrant structures
  - PENMAN (AMR) format increases approachability for those outside DELPH-IN and makes it easier to share data

Developments

- "Arboreal MRS"
  - Strictly following the headedness edge-inversion does not always yield a spanning graph (presented at NW-NLP 2016)

	ERG (abs)	**ERG (rel)**	**Jacy (abs)**	**Jacy (rel)**
parsed	97.21	-	79.97	-
connected	97.18	99.99	78.20	97.70
sem-headed	96.43	99.20	65.30	81.66

- But by relaxing the strict headed inversions, nearly all parses can be captured by a singly-rooted graph (c.f. Stephan's similar transformation for EDS)

Findings

- The hope for semantics reducing complexity for long-distance dependencies…

```
彼女    は １０ 分     前     に  出かけ た
~~~~                          ~~~~~~
kanojyo wa  10  fun    mae    ni  dekake ta
she     TOP 10  minute before LOC leave  PFV
"She left home 10 minutes ago"

       (e2 / _dekakeru_v_1
           :ARG1/NEQ (x4 / pron) ... )
```

Findings

- ...is countered by increased distance in some abstract semantic constructions

```
彼女    は １０ 分     前     に  出かけ た
           ~~~~~~~~~~~~~~~~~~  ~~~~~~
kanojyo wa  10  fun    mae    ni  dekake ta
she     TOP 10  minute before LOC leave  PFV
"She left home 10 minutes ago"
```

```
(e2 / _dekakeru_v_1
  :ARG1/EQ-of (e24 / _ni_p
    :ARG2/NEQ (x5 / _mae_n
      :ARG1/EQ-of (e20 / compound
        :ARG2/NEQ (x9 / generic_entity
          :ARG1/EQ-of (e18 / unspec_adj
            :ARG1/EQ-of (e17 / degree
              :ARG2/NEQ (x13 / _fun_n_3
                :ARG1/EQ-of (e12 / "card"
                  :CARG> "10"))))
      ...)))))
```

Graph Simplification

"I think what he said is true in a sense."

```
(e2 / _think_v_1
  :ARG1/NEQ (x3 / pron
    :RSTR/H-of (q3 / pronoun_q))
  :ARG2/H (e28 / _true_a_of
    :ARG1/NEQ (x11 / nominalization
      :ARG1/HEQ (e26 / _say_v_1
        :ARG1/NEQ (x22 / pron
          :RSTR/H-of (q22 / pronoun_q))
        :ARG2/NEQ (x17 / thing
          :RSTR/H-of (q17 / which_q)))
      :RSTR/H-of (q11 / udef_q))
    :ARG1/EQ-of (e30 / _in_p
      :ARG2/NEQ (x31 / _sense_n_of
        :RSTR/H-of (q31 / _a_q)))))
```

Graph Simplification

"I think what he said is true in a sense."

```
(e2 / _think_v_1

  :ARG1/NEQ (x3 / pron
    :RSTR/H-of (q3 / pronoun_q))

  :ARG2/H (e28 / _true_a_of

    :ARG1/NEQ (x11 / nominalization
      :ARG1/HEQ (e26 / _say_v_1
        :ARG1/NEQ (x22 / pron
          :RSTR/H-of (q22 / pronoun_q))
        :ARG2/NEQ (x17 / thing
          :RSTR/H-of (q17 / which_q)))
      :RSTR/H-of (q11 / udef_q))

    :ARG1/EQ-of (e30 / _in_p
      :ARG2/NEQ (x31 / _sense_n_of
        :RSTR/H-of (q31 / _a_q)))))
```

Graph Simplification

"I think what he said is true in a sense."

```
(e2 / _think_v_1

  :ARG1/NEQ (x3 / pron)

  :ARG2/H (e28 / _true_a_of

    :ARG1/NEQ (x11 / nominalization
      :ARG1/HEQ (e26 / _say_v_1
        :ARG1/NEQ (x22 / pron)
        :ARG2/NEQ (x17 / thing
          :RSTR/H-of (q17 / which_q))))

    :ARG1/EQ-of (e30 / _in_p
      :ARG2/NEQ (x31 / _sense_n_of
        :RSTR/H-of (q31 / _a_q)))))
```

Graph Simplification

"I think what he said is true in a sense."

```
(e2 / _think_v_1

  :ARG1/NEQ (x3 / pron)

  :ARG2/H (e28 / _true_a_of

    :ARG1/NEQ (x11 / nominalization
      :ARG1/HEQ (e26 / _say_v_1
        :ARG1/NEQ (x22 / pron)
        :ARG2/NEQ (x17 / thing
          :RSTR/H-of (q17 / which_q))))

    :in (x31 / _sense_n_of
        :RSTR/H-of (q31 / _a_q))))
```

Subgraph Extraction

- No problem extracting re-entrant subgraphs when target is contained:

```
(e2 / _try_v_1                    (x3 / named
  :ARG1/NEQ (x3 / named             :CARG> "Kim"
    :CARG> "Kim"                    :RSTR/H-of (q3 / proper_q))
    :RSTR/H-of (q3 / proper_q))
  :ARG2/H (e11 / _sleep_v_1
    :ARG1/NEQ x3))
```

- But when breaking a re-entrancy, there's two main choices: remove or resolve

```
(e11 / _sleep_v_1
  :ARG1/NEQ [X] )
```

```
(e11 / _sleep_v_1
  :ARG1/NEQ (x3 / named
    :CARG> "Kim"
    :RSTR/H-of (q3 / proper_q)))
```

Plans

- Designing and implementing the whole MT pipeline was perhaps too ambitious for an individual Ph.D.
- (1) Data preparation (MRS to singly-rooted DAGs; simplifications)
  - mostly done
- (2) Subgraph alignment and training
  - working, room for improvement
  - STSG?
  - MDL?
  - HRG?
  - (this is where I'd prefer to spend my time)

Plans (subgraph alignment)

- Currently I just align subgraphs:

```
(e8 / _walk_v_1                      (e10 / _aruku_v
  :ARG1/EQ-of (e10 / _along_p_dir       :ARG2/NEQ (x5 / _michi_n_2
    :ARG2/NEQ (x11 / _street_n_1           :RSTR/H-of (q5 / udef_q_rel)))
      :RSTR/H-of (q11 / _the_q))))
```

- But don't align internal locations:

```
(e8 / _walk_v_1                      (e10 / _aruku_v
  :ARG1/EQ-of (e10 / _along_p_dir       :ARG2/NEQ ( [X] ))
    :ARG2/NEQ ( [X] )))
```

Plans

- (3) Decoding
  - proof of concept worked at one point
  - can I rely on tree-based MT tools, like Joshua?
- (4) Finishing (adding variable properties, restoring re-entrancies, etc.)
  - partially done
  - maybe remove the need for generable MRSs by using graph-to-string realization (from Matic Horvat or Yannis Kontas)

Thanks