Fuzzy Syntactic Reordering for Arabic-English PSMT

Jacob Andreas Nizar Habash Owen Rambow

Columbia University

31 July 2011

Introduction

- PSMT performs poorly on language pairs that require long-range reordering
- In particular, distortion penalties cause translation systems to disprefer such reorderings, even when they're syntactically necessary
- The problem is particularly acute for Arabic → English (VSO → SVO)
- A frequent solution is to reorder the sentence before translation using a parser, but Arabic parsers are noisy (Green *et al.* 2009)
- So we will *fuzzify* the parser's subject span before reordering

Related Work

- "Clause Restructuring for SMT" (Collins *et al.* '05)
  - Pre-ordering can improve translation output by capturing long-range reordering behavior a phrase table can't
- "Syntactic Preprocessing for Statistical MT" (Habash '07)
  - Such reordering rules work well for Arabic with phrase-based systems
- "Chunk-based Verb Rerordering in VSO Sentences for Arabic-English SMT" (Bisazza & Federico '10)
  - Fuzzification leades to even greater improvements
- "Improving Arabic-to-English Statistical Machine Translation by Reordering Post-Verbal Subjects for Alignment" (Carpuat *et al.* '10)
  - Alignment-only reordering improves translation quality

Our approach

We build on Carpuat *et al.*'s approach by adding reordering for input sentences as well as training data.

- Use a deep parse (like Collins *et al.*)
- Fuzzify (like Bisazza & Federico)

Rather than performing a hard reordering, we wish to take the subject span identified by the parser and fuzzify it.
In general, we'll think of the fuzzification process as:

1. Generating a set of candidate spans
2. Reordering using each of those spans

We present both a simple fuzzification scheme and an error-driven fuzzification scheme.

Simple fuzzification

For simple “fuzzification”, propose only two versions of the sentence:

- The sentence reordered with the parser’s original subject span
- The unreordered sentence

(We can think of this as a degenerate reordering with a zero-width span.)

Experiments with gold parses

We begin by testing the effects of reordering on the NIST MT05 dataset (the Columbia Arabic Treebank provides gold parses).

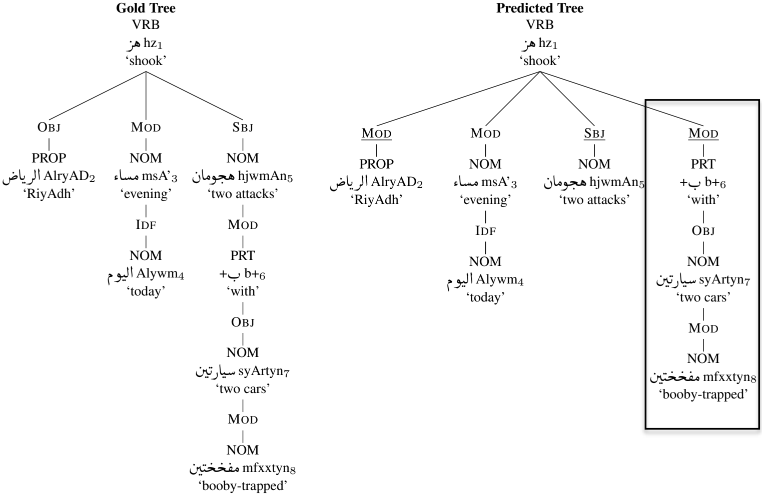
Quasi-upper bound: not strictly guaranteed to outperform all other reordering decisions, but represents best case for any fuzzification algorithm.

- Baseline: 47.13
- Forced reordering: 47.43
- Optional reordering: 47.55

Error-driven fuzzification : Motivation

- We investigate errors in the *subject span* identified by the parser
- We want to know
  1. How often the span is wrong
  2. What goes in the "difference box" when it is

Algorithm motivation



Error-driven fuzzification : Analysis of Parser Error

Matrix VS subject span errors

- 76% correct
- 14% too short
- 10% too long

Number of top-level difference box constituents

- Short span: 56% one constituent (8% of total)
- Long span: 66% one constituent (7% of total)

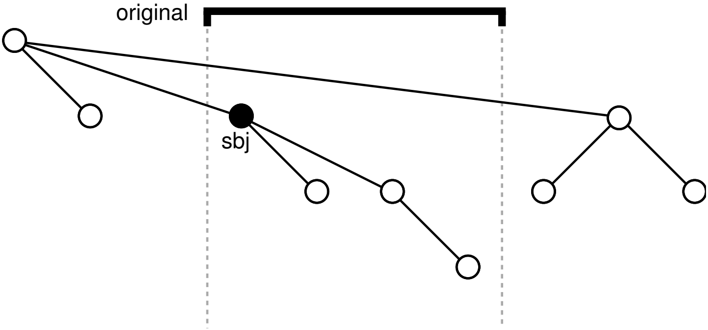
So can correctly recall 91% of subject spans by adding or removing **at most 1** constituent

Fuzzification algorithm

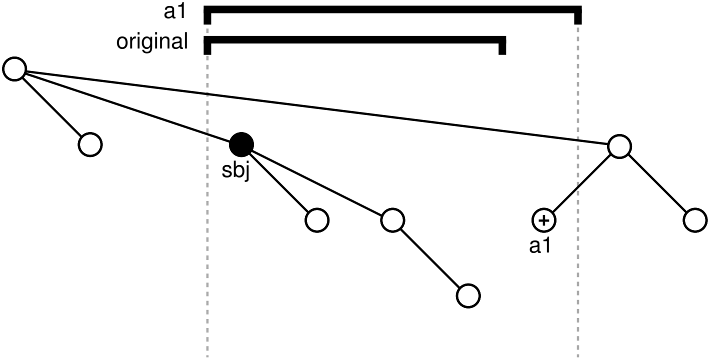
Generate a set of *proposed spans* for reordering, and then select the reordered sentence assigned the highest probability by the translation system.

- Propose the unmodified span
- Propose removal of the smallest rightmost constituent in the span, or any of its ancestors
- Propose addition of the smallest constituent left of the span, or any of its ancestors which do not have in-span children

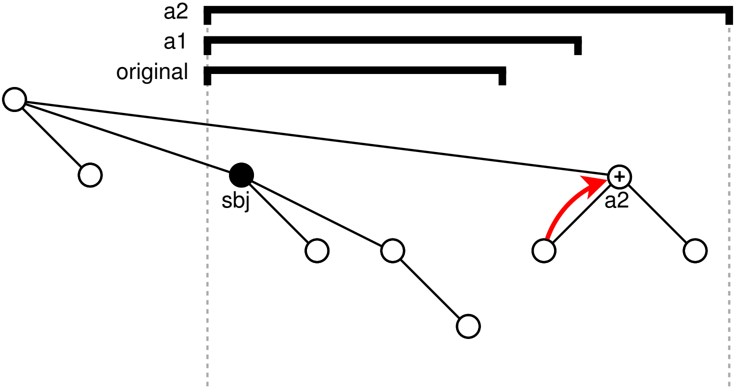
Fuzzification algorithm

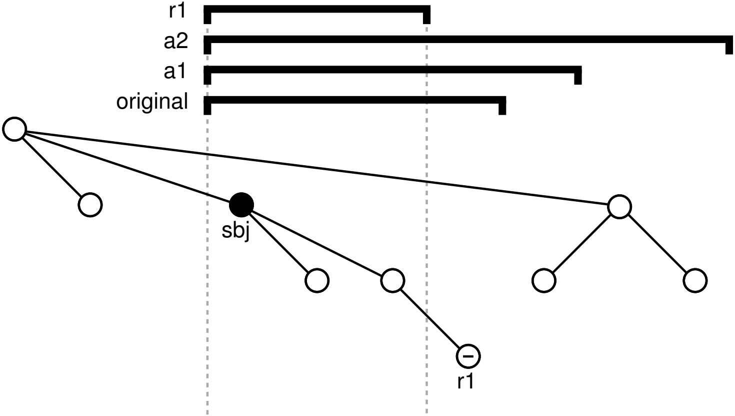


Fuzzification algorithm

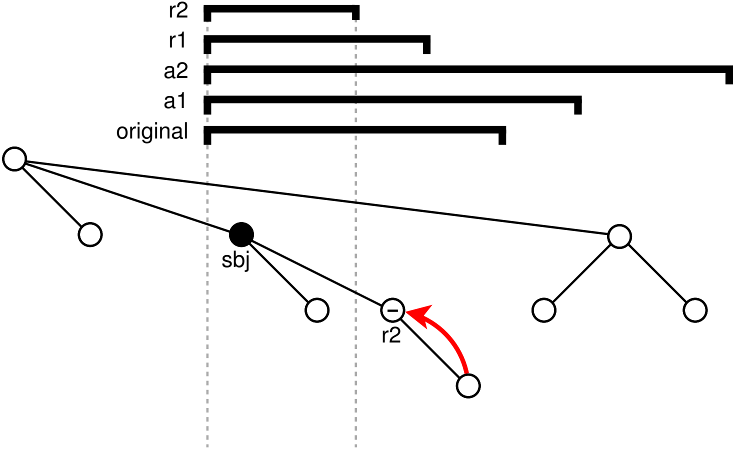


Fuzzification algorithm




r1
a2
a1
original
sbj
r1

Fuzzification algorithm



Automatic evaluation: Setup

1. Moses PSMT toolkit
2. 12M word MSA-English bitext (LDC2007E103) training
3. NIST MT05 test
4. GIZA++-reordered for alignment (as in Carpuat '10)
5. MERT tuned
6. *Same* phrase table & MERT weights for all experiments
7. Arabic dependency parser from Marton *et al.* 2010
8. ATB tokenization with MADA

Automatic evaluation: System overview

1. BASE: The baseline system (no reordering)
2. FORCE: Forced reordering in all cases
3. OPT: optional reordering
4. SPAN: error-driven span correction

Automatic evaluation: Results

System	BLEU	Prec-1g	Prec-4g	METEOR
BASE	47.13	81.91	29.52	53.09
FORCE	47.03	81.78	29.52	53.11
OPT	47.42	81.88	30.04	53.22
SPAN	47.41	81.92	30.03	53.21
GOLD	47.55			

- Forced reordering degrades BLEU score
- As expected, 4-gram precision increases while unigram precision remains approximately the same
- Improvement is observed in both METEOR and BLEU
- Paired bootstrap resampling shows stat. sig. improvement over baseline, but no difference between OPT and SPAN

Manual evaluation : Labeling scheme

- **C**: verb follows subject (as in English), i.e. correct ordering.
  "Recep Tayyip Erdogan announced that Turkey is strong."
- **SO**: subject overlaps with verb.
  "Recep announced Tayyip Erdogan Turkey is strong."
- **SI**: verb precedes subject (as in Arabic).
  "announced Recep Tayyip Erdogan that Turkey is strong."
- **MM**: both verb and subject missing.
  "Turkey is strong."
- **MV**: verb missing.
  "Recep Tayyip Erdogan Turkey is strong."
- **MS**: subject missing.
  "announced that Turkey is strong."

Manual evaluation

System	MM	MS	MV	SI	SO	C	M*	S*	C
BASE	8	13	11	9	3	53	33	12	53
OPT	7	11	10	5	5	61	28	10	61
SPAN	8	10	09	5	2	64	27	7	64

- Both OPT and SPAN decrease the number of missing and out-of-order subjects, and increase the number of correct subjects
- SPAN may do better than OPT ($p < .3$); both definitely outperform the baseline (using McNemar's test)

Conclusions

- We have presented an approach for improving Arabic-English PSMT using syntactic information from a noisy parser
- Quality goes down with forced reordering, but improves with either optional reordering or subject span fuzzification
- Both techniques achieve nearly 3/4 of the maximum possible gain achieved with gold parses
- Manual evaluation shows results consistent with BLEU, and indicates small improvements gained with span fuzzification

Future Work

- Use machine learning to the lattice of possible reorderings
- Try to learn reorderings on all verbs, not just matrix

Questions?