A filter for syntactically comparable parallel sentences

Martin Kroon Sjef Barbiers
{m.s.kroon,l.c.j.barbiers}@hum.leidenuniv.nl
Leiden University Centre for Linguistics, The Netherlands

Universiteit Leiden

Introduction and Motivation

- Filter out parallel sentences that are **syntactically too different**
  - Free translations
  - Idioms, proverbs, etc.
- Enhance and speed up comparative syntax

Data

- Corpus of **parallel sentences** (English-Dutch), labelled `Y` or `N`
- From Europarl (Koehn, 2005)
- Currently 50 `Y`, 77 `N`

E.g. `Y`: You$_{\text{PRON}}$ spoke$_{\text{VERB}}$ about$_{\text{ADP}}$ direct$_{\text{ADJ}}$ negotiations$_{\text{NOUN}}$.$_{\text{PUNCT}}$

U$_{\text{PRON}}$ sprak$_{\text{VERB}}$ over$_{\text{ADP}}$ rechtstreekse$_{\text{ADJ}}$ onderhandelingen$_{\text{NOUN}}$.$_{\text{PUNCT}}$

E.g. `N`: That$_{\text{DET}}$ is$_{\text{VERB}}$ what$_{\text{PRON}}$ will$_{\text{VERB}}$ make$_{\text{VERB}}$ us$_{\text{PRON}}$ strong$_{\text{ADJ}}$.$_{\text{PUNCT}}$

Dan$_{\text{ADV}}$ zijn$_{\text{VERB}}$ wij$_{\text{PRON}}$ sterk$_{\text{ADV}}$!$_{\text{PUNCT}}$

Levenshtein on POS-tags

Baseline

- **UD** POS-tags
- Options:
  - Remove stopwords / function words

Sentence Vectors

- Train `fastText` (Bojanowski, Grave, Joulin, & Mikolov, 2016) vector space models on Europarl corpus
- Calculate **transformation matrix** to translate between vector spaces (Mikolov, Le, & Sutskever, 2013)
- Measure is **cosine similarity** between sentence vectors

$$\vec{S} = \frac{1}{|S|} \sum_{w \in S} \vec{w}$$

- Options:
  - Remove stopwords / function words
  - Weight word vectors with tf-idf
  - Geometric mean
- Drawbacks: computationally expensive; requires big corpus for vector space

Tree Edit Distance

- Draw UD parses **as trees**
- Calculate **tree edit distance** à la Zhang and Shasha (1989)
- Options:
  - Remove stopwords / function words
  - Morphological features as nodes in tree
- Drawbacks: needs UD models; repeating differences; for ordered trees

Sentence Length Ratio

- Computationally cheap

$$d(S_1, S_2) = \frac{\max(|S_1|, |S_2|)}{\min(|S_1|, |S_2|)}$$

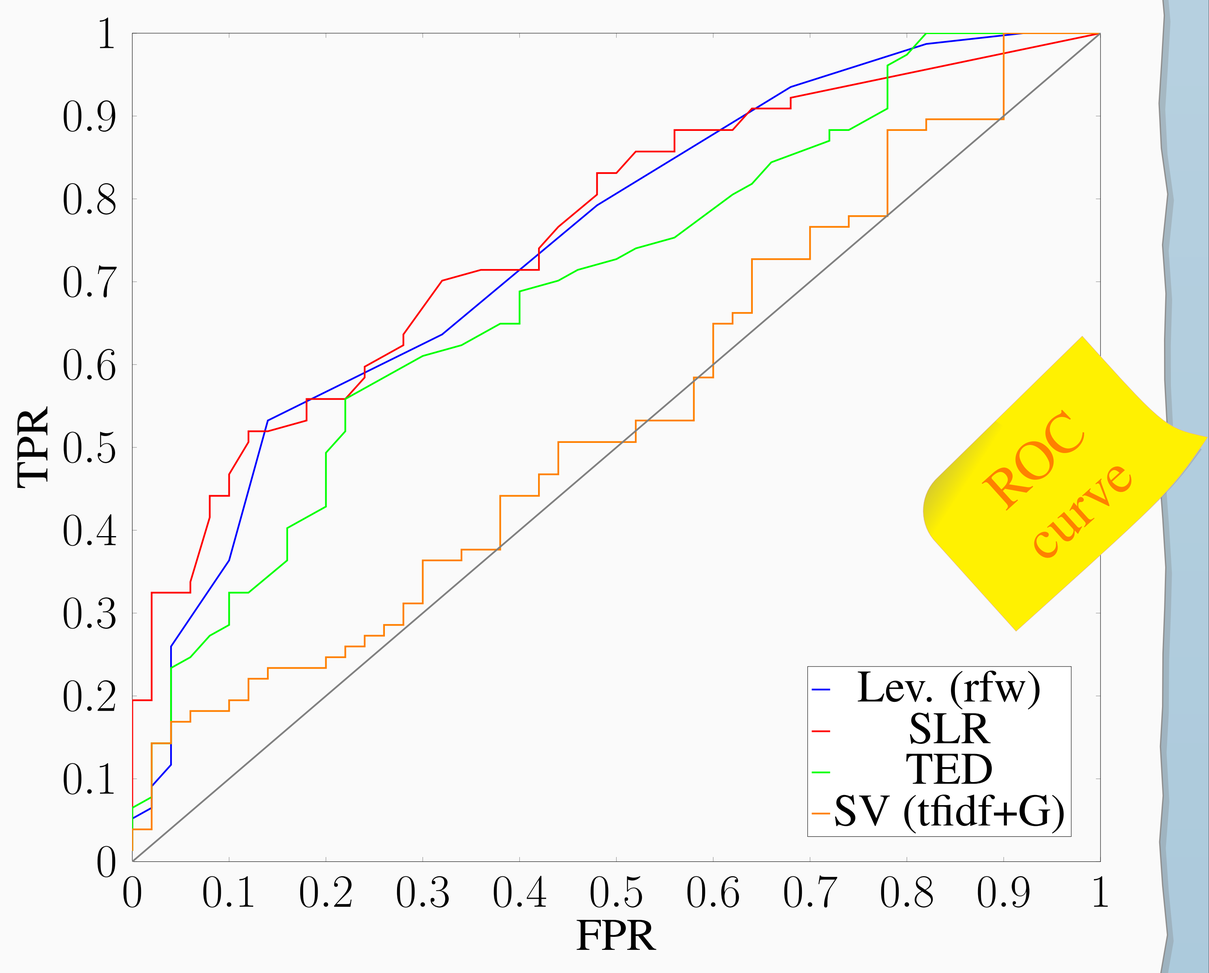
- Options:
  - Remove stopwords / function words
- Drawbacks: not fine-grained for syntax; threshold specific for language pair

Main Idea

- Levenshtein on POS-tags very sensitive to phrases switching around
  - SVO vs. SOV will heavily influence Levenshtein distance, especially with longer objects and verb phrases
- Something different ought to work better...

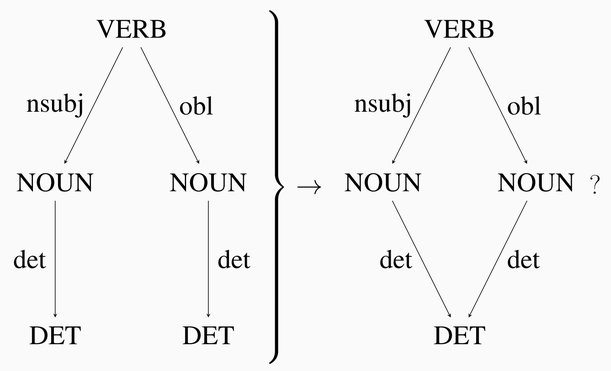
Preliminary Results

- Results for now suggest:
  - Sentence length ratio without removal of stop- and function words works best
  - Tree edit distance worse than Levenshtein distance
  - Sentence vectors don't seem to work: barely work better than chance
  - Levenshtein distance with removal of stop- function words is second best...



Future

- tf-idf for every method?
- Tree to DAGs, repeating differences?
- Other GED for unordered trees/DAGs
- Combinations! Parallel and sequential
- Bigger corpus! More languages!



References

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Mt summit* (Vol. 5, pp. 79–86).

Mikolov, T., Le, Q. V., & Sutskever, I. (2013). Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Zhang, K. & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing*, *18*(6), 1245–1262.

Presented at the 28th Meeting of Computational Linguistics in The Netherlands (CLIN 28), Nijmegen, January 26, 2018