Adapted TextRank for Term Extraction: A Generic Method of Improving Automatic Term Extraction Algorithms

Ziqi Zhang¹, Johann Petrak², Diana Maynard²
ziqi.zhang@sheffield.ac.uk, johann.petrak@sheffield.ac.uk, d.maynard@sheffield.ac.uk

1. Information School, The University of Sheffield, UK
2. Department of Computer Science, The University of Sheffield, UK
The Task of ATE

- **Input:** (reasonably large) domain specific, focused corpus
- **Output:** list terms from the corpus, representing the domain
- **Approach**
  - Candidate extraction: domain-dependent, usually noun phrases, n-grams, or sequence matched by PoS patterns
  - **Candidate ranking & selection:** scoring candidates based on corpus statistics, selection by threshold, or machine learning

---

**Domain specific corpus**

**ATE**

- Candidate Extraction
- Candidate Ranking, Selection

**Terms for the corpus**

[ semantic, 0.67, ontology, 0.34, nlp, 0.33, text mining, 0.12, ... web page, 0.012 ]
The Task of ATE

- **A classic text mining problem**
  - Dating back to 1990s (Bourigault 1992)
  - To date still an active area of research

- **A fundamental step to many complex tasks**
  - Ontology engineering
  - Dictionary, terminology construction
  - Information Retrieval
  - Translation
  - ...

- **Context of this work: KNOWMAK** ([https://www.knowmak.eu/](https://www.knowmak.eu/))
# The Task of ATE

## Differentiation from related tasks

<table>
<thead>
<tr>
<th>Keyword Extraction</th>
<th>ATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>- document specific</td>
<td>- domain specific</td>
</tr>
<tr>
<td>- only a handful</td>
<td>- # depends on corpus</td>
</tr>
<tr>
<td>- mainly for indexing</td>
<td>- mainly knowledge acquisition</td>
</tr>
</tbody>
</table>

| NER | |
| - usually real world named entities | - domain specific terms |
| - sentence context is more important | - corpus level statistics are more important |
| - semantic typing | - no typing |

Source: [https://imanage.com/blog/named-entity-recognition-ravn-part-1/](https://imanage.com/blog/named-entity-recognition-ravn-part-1/)
Motivation and Contribution

● ATE still an unsolved problem
  ■ No ‘all-rounder’ method
  ■ Performance always depends on data and domain
  ■ ‘one-size-fits-all’ solution feasible?

● ATE methods are predominantly unsupervised
  ■ For many domains there are already domain specific resources potentially useful, e.g., unlabelled corpus, pre-compiled named entity lists, partial ontologies, etc
  ■ Can we benefit from those?
Motivation and Contribution

- ATE still an unsolved problem
  - No ‘all-rounder’ method
  - Performance always depends on data and domain
  - ‘one-size-fits-all’ solution feasible?

- ATE methods are predominantly unsupervised
  - For many domains there are already domain specific resources potentially useful, e.g., unlabelled corpus, pre-compiled named entity lists, partial ontologies, etc
  - Can we benefit from those?

A generic method that employs semantic relatedness to a set of **domain specific seed words** to potentially **improve any ATE** algorithms (by up to 25 percentage points in average precision in experiments).
AdaText - Overview

Adapted TextRank for Automatic Term Extraction

- Domain specific seed words/phrases
- Semantic relatedness
- Domain specific corpus
- Extract words
- ATE (any algorithm)
- [ $t_1 = 1.99$, $t_2 = 1.21$, $t_3 = 1.10$, ... ]
- Filter by threshold
- [ $w_1 = 0.67$, $w_2 = 0.34$, $w_3 = 0.22$, ... ]
- TextRank
- Re-rank
- [ $t_1 = 2.19$, $t_3 = 1.41$, $t_2 = 1.29$, ... ]
Adapted TextRank for Automatic Term Extraction

SEEDING

Domain specific seed words/phrases → Extract words → Semantic relatedness → Filter by threshold

CORPUS LEVEL TEXTRANK

ATE (any algorithm) + Re-rank

Combining with ATE

\[
\begin{align*}
\text{TextRank: } & t_1 = 1.99, t_2 = 1.21, t_3 = 1.10, \ldots \\
\text{ATE: } & w_1 = 0.67, w_2 = 0.34, w_3 = 0.22, \ldots \\
\text{Re-rank: } & t_1 = 2.19, t_3 = 1.41, t_2 = 1.29, \ldots
\end{align*}
\]
AdaText - Seeding

**Input**
- $C$ - the target corpus from which terms are extracted
- $S$ - a set of ‘seed’ word/phrases representing the domain
  - taken from existing domain lexicons, or generated in an unsupervised way from available corpora
  - May not contain real terms from $C$

**Process**
- Extract words from $C$, as $W$
- Compute pairwise semantic relatedness for $S \times W$
  - Cosine similarity using GloVe embedding vectors
  - OOV ignored, phrase based on compositional averaging (Iyyer et al. 2015)

**Output**
- $W_{sub}$ a subset of $W$, satisfying relatedness > $min$
  Intuitively, they are more ‘relevant’ to the domain
AdaText - Corpus Level TextRank

**Input**
- $C$ - the target corpus from which terms are extracted
- $W_{sub}$ - the subset of words selected before

**Process**
- Apply TextRank to the graph created for $W_{sub}$ to compute a TextRank ($tr$) score of every word $w$ in $W_{sub}$
- Traditional TextRank (Mihalcea et al., 2004) is a PageRank process to a graph of words from each document, where an edge is created if words co-occur in a context window of $win$
AdaText - Corpus Level TextRank

**Input**
- **C** - the target corpus from which terms are extracted
- **$W_{sub}$** - the subset of words selected before

**Process**
- Apply TextRank to the graph created for $W_{sub}$ to compute a TextRank ($tr$) score of every word $w$ in $W_{sub}$
- Here it is adapted in two ways
  - A graph of words from the entire corpus
  - An edge is created if two words appear within **win anywhere in the corpus** (in any document)

**Output**
- $tr$ scores for every word $w$ in $W_{sub}$
AdaText - Combining with ATE

**Input**
- $C$ - the target corpus from which terms are extracted
- $ATE$ - some ATE algorithm
- $tr$ scores for every word $w$ in $W_{sub}$

**Process**
- Apply $ATE$ to $C$ to extract and score candidate terms
- Revise each candidate term’s score using $tr$ scores for its composing words

\[
\text{score}(t_i) = (1.0 + \frac{\sum_{w_i \in \text{words}(t_i)} tr(w_i)}{|\text{words}(t_i)|}) \times \text{ate}(t_i)
\]

- Then re-rank candidate terms by the new score

**Output**
- Re-ranked list of candidate terms
Experiment and Findings

- **Base ATE methods** (as AdaText needs ATE scores of candidate terms)
  - Modified TFIDF (Zhang et al., 2016)
  - CValue (Ananiadou 1994)
  - Basic (Bordea et al., 2013)
  - RAKE (Rose et al., 2010)
  - Weirdness (Ahmad et al., 1999)
  - LinkProbability (LP, Astrakhantsev, 2016)
  - $X^2$ (Matsuo et al., 2003)
  - GlossEx (Park et al., 2002)
  - Positive Unlabelled (PU) learning (Astrakhantsev, 2016)
  - AvgRel - average relatedness score with seeds

- Use implementations:
  - JATE ([https://github.com/ziqizhang/jate](https://github.com/ziqizhang/jate))
  - ATR4S ([https://github.com/isprats/atr4s](https://github.com/isprats/atr4s))
Experiment and Findings

Evaluation measures
- Precision for top $K$ ranked candidate terms
- $K = \{50, 100, 500, 1000, 2000\}$
- Average P@K for all five $K$’s
Datasets

- GENIA
  - 2,000 semantically annotated Medline abstracts
  - 434k words
  - 33k target terms

- ACLv2
  - 300 ACL paper abstracts
  - 32k words
  - 3k target terms
Experiment and Findings

Seeds and parameters

- For GENIA:
  5,502 named entities from the BioNLP Shared Task 2011, only 25 match candidate terms

- For ACLv2:
  1,301 noun phrases from the titles of ACL, NAACL, and EACL papers (since 2000), none matches candidate terms

- Semantic relatedness threshold $min=0.5$ to 0.85 with 0.05 increment (selects for GENIA/ACL ~ 50/70 % ... 10/5 %)

- TextRank context window $win=5, 10$
- Base ATE performance varies significantly depending on datasets.
- No single, consistently winning method on all five $K$’s.
- E.g., PU is the best performing in $\text{AvgP}@[K]$ on the ACL corpus, but the fourth worst performing on the GENIA corpus.
- The min threshold: too low (creating lots of isolated graphs) or too high (including too many weakly related words) can harm performance
- The win threshold: no strong pattern as to which (5 or 10) is better
- Within \( \text{min} = [0.6, 0.75] \), AvgP@K improvement by 1 ~ 25 percentage points depending on the base ATE, and dataset
Conclusion

● **The takeaway message**
  ■ There is probably never a ‘one-size-fit-all’ ATE method, instead, think about improving existing ones
  ■ AdaText makes use of existing domain resources and builds on the TextRank algorithm
  ■ Generic method able to improve, potentially, any ATE method

● **Future work**
  ■ Whether and how the size and source of the seed lexicon affects performance
  ■ Adapt TextRank to a graph of both words and phrases, and see how this affects results
Resources and Software

● **Data**
  ■ Genia corpus, ACL corpus available
  ■ Glove embeddings available

● **Software**
  ■ JATE ([https://github.com/ziqizhang/jate](https://github.com/ziqizhang/jate))
  ■ ATR4S ([https://github.com/ispras/atr4s](https://github.com/ispras/atr4s))
  ■ Code for this work: [https://github.com/ziqizhang/texpr](https://github.com/ziqizhang/texpr)

● **Slides**
  ■ [https://goo.gl/1sPuhg](https://goo.gl/1sPuhg)
Acknowledgements

This work is supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 726992 (KNOWMAK project)

https://www.knowmak.eu/