Conundrums in Unsupervised Keyphrase Extraction: Making Sense of the State-of-the-Art
Kazi Saidul Hasan and Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas

Unsupervised keyphrase extraction
- Identify the phrases that represent a given document in an unsupervised manner

Many approaches were recently developed
- Language modeling (e.g., Tomokio & Hurst, 2003)
- Graph-based ranking (e.g., Mihalcea & Tarau, 2004)
- Clustering (e.g., Liu et al., 2009)

But ... each approach has only been evaluated on a particular domain of text
→ it is not clear how effective state-of-the-art unsupervised keyphrase extractors will be on datasets from a new domain

Goal: Systematically compare five unsupervised keyphrase (KP) extractors on four corpora with varying domains and statistical characteristics

<table>
<thead>
<tr>
<th>Type</th>
<th>Corpus</th>
<th>DUC-2001</th>
<th>Inspec</th>
<th>NUS</th>
<th>ICSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUC-2001</td>
<td>News articles</td>
<td>308</td>
<td>500</td>
<td>213</td>
<td>161</td>
</tr>
<tr>
<td># Documents</td>
<td>Paper abstracts</td>
<td>876</td>
<td>134</td>
<td>829</td>
<td>1611</td>
</tr>
<tr>
<td># Candidate words/Document</td>
<td>Full papers</td>
<td>312</td>
<td>57</td>
<td>327</td>
<td>453</td>
</tr>
<tr>
<td># Candidate phrases/Document</td>
<td>Meeting transcripts</td>
<td>207</td>
<td>34</td>
<td>202</td>
<td>296</td>
</tr>
<tr>
<td># Tokens/Document</td>
<td># Gold keyphrases</td>
<td>1.5</td>
<td>1.7</td>
<td>1.6</td>
<td>1.5</td>
</tr>
<tr>
<td>@Gold keyphrases/Document</td>
<td>@Gold keyphrase</td>
<td>2.484</td>
<td>491</td>
<td>237</td>
<td>582</td>
</tr>
<tr>
<td>@Gold keyphrases/Document</td>
<td>@Gold keyphrase</td>
<td>8.1</td>
<td>9.3</td>
<td>11.0</td>
<td>3.0</td>
</tr>
<tr>
<td>@Gold keyphrases/Document</td>
<td>@Gold keyphrase</td>
<td>1760/184</td>
<td>1353/259</td>
<td>2750/197</td>
<td>0.82/2.1</td>
</tr>
<tr>
<td># Token/Gold keyphrase</td>
<td># Token/Gold keyphrase</td>
<td>2.1</td>
<td>2.3</td>
<td>2.3</td>
<td>1.3</td>
</tr>
</tbody>
</table>

Five state-of-the-art unsupervised KP extractors

**TF-Idf**
1. Compute the TF-Idf value of each noun and adjective
2. Extract all longest $n$-grams consisting of nouns and adjectives
3. Score each $n$-gram by summing the TF-Idf score of each word
4. Output the $m$ highest-scored $n$-grams as keyphrases

**TextRank** (Mihalcea & Tarau, 2004): graph-based ranking
- Represent a text as a graph. One vertex per word; an edge with weight 1 connects 2 words if they co-occur in a window of 2.
- Goal: compute the score of each vertex iteratively, using:

$$s(v_i) = (1 - d) + d \times \sum_{j \in ad(j_i)} \frac{n(v_j)}{\sum_{k \in ad(j_k)} n(k_j)} s(v_j)$$

where $d=0.85$, and $Ad(v_i)$ is the set of neighbors of $v_i$. (Intuition: $v_i$’s score is high if it has many high-scored neighbors)
- Select 7% top-scored vertices as keywords
- Keyphrases are formed by combining adjacent keywords

**SingleRank** (Wan & Xiao, 2008): graph-based ranking
- Same as TextRank, except that: (1) a co-occurrence window of 10 is used; (2) edge weight is the number of times the two words co-occur; (3) score each $n$-gram (consisting of nouns and adjectives) by summing the vertex score of each of its words; and (4) output the $N$ highest-scored $n$-grams as keyphrases

**ExpandRank** (Wan & Xiao, 2008): graph-based ranking
- Same as SingleRank, except that co-occurrence statistics are computed using not only the text under consideration but also its 5 nearest neighbors (according to cosine similarity). Nearer neighbors have a stronger influence on the co-occurrence stats.

**KeyCluster** (Liu et al., 2009): clustering-based approach
1. Remove stopwords and treat all remaining unigrams as candidates
2. Compute word similarity matrix based on frequency of co-occurrence
3. Cluster the words using spectral clustering into $m$ clusters
4. Take a representative word (i.e., the exemplar) from each cluster
5. Extract all longest $n$-grams consisting of adjectives followed by nouns
6. Any $n$-gram with an exemplar term is selected as a keyphrase
7. Filter the frequent unigrams using a wordlist generated from the Wiki

Experimental setup
Results in terms of recall-precision curves. To generate a curve:
- for TF-Idf, SingleRank, ExpandRank: vary number of keyphrases
- for TextRank: vary $T$ (the percentage of top-scored vertices)
- for KeyCluster: vary $m$ (the number of clusters)

Observations:
- TF-Idf is a strong baseline, offering very robust performance across different datasets
- TextRank is sensitive to $T$ on Inspec, but not on the other datasets
- ExpandRank’s success largely depends on topic-wise similarity among the nearest documents
- KeyCluster works better if more clusters are used
- TF-Idf outputs fewer keyphrases, in comparison to SingleRank and ExpandRank, to achieve its best F-score on most datasets
- Scores on Inspec are higher since it has the lowest number of candidate phrases in each document among all datasets
- It is essential to evaluate a KP extractor on multiple datasets to fully understand its strengths and weaknesses
- Post-processing steps (e.g., forming keyphrases) can have a large impact on the performance of a KP extractor