A graph-based unsupervised N-gram filtration technique for automatic keyphrase extraction

by

Niraj Kumar, Kannan Srinathan, Vasudeva Varma

in

Int. J. Data Mining, Modelling and Management

Report No: IIIT/TR/2014/-1

Centre for Search and Information Extraction Lab
International Institute of Information Technology
Hyderabad - 500 032, INDIA
August 2014
A graph-based unsupervised N-gram filtration technique for automatic keyphrase extraction

Niraj Kumar*, Kannan Srinathan and Vasudeva Varma

IIIT-Hyderabad,
Hyderabad-500032, India
E-mail: niraj_kumar@research.iiit.ac.in
E-mail: srinathan@iiit.ac.in
E-mail: vv@iiit.ac.in
*Corresponding author

Abstract: In this paper, we present a novel N-gram (N>=1) filtration technique for keyphrase extraction. To filter the sophisticated candidate keyphrases (N-grams), we introduce the combine use of: 1) statistical feature (obtained by using weighted betweenness centrality scores of words, which is generally used to identify the border nodes/edges in community detection techniques); 2) co-location strength (calculated by using nearest neighbour Dbpedia texts). We also introduce the use of N-gram (N>=1) graph, which reduces the bias effect of lower length N-grams in the ranking process and preserves the semantics of words (phraseness), based upon local context. To capture the theme of the document and to reduce the effect of noisy terms in the ranking process, we apply an information theoretic framework for key-player detection on the proposed N-gram graph. Our experimental results show that the devised system performs better than the current state-of-the-art unsupervised systems and comparable/better than supervised systems.

Keywords: keyphrase extraction; weighted betweenness centrality; N-gram graph; normalised pointwise mutual information; NPMI.

Reference to this paper should be made as follows: Kumar, N., Srinathan, K. and Varma, V. (xxxx) ‘A graph-based unsupervised N-gram filtration technique for automatic keyphrase extraction’, Int. J. Data Mining, Modelling and Management, Vol. X, No. Y, pp.000–000.

Biographical notes: Niraj Kumar is a PhD research scholar from IIIT-Hyderabad. His areas of interests are: text mining, data mining, web mining, social networking and machine learning. He has worked in several core areas text mining like: keyphrase extraction, document summarisation, summarisation evaluation, automatic evaluation of descriptive answers, automatic question answering, plagiarism detection and recommendation systems.

Kannan Srinathan is an Assistant Professor of Computer Science and Engineering Department at IIIT-Hyderabad, Hyderabad, India. He is associated with the security theory and algorithmic research lab ‘CSTAR’ of IIIT-Hyderabad. His areas of interests are (including but not limited to) cryptography, security, graph theory and distributed computing.
1 Introduction

Keyphrases are sequences of words, which represent the main topics or theme of the document. It also represents a highly condensed summary of the given text and very helpful in effective organisation and retrieval of the huge volumes of text (Turney, 1999; Nguyen and Kan, 2007).

The fast development of the internet has speed up the growth of digital textual information and it is also well known that the language evolves faster than our ability to formalise and catalogue it. A lot of automatic and unsupervised keyphrase extraction algorithms have been proposed to catalogue, organise and boost the information retrieval tasks on electronic texts. Most of these techniques divide the keyphrase extraction task into two sub tasks, i.e.,

1. identification of candidate keyphrases
2. extracting ranked list of keyphrases (Kumar and Srinathan, 2008).

According to Kumar and Srinathan (2008), Hasan and Ng (2010), the variation in the length of documents and bottleneck of statistical and linguistic features is a major hurdle in the identification of candidate keyphrases. Next, the bias behaviour of highly frequent lower length N-grams over higher length N-grams, in the ranking of the identified candidate keyphrases also affects the quality of the result. Kumar and Srinathan (2008), use different strategies to treats N-grams having different length, but, it makes the system more dependent upon the size of N-grams. Hence, it creates the research requirement for a more dynamic system.

To solve these issues, i.e., to achieve the goal of identification of correct candidate keyphrases and to effectively rank the identified candidate keyphrases, we make the following contributions:

1. We introduce a novel N-gram (N >= 1) filtration technique, which uses
   a. statistical feature (i.e., weighted betweenness centrality scores of words, calculated on word graph of text)
   b. semantic expansion strategies (achieved by using co-location strength, calculated by using nearest Dbpedia texts).

   We also use Wikipedia anchor texts, to remove noisy unigrams.

2. We introduce an efficient way to extract the nearest neighbour or contextually similar Dbpedia extended abstracts. It is used in the calculation of co-location strength (which is actually, normalised semantic relatedness strength) of any given
word pairs. This step gives a good score to bigrams, which have strong semantic bindings.

3 To reduce the biasness of lower length N-grams in the ranking of the identified candidate keyphrases and to preserve the local context, based on the semantics of words, we introduce the N-gram (N \geq 1) graph of text. It uses each distinct candidate keyphrase (N-gram) as a node of the graph.

4 To capture the information coverage/reachability of candidate N-grams in semantically reach N-gram graph and to reduce the effect of noisy terms, we introduce the use of the information theoretic model for candidate key player detection (Ortiz-Arroyo, 2010).

*Paper organisation:* in Section 2, we present the motivation behind candidate keyphrase identification and ranking. In Section 3, we briefly describe the related works of this area. In Section 4, we present our, candidate keyphrase identification technique. In Section 5, we discuss the algorithm for ranking of the identified candidate keyphrases. Section 6 represents the pseudo code for the keyphrase extraction system. Section 7 represents the experimental evaluation of the devised system.

2 Motivation

2.1 Phrase identification

According to Kumar and Srinathan (2008), and Liu et al. (2009), majority of candidate keyphrases lies inside the two stopwords or punctuation marks (i.e., called a tentative phrase boundary). But, all such sequences are not as useful as candidate keyphrases. As, these word sequences may contain additional noisy entry(s). To identify the candidate keyphrase inside the tentative phrase boundary, we use an additional boundary-word information (obtained by using weighted betweenness centrality) and the actual span of candidate keyphrase [obtained by using normalised pointwise mutual information (NPMI) score calculated on nearest neighbour Dbpedia extended abstracts].

1 *Weighted betweenness centrality scores of words:* The betweenness centrality describes the frequencies of nodes in the shortest paths between indirectly connected nodes. It is generally used to find the edge between two communities in complex networks. Xie (2005) initially utilised the use of betweenness centrality measure to predict the noun terms at sentence and document level. However, his intention was to identify the insightful terms (i.e., identification of the most relevant terms, which connects different topics or groups in the network) and use those terms in summarisation. We use weighted betweenness centrality score of words to identify the common or between-word in the given word sequence inside the tentative phrase boundary. This common or between-word may be

   a noisy word attached with given candidate N-gram

   b boundary word for the sophisticated candidate N-gram(s).

We use it as an additional tentative boundary word for the given sequence.
Using normalised point wise mutual information (NPMI) score calculated on nearest neighbour Dbpedia extended abstracts: Bouma (2009) successfully used the NPMI score in calculation of co-location strength of word pairs. The value of NPMI lies in the range of \([-1, 1]\). We believe that words related to genuine candidate keyphrase shows relatively different co-location strength with

a. noisy words attached to it
b. word related to different candidate N-grams.

However, to get more effective co-location strength, we concentrate on nearest neighbour/contextually similar Dbpedia extended abstracts. This step is inspired by Wan and Xiao’s (2008) use of nearest neighbour documents for improvements in keyphrase extraction.

Proposed improvements: To achieve this goal, we use the Wikipedia anchor text communities. According to Kumar et al. (2010), anchor texts have great semantic value and thus, each community represents the set of anchor texts related to similar concepts. By using Wikipedia anchor text communities and title of Dbpedia extended abstracts, we categorise all Dbpedia extended abstracts. We use the categorised Dbpedia extended abstracts in the calculation of NPMI scores of bigrams. We use LUCENE-based offline system for pre indexing of each Dbpedia document’s category and then apply a dynamic merging of indexes (if required). This arrangement improves the speed of calculation.

Using Wikipedia anchor text: Actually, each link in Wikipedia is associated with an anchor text, which can be regarded as a descriptor of its target article. Thus, unigrams, which exactly match with Wikipedia anchor texts, contain useful information. We use it in the extraction of useful unigrams.

2.2 Extracting ranked list of keyphrases

To reduce the biasness of lower length N-grams, having a higher occurrence frequency we propose the use of the N-gram graph of text. Next, to effectively utilise the reachability of candidate keyphrases (sophisticated key N-grams), and to reduce the effect of noisy N-grams in ranking, we introduce the use of information theoretic model for candidate key player detection (Ortiz-Arroyo, 2010).

The proposed N-gram graph treats each identified candidate N-gram as node of the graph. Next, it utilises the co-location strength of each word inside the node/N-gram, in the calculation of link weight. This scheme slightly neutralises the effect of highly frequent lower length N-grams in the calculation of link weight. As, it is clear that higher N-grams (\(N > 1\)) contain more number of words, while lower N-grams contain relatively high frequency. If, we include the product of

a. summation of semantic/co-location strength of each word pair of adjacent N-grams
b. co-occurrence frequency of adjacent N-grams, in the calculation of link weight.

Then, each word pair of the adjacent N-grams will participate in the calculation of link weight. This scheme not only neutralises/balances the effect of relatively higher occurrence frequency of lower length N-grams, but also includes the semantic binding of all words, in the adjacent candidate N-grams. This scheme also serves the local context of
the identified candidate phrases/N-grams (N >= 1) and this is especially beneficial, when the semantics of words depend upon its locality. For example, ‘Taj Mahal Hotel’ and ‘Taj Mahal Tea’ do not have anything to do with ‘Taj Mahal’ (i.e., 7th wonder).

3 Related work

A lot of methods have been proposed for automatic keyphrase extraction. The most frequently used steps among all proposed methods are the

1 identification of candidate keyphrases
2 extracting the ranked list of keyphrases:
   a Identification of candidate keyphrases: This is generally achieved by detecting nouns or noun phrases mentioned in the document. The majority of the methods uses N-grams (Hulth, 2003; Tomokiyo and Hurst, 2003) or POS sequences (Turney, 1999; Nguyen and Kan, 2007), or both (Grineva et al., 2009). The most of the graph-based system uses a keyphrase formation step as the final step, where, the ranked list of candidate words is used to form keyphrases (Mihalcea and Tarau, 2004; Wan and Xiao, 2008). Kumar and Srinathan (2008) applied statistical features to select candidate keyphrases. Medelyan and Witten (2005, 2006) use controlled indexing based approach. In this approach keyphrases are chosen from a controlled vocabulary (a dictionary, thesaurus, or a list of terms).
   b Extracting a ranked list of keyphrases: Most of the existing algorithms use different statistical, lexical and/or semantic features to rank the identified candidate keyphrases. For example, Kumar and Srinathan (2008) uses the combination of statistical and limited linguistic features to rank the identified candidate keyphrases. Some use graphs and semantic networks to rank candidates (Mihalcea and Tarau, 2004; Liu et al., 2009; Li et al., 2010). Some techniques depends upon knowledge base. For example, Nguyen and Phan (2009) use ontology and Li et al. (2010) use semantic relatedness score calculated by using Wikipedia or other knowledge base for ranking. Additionally, some approaches use Wikipedia to create a semantic graph for ranking (Tsatsaronis et al., 2010; Li and Li, 2011).

However, none of them claim to solve all the issues at a time, like:

1 handling the issues related to the difference in the length of documents
2 bottleneck of statistical approaches for candidate keyphrase identification
3 the biased behaviour of lower length frequent N-grams over higher length N-grams, in the ranking process.
4 Candidate keyphrase identification

4.1 Preprocessing and tentative phrase boundary detection

The main aim of this stage is to filter sentences and prepare the tentative phrase boundary. To identify the tentative phrase boundary, we use stopwords and punctuation marks (Kumar and Srinathan, 2008). Due to the importance of stopwords ‘of’ and ‘and’ in keyphrase, these are not replaced. For the simplicity of calculation, we temporarily remove all occurrences of both stopwords i.e., ‘of’ and ‘and’ and keep a record of their occurrences. For example, in the case of ‘Theory of computation’, we consider just ‘Theory computations’ for calculation purpose. Finally, we stem the text by using porter stemming algorithm. We also introduce the use of semantic relatedness score of bigrams for more refinement (Subsection 4.3) in the tentative candidate keyphrase boundary.

4.2 Calculating NPMI score of bigrams

To calculate the \(NPMI\) (Bouma, 2009) score of word pairs (bigrams) of the given candidate document, we use the nearest neighbour Dbpedia extended abstracts. Actually, Dbpedia extended abstracts are extended summary of corresponding Wikipedia articles, so, we use Wikipedia link structure (anchor text-based) for faster calculation of the nearest neighbour Dbpedia extended abstracts. The following contains the necessary steps to calculate the \(NPMI\) score of bigrams.

4.2.1 Identifying Wikipedia anchor text community

Actually, Wikipedia has the well-organised anchor text link structure and most of the Wikipedia anchor texts have a corresponding descriptive article, which in turn contains other anchor texts (related to the context of the topic) in its body text. We use this link structure in the preparation of the graph. In this scheme, we consider every anchor text as a node of the graph. Finally, we apply the edge betweenness strategy, as applied in Girvan and Newman (2002) to identify the anchor text communities.

4.2.2 Identifying document’s category and indexing

We uniquely map Dbpedia extended abstracts w.r.t., each of the identified anchor text community. For this, we use the title-based matching. We also check the category information for any category-title related disambiguation. Next, we apply the \(LUCENE\)-based indexing for each of the identified document categories. We use these indexes in the calculation of \(NPMI\) scores of bigrams.

4.2.3 Calculating \(NPMI\)

From the list of the Wikipedia anchor-text communities, we extract all Wikipedia anchor text communities, which show high cosine similarity with the given candidate document. Next, we select the related Dbpedia extended abstract’s category and merge the indexes of all extracted document categories by using \(LUCENE\) (if the given candidate document matches with more than one Wikipedia anchor text communities). As we use pre-calculated indexes of Dbpedia extended abstracts, related to each of the identified Wikipedia anchor text community, so merging the required number of identified indexes
A graph-based unsupervised N-gram filtration technique

is a lightweight process. Finally, we use the LUCENE-based indexes to calculate NPMI. For this we use the text window of size 20 words (1–2 sentences approx.). The equation for NPMI can be given as:

\[
NPMI(t_i, t_j) = \begin{cases} 
-1 & \text{if } p(t_i, t_j) = 0 \\
\log \frac{p(t_i) + \log(\text{tp}(t_i, t_j))}{\text{lgp}(t_i, t_j)} - 1 & \text{otherwise}
\end{cases}
\]

where \(p(t_i, t_j)\) is the joint probability, and can be calculated by counting the number of observations of words \(t_i\) and \(t_j\) in a window of size 20 words. \(p(t_i)\) = probability of occurrence of \(t_i\) in Wikipedia extended abstracts and so on.

4.3 Additional refinements in tentative phrase boundary

Some bigrams which exist in the tentative phrase boundary (as discussed above) may show very poor co-location strength. The NPMI score of such bigrams become zero or less than zero. We break that word sequence into two different sub-sequences. We consider it as a redefining of phrase boundary (see Figures 2 and 3).

Figure 1 A sample text, containing two sentences, ‘S1’ and ‘S2’

\[S1: \quad a \ b \ c \ d \ e \ # \ f \ g \ # \ h \ i \ # \ j \ k \ l\]
\[S2: \quad a \ c \ d \ # \ m \ # \ k \ l \ f \ g\]


Figure 2 Word sequences in sentences with pre-defined phrase boundary

\[S1: \quad [a \ b \ c \ d \ e] \ [f \ g] \ [h \ i] \ [j \ k \ l]\]
\[S2: \quad [a \ c \ d \ m] \ [k \ l \ f \ g]\]

Notes: Brackets ‘[’and ‘]’ are used to represent the phrase boundary in both sentences. Underlined bigrams, i.e., ‘h i’ and ‘j k’ have zero or less co-location score (which requires redefining the phrase boundary for both word sequences).

Figure 3 Redefining phrase boundaries, on the basis of co-location score

\[S1: \quad [a \ b \ c \ d \ e] \ [f \ g] \ [h] \ [i] \ [j \ k \ l]\]
\[S2: \quad [a \ c \ d \ m] \ [k \ l \ f \ g]\]

Notes: Word inside the circle shows the highest betweenness centrality in corresponding word sequence. E.g., word ‘e’ shows highest betweenness centrality in the sequence [a b c d e].
4.4 Calculating weighted betweenness centrality of all distinct words

For this, we prepare a weighted undirected word graph of text and treat every distinct word of the given text as node of the graph. We add an undirected link between every word pair in the text, if they occur adjacent to each other in the given text (see Figure 4, an undirected word graph of text). We denote \( G = (V, E) \) as an undirected graph. Where \( V = \{v_1, v_2, ..., v_n\} \) denotes the vertex set and a link set \((v_i, v_j) \in E\) if there is a link between \(v_i\) and \(v_j\).

Figure 4  Undirected word graph of a text, containing three sentences: S1, S2 and S3, and words: ‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘g’, ‘h’, and ‘i’

Calculating link weight: To calculate the link weight between any two vertexes \(v_i\) and \(v_j\), (denoted by \( \text{LinkWt}(v_i, v_j) \)) we use co-location strength and occurrence frequency of bigrams \((v_i, v_j)\) (same as used in Kumar et al., 2013).

\[
\text{LinkWt}(v_i, v_j) = \text{SR}(v_i, v_j) \times \#\text{Links}(v_i, v_j)
\]  

(2)

where \(\text{SR}(v_i, v_j) = \) co-location strength of bigrams \((v_i, v_j)\), i.e., \(\text{SR}(v_i, v_j) = n\text{PMI}(v_i, v_j)\). Here we only take \(n\text{PMI}(v_i, v_j)\) score, which lies in the range \([0, 1]\). We convert the range \([0, 1]\) to \([0, 10]\). This conversion improves the effect of \(n\text{PMI}(v_i, v_j)\) w.r.t., occurrence frequency of bigrams. For the rest of the bigrams, if \(n\text{PMI}(v_i, v_j)\) show zero or negative score, then we use minimum positive value (i.e., minimum positive NPMI score) of \(n\text{PMI}(v_i, v_j)\) for that edge, \(\#\text{Links}(v_i, v_j) = \) Count of links between \(v_i\) and \(v_j\).

Calculation of path length: we consider the inverse of link weight between two adjacent vertices as path length between them (same as used in Kumar et al., 2013).

\[
\text{Path}(v_i, v_j) = \frac{1}{\text{LinkWt}(v_i, v_j)}
\]  

(3)

where \(\text{Path}(v_i, v_j) = \) path length of path between \(v_i\) and \(v_j\).

Note: we use this path length in the calculation of geodesic distance between any two nodes.

Calculating weighted betweenness centrality: now, to calculate the betweenness centrality score of any node (word) \(v_k\), we use:

\[
C_B(v_k) = \sum_{s \neq v_k \neq t \neq v_k} \frac{\sigma_{st}(v_k) / \sigma_{st}}{(n-1)(n-2)}
\]  

(4)
A graph-based unsupervised N-gram filtration technique

where \( \sigma_{st} \) represents the number of shortest (geodesic) paths between ‘s’ and ‘t’ and \( \sigma_{st}(v_i) \) represents the number of shortest paths between ‘s’ and ‘t’ that passes through \( v_i \).

4.5 N-gram filtration/ candidate keyphrases identification

We apply different strategies to filter the N-grams of different lengths. For this, we use all identified word sequences which lie inside the tentative phrase boundary. The following contains the detailed process.

- **Unigrams (i.e., N-grams, \( N \geq 1 \))**: We identify all unigrams which match with any Wikipedia anchor texts. We discard the rest of them.
- **Bigrams (i.e., N-grams, \( N = 2 \))**: Suppose, we have bigram ‘W1 W2’ (where, \( W_1 \) and \( W_2 \) represent distinct words inside the tentative phrase boundary). Now, if the weighted betweenness centrality score of both words show high difference (i.e., fixed at 40% in entire experiments) or \( \text{NPMI} \) is below average then, we break that bigram into unigrams, i.e., ‘\( W_1 \)’ and ‘\( W_2 \)’ and check, whether they fit as unigrams or not. Otherwise we consider it as candidate keyphrase.
- **N-grams (\( N \geq 3 \))**: We use the following two cases to handle the higher length N-grams.
  
  **Case 1** When the starting/last word in the sequence shows the highest weighted betweenness centrality score (see Figure 5): suppose the last word of the sequence ‘a b c d e’, i.e., ‘e’, shows highest weighted betweenness centrality score. We consider ‘e’ as a tentative boundary word. Next, to identify the actual span of the candidate N-gram, we check the co-location strength (NPMI score) of the nearest surrounding bigrams, i.e., ‘c d’ and ‘d e’. If the co-location strength of ‘c d’ is higher than ‘d e’ then we break the word sequence into ‘a b c d’ and ‘e’. Similarly, if (co-location strength of ‘d e’) \( \geq \) (co-location strength of ‘c d’), then we do not break that word sequence.
  
  **Case 2** When intermediate word shows highest weighted betweenness centrality score (see Figure 5): suppose the intermediate word in the sequence ‘k l f g’, i.e., ‘l’ shows the highest betweenness centrality score. We consider the word ‘l’ as a tentative boundary word. Next, to calculate the actual span of candidate N-grams, we check the co-location strength between the boundary word (i.e., ‘l’) and its adjacent words (i.e., ‘k’ and ‘f’). In this case, if (co-location strength of ‘k l’) > (co-location strength of ‘l f’), then we split the word sequence into ‘k l’ and ‘f g’. Next, if (co-location strength of ‘k l’) \( \leq \) (co-location strength of ‘l g’) then break the sequence into ‘k’ and ‘l f g’. After splitting, we check the stability of unigrams and bigrams (by using the method discussed above).
Figure 5  Identifying phrase boundary for N-grams (N > = 2)

5 Ranking identified candidate phrases/N-grams

The main aim of this step is to rank the identified candidate N-grams (N > = 1) and present top ‘k’ (k > = 1) candidate N-grams as keyphrases for the given text document.

For this, we use a list of the identified candidate keyphrases/N-grams (obtained from the previous step) and prepare N-gram graph of the given text. Next, we apply improved entropy-based algorithm (Ortiz-Arroyo, 2010) to rank the candidate N-grams. The detailed scheme is given below.

5.1 Preparing N-gram graph

We represent the given text as a sequence of the identified candidate keyphrases/N-grams [see Figure 6(a) and Figure 6(b)]. For this we remove all words from the given text, which do not have a match with identified candidate keyphrases/N-grams. Next, we represent each distinct N-gram as a node of graph and prepare an N-gram graph. Formally, we can represent an N-gram graph as, \( G = (V, E) \), where each node, \( V = \{V_1, V_2, ..., V_n\} \) represents the distinct candidate keyphrase/N-gram. Each link \( E = \{E_1, E_2, ..., E_n\} \), represents the link between two adjacent nodes (N-grams) (i.e., nodes captured by window of size two candidates keyphrases/N-grams) [Figure 6(c)].

**Calculation of link weight:** to calculate the link-weight, we use

1. the co-location strength of words between adjacent nodes (candidate N-grams)
2. co-occurrence frequency of adjacent N-grams.

Now, to calculate the co-location strength between words of adjacent nodes, we add NPMI score of all distinct word pairs between adjacent N-grams. For example, to
calculate the co-location strength between adjacent nodes, i.e., 'a b c d' and 'f g', [see Figure 6(c)] we add the NPMI score between all distinct word pairs, i.e., \{(a, f), (a, g), (b, f), (b, g), (c, f), (c, g), (d, f), (d, g)\}. If NPMI score for any word pair shows negative value, then we consider it zero.

Figure 6  Stages of preparing text graph, (a) a sample document with two sentences (b) using candidate phrases/N-grams to represent the document (c) N-gram graph of document

Notes: Figure 6(a) represents a sample text with two sentences (same as given in Figure 3). In Figure 6(b), we represent the same text as a sequence of the identified candidate phrases/N-grams (see phrases inside rectangles). The noisy phrases/N-grams are removed (shown by the crossed rectangle). Figure 6(c) represents the N-gram graph of text represented in Figure 6(b). The number attached to each link shows the count of edges between two vertexes.

Based on the above discussion, the link weight between two nodes, 'a b c d' and 'f g', i.e., \(\text{LinkWt}(a b c d, f g)\) will be

\[
= (n' \text{PMI}(a, f) + n' \text{PMI}(a, g) + n' \text{PMI}(b, f) + n' \text{PMI}(b, g) + n' \text{PMI}(c, f) + n' \text{PMI}(c, g) + n' \text{PMI}(d, f) + n' \text{PMI}(d, g)) \times 1
\]

The generalised formula to calculate the link weight can be given as

\[
\text{LinkWt}(V_i, V_j) = \sum_{vi \in V_i, vj \in V_j} n' \text{PMI}(vi, vj) \times \# \text{Links}(V_i, V_j)
\]  \hspace{1cm} (5)

where \(\text{LinkWt}(V_i, V_j)\) = link weight between two adjacent (connected) nodes \(V_i\) and \(V_j\). As here each distinct N-gram is used as a node in the N-gram graph. So each node (N-gram) may contain a different number of words. Thus, here, \(vi \in V_i\) represents word(s) which exist in node (N-gram) \(V_i\). \(n' \text{PMI}(vi, vj)\) represents the NPMI score of word pairs [normalised from [0–1] to [0-10]].

Calculating path length: similar to the previous discussion (Subsection 3.4), the path length between two adjacent (connected) nodes can be given as.

\[
\text{PathLen}(V_i, V_j) = \frac{1}{\text{LinkWt}(V_i, V_j)}
\]  \hspace{1cm} (6)
5.2 Ranking candidate keyphrases/N-grams

At this step, we use Ortiz-Arroyo’s algorithm to identify top ‘k’ keyphrases for the given document. In Everett and Borgatti (1999), Borgatti (2003), and Borgatti et al., note that selecting the nodes with the top ‘k’ centralities is not the same as selecting the ‘k’ most important nodes. This is due in part to redundancies between top ranking nodes.

In Ortiz-Arroyo (2010), introduced the use of centrality measures with entropy-based algorithm to identify sets of key players. We use this scheme to rank the identified candidate keyphrases. For this, we use the text graph built in the previous section. The detail of entire approach is given below.

Let \( \theta_i \) be a centrality score for the node \( i \). The entropy of a graph \( G \) can be given as:

\[
H(G) = - \sum_{i \in G} \theta_i \log_2 \theta_i
\]  

(7)

Using this definition, ‘Ortiz-Arroyo’s’ algorithm consists of removing each node from the graph, calculating the resulting sub-graph’s entropy and selecting the ‘k’ nodes that caused the greatest decrease in the graph entropy. The following contains pseudo code for finding the set \( B_k \) of ‘k’ most important nodes in a graph based on the centrality measure \( \theta \).

**Algorithm 1** Ortiz-Arroyo’s entropy-based algorithm for finding sets of key nodes in a graph

<table>
<thead>
<tr>
<th>Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
</tr>
<tr>
<td><strong>Step 6</strong></td>
</tr>
<tr>
<td><strong>Step 7</strong></td>
</tr>
</tbody>
</table>

To accompany this algorithm we use the closeness centrality measure.

Centrality measure : \( \gamma(v) = \frac{\sum_{s,t \in G} \sigma_{st}}{\sum_{s,t \in E} \sigma_{st}} \),  
Entropy : \( HC(G) = - \sum_{i \in G} \gamma_i \log_2 \gamma_i \)

(8)

where \( \sigma_{st} \) represents the distance between nodes ‘s’ and ‘t’. Similarly, \( \sigma_{st} \) represents the distance between nodes ‘v’ and ‘t’. \( |E| \) represents total number of edges in the graph.

\( \gamma(v) \) is called the centrality probability distribution, it reflects a node’s ability to reach other nodes in the graph. The entropy-based on it, i.e., \( HC(G) \), identifies the nodes that most affect the graph connectivity. These nodes serve as critical bridges that keep the network connected.
6 Pseudo code

- **Input**: text document, DBpedia extended abstracts, Wikipedia anchor texts.
- **Output**: a list of top ‘k’ extracted keyphrases, where, ‘k’ may be any user defined number (>= 1).

6.1 Algorithm (phrase identification)

Step 1  Apply preprocessing steps for a given document, filter sentences and identify tentative phrase boundaries (Subsections 4.1, and 4.3).

Step 2  Prepare a word graph of text and calculate weighted betweenness centrality score for all words (Subsection 4.4).

Step 3  Now, identify all sophisticated candidate keyphrases (Subsection 4.5). For this, use
   a  word sequences inside the tentative phrase boundary
   b  Wikipedia anchor text collection
   c  co-location strength (Subsection 4.2) of bigrams of the given document
   d  weighted betweenness centrality score of all words of the given document.

6.2 Algorithm (ranking of candidate keyphrases)

Step 1  Prepare N-gram graph by using identified candidate N-grams (keyphrases) (see Subsection 5.1).

Step 2  Apply the ranking algorithm to get top ‘k’ ranked keyphrases (see Subsection 5.2).

7 Experiments

We have tested our devised system by using publicly available and standard dataset related to two different domains, i.e.,

1 SemEval-2010 dataset (Subsection 7.1)
2 DUC 2001 dataset (Subsection 7.2). We also test and compare the quality of the identified candidate keyphrases (Subsection 7.3)

7.1 Experiment with SemEval-2010 dataset

*Details of dataset*: it contains purposefully selected papers from four different research areas of ACM digital library (Kim et al., 2010, 2013). Table 1 contains the details of dataset (trial, training and test), i.e., number of documents per topic in the trial, training and test datasets, across the four ACM document classifications.
Table 1  Number of documents per topic in the trial, training and test dataset, across the four ACM document classifications dataset total document topic

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total</th>
<th>Document topics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>Trial</td>
<td>40</td>
<td>10</td>
</tr>
<tr>
<td>Training</td>
<td>144</td>
<td>34</td>
</tr>
<tr>
<td>Test</td>
<td>100</td>
<td>25</td>
</tr>
</tbody>
</table>

Table 2  Number of author- and reader-assigned keyphrases in the different datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Author</th>
<th>Reader</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial</td>
<td>149</td>
<td>526</td>
<td>621</td>
</tr>
<tr>
<td>Training</td>
<td>559</td>
<td>1,824</td>
<td>2,223</td>
</tr>
<tr>
<td>Test</td>
<td>387</td>
<td>1,217</td>
<td>1,482</td>
</tr>
</tbody>
</table>

7.1.1 Evaluation metrics and strategies

To evaluate the devised system, we calculate the micro-averaged precision, recall and F-score on the test dataset available with (Kim et al., 2010, 2013). For this, we use the evaluation script available with (Kim et al., 2010, 2013).

7.1.2 Sample result

Table 3 presents a sample result on document ‘C-1’ of the SemEval-2010 dataset. Table 3 contains reader assigned keyphrases, Author assigned keyphrases and top 15 keyphrases extracted by our system.

Table 3  Result on document ‘C-1’ of SemEval-2010 dataset

| Reader assigned keyphrases: grid servic discoveri, uddi, distribut webservic discoveri architectur, dht base uddi registri hierarchi, deploy issu, bamboo dht code, case-insensit search, queri, longest avail prefix, qo-base servic discoveri, autonom control, uddi registri, scalabl issu, soft state. |
| Author assigned keyphrases: uddi, dht, web servic, grid comput, md, discoveri |
| Our system: distribut web servic discoveri architectur, grid servic discoveri, grid comput, grid service, uddi, dht, web service, soft state, uddi registry, servic, distribut hash tabl, scalabl issu, autonom control, case-insensit search, queri |

7.1.3 Experiments details

We compare the results of our devised system with the published results of top eight systems of task 5 of SemEval 2010, i.e.,

1  HUMB
2  WINGNUS
3  KP-Miner
4  SZTERGAK
5  ICL
A graph-based unsupervised N-gram filtration technique

We also compare the result of our devised system with two graph-based state-of-the-arts. The details are given below:

- **Single rank**: The single rank system uses a graph-based ranking algorithm to compute the word scores for each single document, based on the document’s local graph. The implementation is similar to Wan and Xiao (2008).

- **Expand rank**: The expand rank (Wan and Xiao, 2008) system uses neighbourhood knowledge for the extraction of keyphrases, along with the use of a different neighbour’s number ($k = 1, 5, \text{ and } 10$). The high precision, recall and F-measure in this system is obtained when $k = 5$, i.e., five neighbours and window of size five for keyphrase extraction are used.

### 7.1.4 Result and evaluation

Table 4 shows the precision, recall and F-measure scores over the reader assigned keyphrases, when the top 5, 10 and 15 keyphrases are extracted from all systems. In this case, our system performs better than the published results of all top-systems. Table 5 shows the precision, recall and F-measure score over the ‘combined author-and-reader assigned’ keyphrases when we extract top 5, 10 and 15 keyphrases from all systems. In this case, the performance of our devised system is comparable to the top-scored supervised systems of SemEval-2010 task (if not significantly better). However, our devised system performs better than the other graph-based state-of-the-arts system. Table 6 shows precision, recall and F-measure score over author-assigned keyphrases when top 5, 10 and 15 keyphrases are extracted from all systems. In this case our system’s performance is slightly poor, but still near to the score of top-supervised systems of task 5 of SemEval-2010.

The results given in all four tables, i.e., Tables 4, 5 and 6 shows that our system is comparable to the top supervised systems of this area and performs better than state-of-the-arts graph-based systems.

#### Table 4  Precision recall and F-measure score over ‘reader-assigned keyphrases’ (SemEval-2010 dataset)

<table>
<thead>
<tr>
<th>System</th>
<th>Top 5 keywords</th>
<th></th>
<th>Top 10 keywords</th>
<th></th>
<th>Top 15 keywords</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P$</td>
<td>$R$</td>
<td>$F$</td>
<td>$P$</td>
<td>$R$</td>
<td>$F$</td>
</tr>
<tr>
<td>Our system</td>
<td>31.4</td>
<td>12.8</td>
<td>18.2</td>
<td>26.1</td>
<td>21.4</td>
<td>23.5</td>
</tr>
<tr>
<td>HUMB</td>
<td>30.4</td>
<td>12.6</td>
<td>17.8</td>
<td>24.8</td>
<td>20.6</td>
<td>22.5</td>
</tr>
<tr>
<td>KX_FBK</td>
<td>29.2</td>
<td>12.1</td>
<td>17.1</td>
<td>23.2</td>
<td>19.3</td>
<td>21.1</td>
</tr>
<tr>
<td>SZTERGAK</td>
<td>28.2</td>
<td>11.7</td>
<td>16.6</td>
<td>23.2</td>
<td>19.3</td>
<td>21.1</td>
</tr>
<tr>
<td>WINGNUS</td>
<td>30.6</td>
<td>12.7</td>
<td>18.0</td>
<td>23.6</td>
<td>19.6</td>
<td>21.4</td>
</tr>
<tr>
<td>ICL</td>
<td>27.2</td>
<td>11.3</td>
<td>16.0</td>
<td>22.4</td>
<td>18.6</td>
<td>20.3</td>
</tr>
<tr>
<td>SEERLAB</td>
<td>31.0</td>
<td>12.9</td>
<td>18.2</td>
<td>24.1</td>
<td>20.0</td>
<td>21.9</td>
</tr>
<tr>
<td>Single rank</td>
<td>25.8</td>
<td>21.3</td>
<td>15.2</td>
<td>22.6</td>
<td>18.8</td>
<td>19.3</td>
</tr>
<tr>
<td>Expand rank</td>
<td>27.3</td>
<td>11.5</td>
<td>16.2</td>
<td>22.6</td>
<td>18.8</td>
<td>20.6</td>
</tr>
</tbody>
</table>
Table 5  Precision recall and F-measure score over 'combine author- reader-assigned keyphrases' (SemEval-2010 dataset)

<table>
<thead>
<tr>
<th>System</th>
<th>Top 5 keywords</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Our system</td>
<td>41.8</td>
<td>13.9</td>
<td>20.9</td>
<td>32.8</td>
<td>22.1</td>
<td>26.4</td>
<td>28.2</td>
<td>27.9</td>
</tr>
<tr>
<td>HUMB</td>
<td>39.0</td>
<td>13.3</td>
<td>19.8</td>
<td>32.0</td>
<td>21.8</td>
<td>26.0</td>
<td>27.2</td>
<td>27.8</td>
</tr>
<tr>
<td>WINGNUS</td>
<td>40.2</td>
<td>13.7</td>
<td>20.5</td>
<td>30.5</td>
<td>20.8</td>
<td>24.7</td>
<td>24.9</td>
<td>25.5</td>
</tr>
<tr>
<td>Kp-MineR</td>
<td>36.0</td>
<td>12.3</td>
<td>18.3</td>
<td>28.6</td>
<td>19.5</td>
<td>23.2</td>
<td>24.9</td>
<td>25.5</td>
</tr>
<tr>
<td>SZTERGAK</td>
<td>34.2</td>
<td>11.7</td>
<td>17.4</td>
<td>28.5</td>
<td>19.4</td>
<td>23.1</td>
<td>24.8</td>
<td>25.4</td>
</tr>
<tr>
<td>ICL</td>
<td>34.4</td>
<td>11.7</td>
<td>17.5</td>
<td>29.2</td>
<td>19.9</td>
<td>23.7</td>
<td>24.6</td>
<td>25.2</td>
</tr>
<tr>
<td>SEERLAB</td>
<td>39.0</td>
<td>13.3</td>
<td>19.8</td>
<td>29.7</td>
<td>20.3</td>
<td>24.1</td>
<td>24.1</td>
<td>24.6</td>
</tr>
<tr>
<td>Single rank</td>
<td>33.6</td>
<td>10.9</td>
<td>16.4</td>
<td>27.9</td>
<td>19.5</td>
<td>22.9</td>
<td>22.4</td>
<td>24.0</td>
</tr>
<tr>
<td>Expand rank</td>
<td>36.1</td>
<td>11.9</td>
<td>17.9</td>
<td>28.3</td>
<td>19.7</td>
<td>23.2</td>
<td>24.1</td>
<td>25.3</td>
</tr>
</tbody>
</table>

Table 6  Precision recall and F-measure score over author-assigned keyphrases (SemEval-2010 dataset)

<table>
<thead>
<tr>
<th>System</th>
<th>Top 5 keywords</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
<td>R</td>
<td>F</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>Our system</td>
<td>22.9</td>
<td>26.3</td>
<td>22.9</td>
<td>15.0</td>
<td>39.7</td>
<td>21.8</td>
<td>12.0</td>
<td>46.7</td>
</tr>
<tr>
<td>HUMB</td>
<td>21.2</td>
<td>27.4</td>
<td>23.9</td>
<td>15.4</td>
<td>39.8</td>
<td>22.2</td>
<td>12.1</td>
<td>47.0</td>
</tr>
<tr>
<td>Kp-MineR</td>
<td>19.0</td>
<td>24.6</td>
<td>21.4</td>
<td>13.4</td>
<td>34.6</td>
<td>19.3</td>
<td>10.7</td>
<td>41.6</td>
</tr>
<tr>
<td>ICL</td>
<td>17.0</td>
<td>22.0</td>
<td>19.2</td>
<td>13.5</td>
<td>34.9</td>
<td>19.5</td>
<td>10.5</td>
<td>40.6</td>
</tr>
<tr>
<td>Maui</td>
<td>20.4</td>
<td>26.4</td>
<td>23.0</td>
<td>13.7</td>
<td>35.4</td>
<td>19.8</td>
<td>10.2</td>
<td>39.5</td>
</tr>
<tr>
<td>SEERLAB</td>
<td>18.8</td>
<td>24.3</td>
<td>21.2</td>
<td>13.1</td>
<td>33.9</td>
<td>18.9</td>
<td>10.1</td>
<td>39.0</td>
</tr>
<tr>
<td>SZTERGAK</td>
<td>14.6</td>
<td>18.9</td>
<td>16.5</td>
<td>12.2</td>
<td>31.5</td>
<td>17.6</td>
<td>9.9</td>
<td>38.5</td>
</tr>
<tr>
<td>Single rank</td>
<td>14.1</td>
<td>18.5</td>
<td>16.0</td>
<td>12.3</td>
<td>31.4</td>
<td>17.6</td>
<td>9.9</td>
<td>39.1</td>
</tr>
<tr>
<td>Expand rank</td>
<td>16.3</td>
<td>22.7</td>
<td>18.9</td>
<td>13.9</td>
<td>38.0</td>
<td>20.4</td>
<td>10.8</td>
<td>44.0</td>
</tr>
</tbody>
</table>

7.2  Experiment with DUC 2001 dataset

Details of DUC 2001 dataset: It contains 309 news articles, categorised into 30 news topics where the average length of a document is 740 words (collected from TREC-9). For the evaluation of keyphrases extracted from all the systems, we employed the annotated set obtained from Wan and Xiao (2008).

7.2.1  Evaluation metrics and strategies

We use same evaluation metrics and strategy as used in Subsection 7.1.1, but, here we also extract the average number of correct keyphrases.
7.2.2 Sample result

A sample result on DUC 2001 dataset: a sample result is given in Table 7. Table 7 contains an abstract dataset [Doc ID: FT923-5089 (DUC-2001)] and annotated keyphrases obtained from Wan and Xiao (2008).

Table 7 Top 10 keyphrases extracted by our devised system on DUC 2001

<table>
<thead>
<tr>
<th>Annotated keyphrases</th>
<th>Our system (Stemmed result, Porter stemmer is used)</th>
</tr>
</thead>
<tbody>
<tr>
<td>hurricane andrew; florida; louisiana; president george bush; election campaign; emergency relief; disaster</td>
<td>presid georg bush; hurrican andrew; mr bush; elect campaign; florida; emergency relief; devast inhabit; disaster; presidenti rival; Louisana</td>
</tr>
</tbody>
</table>

7.2.3 Experiments details

We compare our devised system with the graph-based state-of-the-arts systems, like: single rank, expand rank and TFIDF-based system. The experimental setup and details of systems used in keyphrase extraction by using DUC 2001 dataset, is given below:

1 TFIDF: We use TFIDF-based system [the most robust and stable system according to the latest survey (Hasan and Ng (2010)) and present the result by taking $N = 40\%$ of the size of the document. Here ‘$N$’ represents the number of top scored words used in the formation of phrases. At $N = 40\%$ of size of document (count of the number of distinct words in the document) the TFIDF-based system shows the best result.

2 Single rank: We use the same experimental setup as given in Wan and Xiao (2008).

3 Expand rank: We use the same experimental setup as given in Wan and Xiao (2008). The high precision, recall and F-measure is obtained by this system, when $k = 5$, five neighbours and window of size 5 for keyphrase extraction is used.

7.2.4 Results

The results are given in Table 8. From the results, it is clear that our devised system shows significant improvement in the F-measure score, when top 5 keyphrases and top 10 keyphrases are extracted. The improvements in the quality of results over state-of-the-arts graph-based systems and statistical (i.e., TFIDF) systems, also shows the effectiveness of our devised system.

Table 8 Evaluation results with DUC-2001 dataset

<table>
<thead>
<tr>
<th>System</th>
<th>5-keys extracted</th>
<th>10-keys extracted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>Precision</td>
</tr>
<tr>
<td>TFIDF</td>
<td>1.756</td>
<td>0.351</td>
</tr>
<tr>
<td>Single rank</td>
<td>1.821</td>
<td>0.364</td>
</tr>
<tr>
<td>Expand rank ($k = 1$)</td>
<td>1.834</td>
<td>0.367</td>
</tr>
<tr>
<td>Expand rank ($k = 5$)</td>
<td>1.957</td>
<td>0.391</td>
</tr>
<tr>
<td>Expand rank ($k = 10$)</td>
<td>1.844</td>
<td>0.369</td>
</tr>
<tr>
<td>Our system</td>
<td>2.077</td>
<td>0.415</td>
</tr>
</tbody>
</table>
7.3 Justification of our approach and quality of extracted phrases

In this section, we compare the quality of our candidate keyphrase (N-gram) identification system with the state-of-the-arts unsupervised keyphrase extraction systems. The details are:

- **TFIDF**: We use the phrase formation steps by TFIDF-based system [the most robust and stable system according to the latest survey (Hasan and Ng, 2010)] and present the result by taking $N = 40\%$ of the size of the document. Here ‘$N$’ represents the number of top scored words used in the formation of phrases. At $N = 40\%$ of size of document (count of the number of distinct words in the document) the TFIDF-based system shows the best result.

- **Single rank**: We use the phrase formation steps (as discussed in Hasan and Ng, 2010) to prepare all candidate N-grams.

- **Expand rank**: We use the best results of phrase formation step (Hasan and Ng, 2010), obtained by using $k = 5$ nearest neighbour documents at a time.

From the results given in Table 9, it is clear that our devised system effectively identifies the candidate keyphrases (N-grams).

<table>
<thead>
<tr>
<th>Systems</th>
<th>SemEval-2010 dataset</th>
<th>DUC 2001 dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Author</td>
<td>Reader/annotator</td>
</tr>
<tr>
<td>TFIDF</td>
<td>74.04%</td>
<td>76.59%</td>
</tr>
<tr>
<td>Single rank</td>
<td>74.00%</td>
<td>75.14%</td>
</tr>
<tr>
<td>Expand rank ($k = 5$)</td>
<td>77.15%</td>
<td>79.32%</td>
</tr>
<tr>
<td>Our devised system</td>
<td>97.03%</td>
<td>93.71%</td>
</tr>
</tbody>
</table>

7.3.1 Time complexity and related issue

The running time of the devised system depends upon the running time of the following key ingredient techniques:

1. **Weighted betweenness centrality (Subsection 4.2)**: The running time of this algorithm is: $O(n^2 \log n + nm)$; where ‘$n$’ = total number of nodes (here distinct words) in the word graph of text and ‘$m$’ = total number of edges in the word graph of text.

2. **Calculating NPMI score (used in Subsection 4.4)**: We consider the LUCENE-based indexing part as offline process. Thus, we only concentrate on calculation of NPMI score. In a traditional system having 2GB RAM and 2.2 GHz dual core processor, the devised system calculates the NPMI score of 7000 distinct bigrams.

3. **Using Ortiz-Arroyo’s entropy-based algorithm for finding sets of key nodes in a graph (Section 5)**: The running time of this algorithm is: $O(n^3)$; where ‘$n$’ = total number of nodes (here distinct candidate keyphrases/N-grams) in the N-gram graph.
However, the count of distinct candidate N-grams lie in the range of 30 to 40% of total number of words of the given document.

Note: The above discussed complexity issues, clears that our devised system can be effectively applied for offline and online keyphrase extraction.

8 Conclusions

In this paper, we present a two-step process for keyphrase extraction, i.e.,

1 identification of candidate keyphrases
2 ranking of the identified candidate keyphrases.

To identify the candidate keyphrases, we effectively combine the global statistical features (obtained by using weighted Betweenness centrality) and semantic expansion strategies (obtained by using NPMI score of words pairs, calculated on nearest neighbour Dbpedia extended abstracts). To improve the speed of calculation, we also introduce the use of Wikipedia anchortext community detection-based approach.

We introduce the N-gram graph-based scheme which reduces the biasness effect of highly frequent lower length candidate keyphrases (N-grams) and also preserves the local context of words. To reduce the effect of noisy terms and to improve the quality of ranking, we effectively utilise the combined use of information theoretic measure and statistical features. Experimental results show that our devised system is more stable than other state-of-the art systems on different datasets. The limited linguistic dependencies and evolving nature of Wikipedia/Dbpedia, opens the scope to extend the system into multiple domains and multi-lingual environments.

References


**Notes**