# Chapter 4

# Topic Model based Keyphrase Extraction

Among the unsupervised keyphrase extraction methods, Graph-based ranking is one of the popular approaches. In these methods, a graph is built from the input documents and its nodes are ranked according to their importance [62]. Each node of the graph represents a candidate keyphrase from the document and each edge establishes the connection between two related candidates. Depending on the weight of the edges, *i.e.*, how strong the connection is between two words, the ranking is done among the words or keyphrases. But this technique does not guarantee whether the high ranked words or keyphrases are relevant to the major topics of the document collection or not. Another bottleneck of the graph-based methods arises when multiple topics are present in the documents. These methods often converge

into a single topic instead of considering other substantial topics of the documents. To overcome these issues, some unsupervised keyphrase extraction techniques based on topic decomposition are proposed in literature. The candidate keyphrases of each documents are grouped into topics in such a way that each topic is composed of all and only those candidate keyphrases that are related to that topic [36]. Taking cue from these facts, [57] has proposed a Topical PageRank (TPR) on word graph to decompose traditional random walk into multiple random walks specific to various topics.

### 4.0.1 Methodology

Topic PageRank based Keyphrase Extraction method consist of two phases as describe below:

1. First, topics of documents are determined and obtained by using a suitable topic interpreter. For this particular scenario, Latent Dirichlet Allocation (LDA) performs the role of topic interpreter.

2. Next step is the introduction of Topical PageRank, which incorporates topical information in random walk on the constructed word graph according to word co-occurrences within the given documents.

Let us discuss few details about the above two points.

A. **How to select topics from a document using LDA?:**

Let us consider a sample of words, $w$, taken from document $d$. A topic, say $z$, is sampled from the same document $d$ where $\eta$ is the total number of topic present in $d$ (the number of topics per document is pre-defined as explained later). Then the prior probability of words $w$ in the given document $d$ is calculated as in the following Equation 4.1.

$$p(\frac{w}{d,\varphi,\nu}) = \sum_{z=1}^{\eta} p(\frac{w}{z,\nu}) p(\frac{z}{d,\varphi}) \tag{4.1}$$

Here, $p(\frac{w}{z})$ indicates how much that topic $z$ focuses on the words $w$ and $p(\frac{z}{d})$ is the measure of how much the document $d$ relates to the topic $z$. $\varphi$ and $\nu$ are the conjugate Dirichlet priors (as generally documents follows the Dirichlet distribution).

B. **How to extract keyphrases from decomposed topic?:**

A word graph is constructed for each document $d$ according to word co-occurrences within $d$. The word graph consists of words $w_i$ as nodes and edges between them (say $w_i$ and $w_j$). The weight, $\omega_{(w_i,w_j)}$, of any edge $(w_i, w_j)$ is their co-occurence count within a sliding window with maximum $\widetilde{M}$ words.

Like PageRank algorithm, the importance scores, $I(w_i)$, of any word/node depends on the out-degree of the node (say $\gamma(w_i)$). Thus, the importance score ($I_z(w_i)$) of each word with respect to different topics ($z$) is calculated by the following Equation 4.2.

$$I_z(w_i) = \xi \sum_{j:w_j \to w_i} \frac{\omega_{(w_i,w_j)}}{\gamma(w_j)} I(w_j) + (1-\xi)p(C) \tag{4.2}$$

Where, $\xi$ is a damping parameter, defined in Subsection 4.0.1.1 and the out-degree is calculated as in the Equation 4.3. We calculate the probability ($p(w_i)$) of any $i^th$ word $p(w_i)$ as described in the subsection 4.0.1.

$$\gamma(w_i) = \sum_{j:w_i \to w_j} \omega_{(w_i,w_j)} \tag{4.3}$$

The construction of word graph followed by keyphrase extraction and ranking is presented in Algorithm 3. In algorithm 3, the steps 1 to 7 select each distinct words from document corpus and save it as a node. The steps 8 and 9 initially connect each node ($w_i$) to every other node ($w_j$). Now, if for a window size $\widetilde{M}$, $w_i$ and $w_j$ do not co-occur at least once, step 11 to 18 delete ($w_i - w_j$) link from the graph and for all other edges, weight of ($w_i - w_j$) is equal to number of co-occurrences within $\widetilde{M}$. If the formed graph is cyclic, then step 19 to 21 convert it to DAG (Directed

**Algorithm 3:** Construction of the Word Graph and Ranking

**Data:** Corpus $D$, Topic Set $Z$, Graph $G = V, E$
**Result:** Keyphrases $\mathcal{K}$

1 $V \leftarrow \Phi$;
2 $E \leftarrow \Phi$;
3 **for** $\forall d \in D$ **do**
4  **for** $\forall w_i \in d$ **do**
5  $v_i \leftarrow w_i$ ▷ $i \leq |d|$;
6  **end**
7 **end**
8 **for** $(\forall w_i, w_j \in D) \wedge (i \neq j)$ **do**
9  $e_{ij} \leftarrow (w_i, w_j)$ ▷ $1 \leq i, j \sum |d|$;
10 **end**
11 **for** $(\forall e_{ij} \in E) \wedge$ *(window_size=* $\widetilde{M}$*)* **do**
12  **if** *co-occur*$(w_i, w_j)$=0 **then**
13  $e_{ij} \leftarrow \phi$;
14  **end**
15  **else**
16  $\omega_{e_{ij}} \leftarrow$ *co-occur*$(w_i, w_j)$;
17  **end**
18 **end**
19 **if** *cyclic*$(G) =$ *true* **then**
20  $\hat{G} \leftarrow G$ ▷ $\hat{G}$ is a DAG;
21 **end**
22 **for** $\forall v_i \in V$ **do**
23  $\gamma(v_i) \leftarrow$ *out-degree*$(v_i)$;
24  Calculate $I_z(v_i)$ using Equation 10;
25 **end**
26 **for** $\forall z \in Z$ **do**
27  $I_z(\pi) = \sum_{w_i \in \pi} I_z(v_i)$▷ $I_z(\pi)$ is the final ranking score for each candidate keyphrase $\pi$;
28 **end**
29 Sort$(\pi)$ ▷ In descending order;

Acyclic Graph), else the flow goes to step 22 and calculate the out-degree of each node/vertex in the graph,$\gamma(v_i)$, in the step 23 and the importance score of candidate keyphrases separately for each topic,$I_z(v_i)$, in the following step 24. Finally, steps 26 to 29 calculate the final ranking score for each candidate keyphrase, *i.e.* $I_z(\pi)$

and sort them in descending order.

#### 4.0.1.1 Parameter Declaration

In the paper by Liu *et al.* [57], experiments are performed by varying some antecedent parameters. The experimental results show that for certain range of values of these parameters, the keyphrase extraction method works better. From these range of values, some particular values for each parameter are chosen prudently and assigned. We select the sliding window size as $\widetilde{M} = 20$. The sliding window size pertains to the length of word sequence up to which we consider the co-occurrence of any two words. We assign a maximum number of topics per documents, $\eta$, is as 100. The reason behind this is that observation says if the number of topics is very high, the performance does not change much.

The damping factor, $\xi$, which indicates that each vertex has a probability of $(1 - \xi)$ to perform a random jump to another vertex within this graph, is set as 0.5. We consider top $\mathcal{K}$ keyphrases ($\mathcal{K} = 5$ here).

### 4.0.2 Experimental Results and Analysis

The same baseline framework which is discussed in the previous chapter in section 3.1, is used here for the experiment and the comparison of the Topic model based keyphrase extraction technique; let us call it TMQEM. This TMQEM model is also executed and tested on the same Wiki11 dataset in the section 3.4.1 of the previous

chapter. Similarly for Evaluation Measures R-precision and MAP are calculated and the a five-fold cross validation is performed on the entire topic set. In the previous chapter, it is also established that The MIQEM is the best performing keyphrase extraction model and it provides the best retrieval results. So in this section we compare the TMQEM against the MIQEM. The quantitative comparison between the two model is presented in the following table 4.1.

| Metric/Method | $P_{\text{Set 1}}$ | $P_{\text{Set 2}}$ | $P_{\text{Set 3}}$ | $P_{\text{Set 4}}$ | $P_{\text{Set 5}}$ | MAP |
|---|---|---|---|---|---|---|
| **MIQEM** | 0.3849 | 0.3765 | 0.3451 | 0.428 | 0.3183 | 0.3655 |
| **TMQEM** | 0.4484 | 0.4412 | 0.4459 | 0.4845 | 0.4216 | 0.4483 |

TABLE 4.1: Average Precision per query set and MAP

The columns of the table depict the Average Precision values of retrieved results for each test set of 10 queries between the two query expansion approaches. We present the Mean Average Precision (MAP) values over the total set of queries in the last row of the table 4.1. It is clear from the table that TMQEM performs better than MIQEM.

Figures 4.1 through 4.5, exhibit the per query results for each of the five test query sets. This clearly demonstrates the fact that stressing on topics rather than keyphrases can generate better results. While methods for keyphrase extraction can perform decently in some cases, it can induce various biases and often fails to comprehensively capture the linguistic nature of words or terms embedded across various documents. The biases are induced due to the fact that standalone keyphrase extraction models do not consider for inter-relation between topics across the dataset,

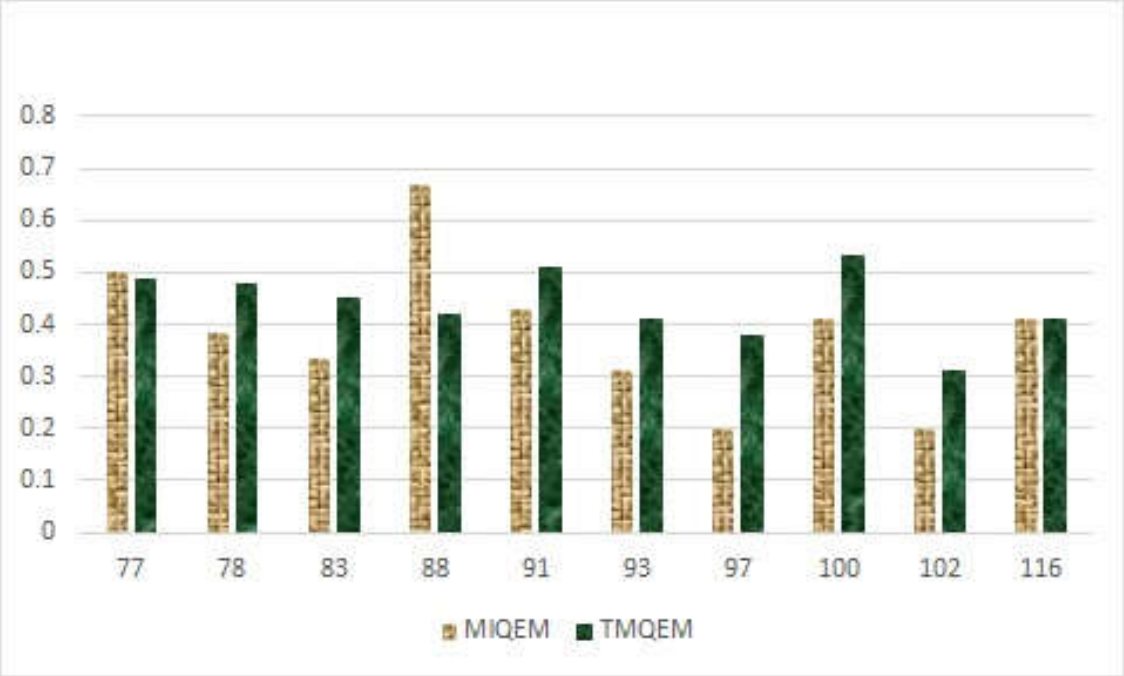


FIGURE 4.1: Per Query Precision Comparison for Set 1

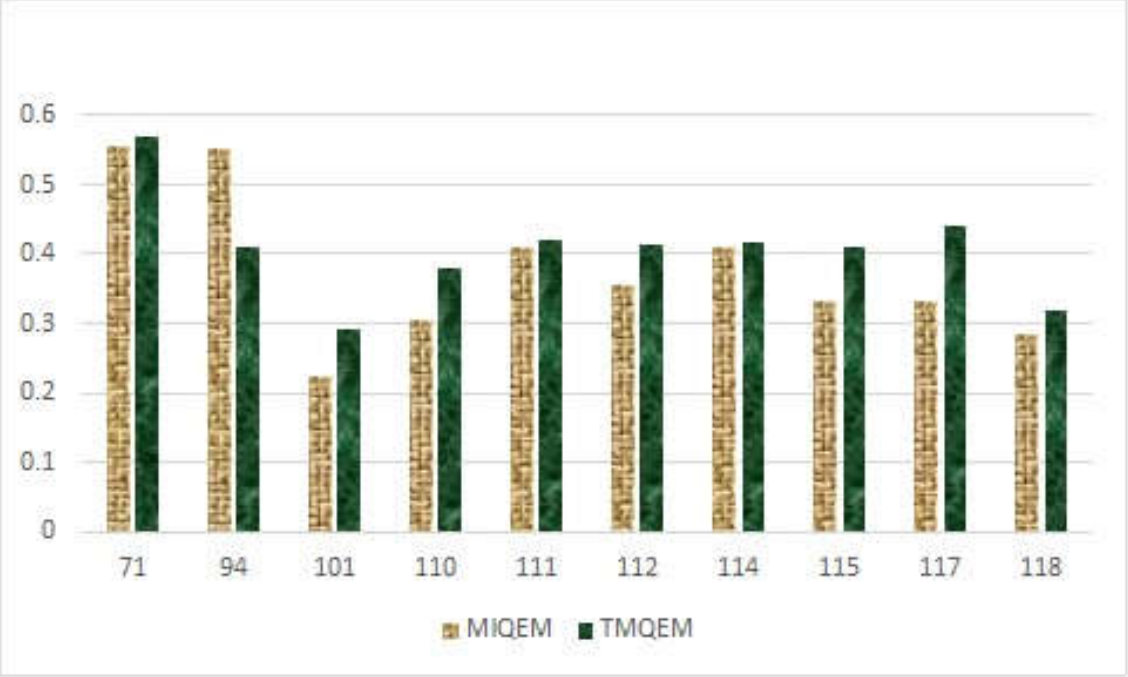


FIGURE 4.2: Per Query Precision Comparison for Set 2

which can heavily influence performance of any information retrieval engine. So, if a group(s) of word occurs in a particular document with a high statistical measure,

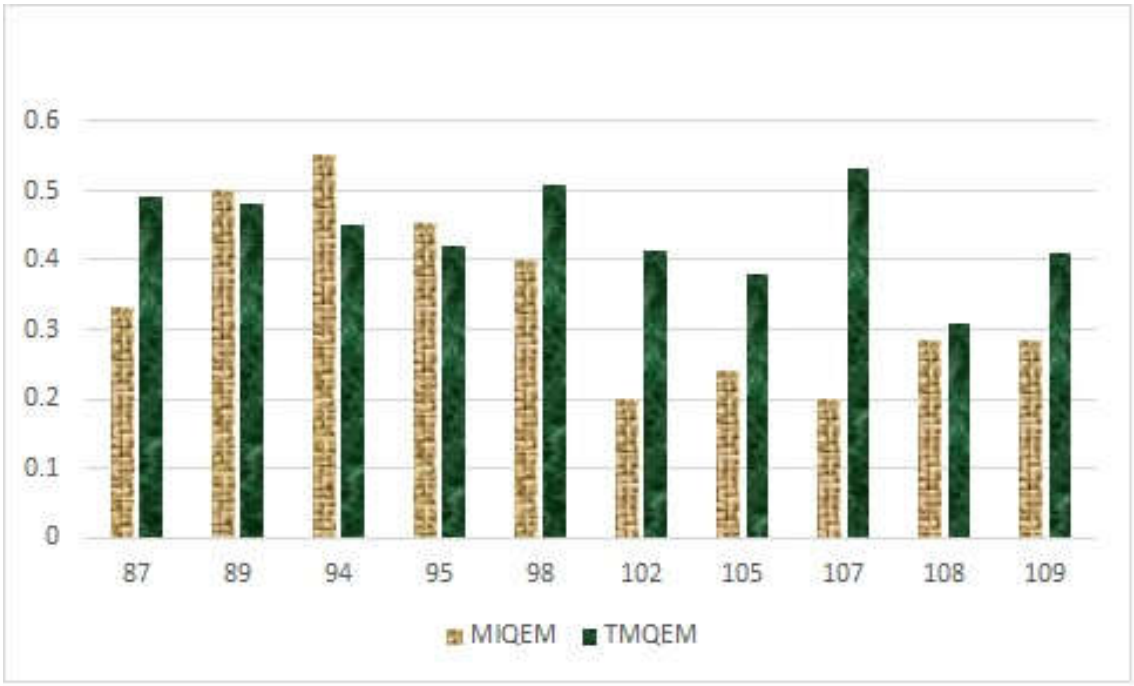


FIGURE 4.3: Per Query Precision Comparison for Set 3

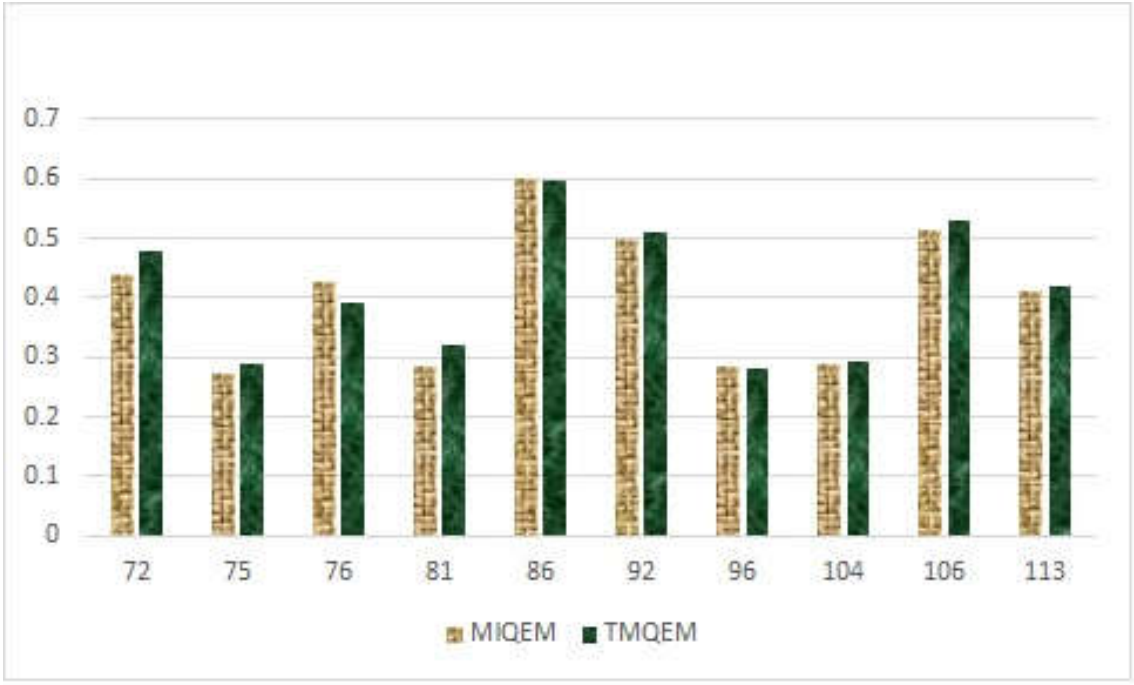


FIGURE 4.4: Per Query Precision Comparison for Set 4

the simple keyphrase extraction techniques may be tricked into believing that it is a key term and might be included as one of the candidates. While if the experiment

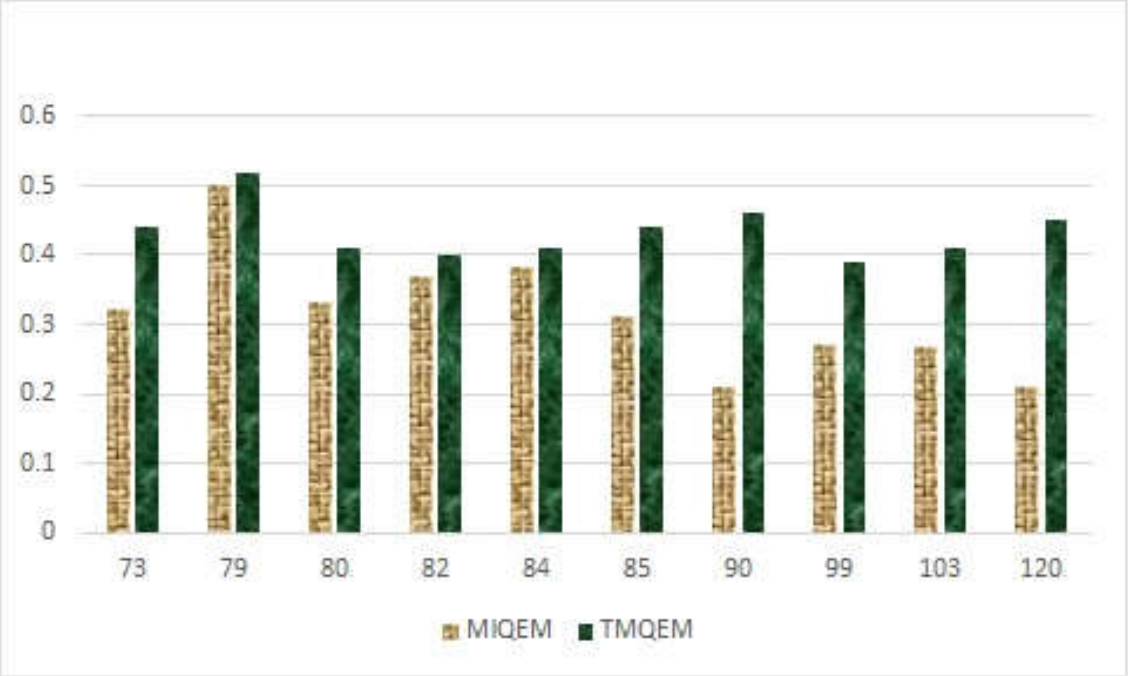


FIGURE 4.5: Per Query Precision Comparison for Set 5

is considered across the document set it may turn out be an outlier rather than a norm compared to other terms.

### 4.0.3 Discussion

In this chapter we adopt a Topical PageRank based keyphrase extraction approach. Basically it is a graph-based ranking method where Latent Dirichlet Allocation (LDA) is used for topic decomposition. This approach is a purely unsupervised learning where multiple random walk is performed to choose the candidate keyphrases. Initially the topics, present in the documents are identified by LDA, then importance of each words are calculate with respect to each topic based on the word co-occurrence. Thus a word graph is constructed based on the word importance

where out degree of each node represent importance of words. Hence, this method captures both the word distribution as well as topic distribution and this makes the approach insightful. Our comparative study reveals that the TMQEM outperforms other approaches. It is also effective and robust although it does not perform well when the corpus size is small.