Proceedings of 15th IOE Graduate Conference
Peer Reviewed
Year: 2024 Month: May Volume: 15
ISSN: 2350-8914 (Online), 2350-8906 (Print)

Topic-Wide Keyword Generation From Nepali Documents Through Graph-Based Extraction and LDA Topic Models

Sagar Shrestha [a], Sitaram Pokhrel [b], Smita Adhikari [c]

[a, b, c] *Department of Electronics and Computer Engineering, Paschimanchal Campus, IOE, TU, Nepal*
[a] seashrestha120@gmail.com, [b] sitaram@wrc.edu.np, [c] smita@wrc.edu.np

Abstract
Keywords generally represent a document's key information that provide an insight into what the document is about. Most traditional keyword extraction methods often assume a document include discussions about a single topic when generating keywords. Consideration of the role of the generated keywords in representing all the unidentified latent topics in a document is severely lacking. Hence, a hybrid unsupervised keyword extraction technique is presented that leverages graph-based extraction to evaluate the topic-wide keywords for each document by feeding the candidate word list generated from it to the LDA topic modeler. La-BSE is used to embed syntactic and semantic information of all the words and sentences in a Nepali document and the similarities among all words and sentences are evaluated. Three graphs (sentence-to-sentence, word-to-word, and sentence-to-word) are generated to incorporate the relationships between the respective entities using the similarities. A list of candidate keywords and their respective importance are obtained by using an iterative algorithm, at which traditional extraction techniques often end. LDA is used for topic modeling to identify the groups of words with high probability to fall within each topic by aggregating the extracted candidate keywords and their importance values from graph-based extraction. Since one word can appear in multiple topics, the topic-wide keywords are selected based on which words appear in the greatest number of topics. The results are compared with the human-generated keyword lists and evaluated against existing baselines. The performance evaluation shows considerable improvement from just using the graph-based extraction technique or other existing literature like applying K-means after graph-based extraction to allow representation of low-ranked words as well. The architecture can be applied in topic recommendation by analyzing the sets of keywords obtained from a huge corpus to write about the popular topic identified from human inference.

Keywords
Keyword Extraction, Graph-Based Extraction, LDA, Topic-Wide Keywords, Sentence Embeddings

1. Introduction

Keywords help identify the kind of topics a certain document is diving into as it generally represents the content within it. A single document can belong to different categories since various parts of the document, or even a single sentence, might be recognized as part of a different category. So, keywords for those documents should be chosen such that the context they are used for in the document should be reflective of many topics found in the document, even those that appear less frequent. There have been strides in development of automatic keyword extraction process, since manually annotating and extracting keywords is grueling and time-consuming. Adding the process of topic identification and allocation of these keywords in different topics to generate topic-wide keywords makes it even more complicated. One thing of grave importance to note here is that a word can belong to multiple topics based on the context it was used within the document.

Automatic keyword generation can be either supervised or unsupervised. Supervised methods require manually annotated training datasets which makes it slightly time-consuming, and it also does not work for documents that belong to categories beyond the domain of its training corpus. Among the unsupervised methods of keyword extraction, graph-based methods are the most prominent and the most efficient [1]. The document is first pre-processed to get a list of words without whitespaces or containing any stopwords, symbols, and numbers. These words are called the candidate keywords of the document. The words are then represented with numbers by using either statistical methods, for example the sliding window technique, or embedding techniques that encode syntactic or semantic information in those words. Graph-based extraction methods generally create a graph to represent the relationships, represented by edges, between the entities in the document, viz., candidate words, represented by the nodes in the graph. Keywords are ranked and finalized according to the importance evaluated from various means like PageRank [2] using the information obtained from the graphs. Existing approaches have evolved to creating three graphs from the document, namely, word-to-word graph, sentence-to-sentence graph, and sentence-to-word graph, to incorporate the impact of sentences when choosing keywords. Embedding sentences for vectorization is a relatively new technique that replaced the statistical approached that was almost always used. The relationship between the respective entities are evaluated as the cosine similarity between them. These relationships are used to calculate the importance of each candidate keywords. A select few words with the highest importance values are selected as the keywords for the document.

These extraction methods produce keywords based on the semantic placement of the words in the document which regard the conceptual meaning of the words. There is no

mention of how much of the many underlying topics the document covers are considered when generating the keywords. The proposed architecture remedies it by evaluating how much each word is used in context of the undetermined underlying topics in the document based off of the importance value calculated from traditional extraction methods. Topic modeling is the additional step employed to produce topic-wide keywords.

The entire process is performed on a Nepali News Dataset [3]. Pre-processing is performed to obtain a list of words in their root form after removing the stopwords and symbols using the Nepali_nlp pre-processor [4]. The embeddings for words and sentences are generated from Language-agnostic BERT Sentence Embeddings (La-BSE) vectorization [5]. Cosine similarity is used to quantify the relationship between and among words and sentences which become the edges to one of the respective aforementioned three graphs. An iterative formula uses every single value of the edges a word is related to in all three graphs to generate importance of each word. Topic modeling is done by the Latent Dirichlet Allocation (LDA) approach so that each word has values assigned to it which represent the probabilities that it belongs to each underlying topic in the document. The input to LDA is the list of importance values from the iterative formula at a document level instead of the traditional use of the frequency count of words at a corpus level. This is the main contribution to this literature. The final keywords are chosen as 10% of the total unique words in the document that are prevalent in the most amount of topics with highest probabilities.

2. Related Work

TextRank [1] established the basis of graph-based keyword extraction by using the PageRank formula for a single undirected graph of words and their relationships. The relationship quantification used in PageRank were obtained as the co-occurrence values from the sliding window technique. Wan et al. [6] introduced using three graphs in keyword extraction to incorporate the impact of sentences each word belong to during importance evaluation as well. The relationship between the entities was represented by the semantic similarity between their TF-IDF vector representations. EmbedRank [7] quantifies the sentences and the document into higher-dimensional vector space by using pre-trained models, Sent2Vec [8] and Doc2Vec [9], which also expanded the sliding window to cover sentences and documents as well. Kazemi et al. [10] used Facebook's LASER (Language-Agnostic SEntence Representations) to embed the text during graph-construction for the tasks of focused summarization and explanation extraction. LASER supports 93 different languages. Meanwhile, La-BSE [5] is a pre-trained sentence embedding technique introduced by Google to project words and sentences to a higher dimensional space of 768 dimensions. It uses language model pre-training advances like Masked Language Modeling (MLM) and Translation Language Modeling (TLM) on an architecture similar to BERT and then fine-tune on a translation ranking task. It is trained on 17 billion monolingual sentences and 6 billion bilingual sentence pairs. To enhance the model and expand vocabulary coverage, a transformer architecture with twelve layers and a vocabulary of 500,000 tokens undergoes pre-training employing MLM and TLM across 109 languages. This is the current state-of-the-art in terms of embedding techniques and is used as a contribution to this keyword extraction technique to enhance the representation of words and sentences in terms of numbers.

Gu Yijun and Xia Tian [11] used the LDA topic model to first generate a word-topic distribution and then use TextRank on top of it to generate a list of topic-wide keywords. Sun et al. [12] also used LDA to extract significant keyphrases but by first utilizing the FP-growth algorithm to identify frequent co-occurring neighborhood words as candidate phrases instead of a graph-based approach. Ying et al. [13] used co-occurrence method to quantify sentences while using fastText to quantify words during graph generation. K-Means clustering approach was employed afterward for the keywords to also reflect the lower ranked words by including the centroid words of the clusters that contain those lower ranked words as well so that it covers multiple topic clusters. ConceptRank [14], meanwhile, used hierarchial topic modeling to have the keywords consider the context in which the words are used so that a word isn't deemed important just because it appears in multiple different context scenarios with different meaning. It identifies them as separate words due to their differences in meaning in context.

For much of the existing literature, the keywords generated do not entirely tell a tale of whether they cover as many of the topics in the document that the algorithm is blind to. For consideration of the fact that a word can be used in different context to discuss different topic within the same document, LDA is used after the refined graph-based keyword extraction phase to assign words to each topic with probability distributions. For instance, the word "money" can be used in the main topic discussing "finance" but also in context of the "entertainment" topic, say, when discussing how much a movie has made in the box office. So, "money" might belong to the finance topic with a higher probability and to the entertainment topic with a lower probability albeit higher than most other words that might belong to both topics. Thus, "money" is considered a keyword since it belongs to the most number of underlying topics in the document with high probabilities. Note that the proposed model doesn't specify the labels of the topics since LDA is unsupervised; instead LDA just gives them a pre-determined label when learning, which is one of the whole numbers in order.

The rest of the paper goes through the following process: Section 3 discusses the entire process of topic-wide keyword extraction from pre-processing to word-topic distribution and keyword selection. Section 4 provides the results of evaluation of the proposed architecture against three other existing architecture that form the bases of this paper. Finally, section 5 provides the conclusion to the work performed and the future work that can be done as alternatives.

3. Methodolgy

The traditional method of keyword extraction has been updated by including the state-of-the-art techniques in each

phase. The vectorization of sentences and words have been updated from statistical methods like sliding window technique or the Term Frequency-Inverse Document Frequency (TF-IDF) technique to the state-of-the-art embedding techniques to encode their semantic and syntactic information.

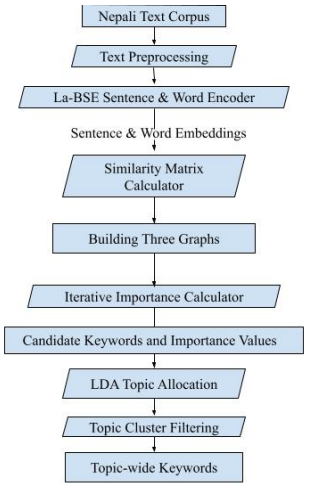

Figure 1: Workflow of Topic-Wide Keyword Extraction

Three graphs are used instead of only word-to-word graph to incorporate the importance of sentences to each word. The extraction method is finally expanded to have the keywords selected cover majority of the topics underlying in the document by feeding the importance values obtained from extraction phase into the LDA topic model.

3.1 Dataset for Keyword Extraction

Nepali News Dataset [3] contains over 4000 text documents from 20 variety of categories extracted by web-scraping various news portals written in Nepali language like eKantipur, OnlineKhabar, Setopati, etc. All of the documents are fed into the model without regard for their category in random. The documents are all of varying lengths including documents with 10-20 sentences as well as ones with over 100 sentences.

3.2 Text Pre-Processing

The document is converted to a format that the model can make use of and that it only needs. This includes removal of noisy data like whitespaces, stopwords (from both the NLTK library and additional stopwords manually provided that are not in the NLTK library), and symbols to get words for word-to-word and sentence-to-word graphs. As for the sentences, they can be fed in full without any change except for tokenization. Each individual sentences are recognized with the 'l' identifier at the end. Meanwhile, LDA needs the input in the format of a collection of words that each sentence contains, which is easily obtained by tokenizing each sentence to get its constituent words while also removing the said noisy data. Each of these words in individual sentences will later be associated with their respective importance values obtained from the first phase of graph-based extraction which will later be aggregated with the words to be fed into the LDA topic model.

3.3 Sentence and Word Embeddings

The words and sentences in the document are to be quantified for the need to encode the semantic and syntactic information contained within them and use them to evaluate the relationships between each other. Use of embeddings of a certain dimension size instead of statistics is relatively new in the world of graph-based keyword extraction. The state-of-the art is the La-BSE encoder provided by Google that can generate embeddings for words or sentences for the 109 languages it handles, including Nepali, without having to specify the language of the input. Each word and sentence in the document is thus encoded into a vector of 768 dimensions.

For the set of 'v' words, W = $\{w_i \mid 1 \le i \le v\}$, the output of La-BSE encoder is list of word_vectors = $\{\vec{w}_i \mid 1 \le i \le v\}$. As for the set of 'u' sentences, S = $\{s_i \mid 1 \le i \le u\}$, the set of sentence vectors we obtain is sentence_vectors = $\{\vec{s}_i \mid 1 \le i \le u\}$. Each of $\vec{w}_i$ and $\vec{s}_i$ are of 768 dimensions which is relatively large but efficiently encodes the semantic and syntactic information of the words and sentences.

3.4 Similarity Matrix Calculation

Similarity matrices replace the traditional co-occurrence matrix in graph-based keyword extraction to evaluate the impact of all the words and sentences in the document on each other. Cosine similarity is used to evaluate the similarities between two words and sentences using the sets of vectors obtained above, given by the following two formulae:

$$\text{sim}(s_1, s_2) = \frac{\vec{s_1} \cdot \vec{s_2}}{|\vec{s_1}| \cdot |\vec{s_2}|} \tag{1}$$

$$\text{sim}(w_1, w_2) = \frac{\vec{w_1} \cdot \vec{w_2}}{|\vec{w_1}| \cdot |\vec{w_2}|} \tag{2}$$

The similarity matrix $Q_{u\times u}$ is used to save the weights of the edges of the sentence-to-sentence graph for a document with u sentences S = $\{s_i \mid 1 \le i \le u\}$, where the element Q_{ij} denotes similarity between sentences s_i and s_j. The matrix is then normalized to $\tilde{Q}_{u\times u}$ so that the values remain consistent within the interval [0,1] in accordance with the iterative importance calculation phase. The elements of $Q_{u\times u}$ are the edges of the sentence-to-sentence graph. Similarly, the similarity matrix among words is saved in $R_{v\times v}$ with v distinct words. Thus, each element $R_{ij} = \text{sim}(w_i, w_j)$. The upper limit of both i and j for R_{ij} is v. When i = j, $R_{ij} = 1$ since they represent the same word. Again, the $R_{v\times v}$ matrix is normalized to $\tilde{R}_{v\times v}$ so that the sum of each line is 1, i.e., the similarity values of one word with all other words are normalized. The elements of $R_{v\times v}$ are the edges of the word-to-word graph.

For evaluating the edges of the sentence-to-word graph, the TF-IDF formula is applied at a document level instead of a corpus level to sufficiently evaluate the impact of words on the sentences it is present in and vice versa. The following formula is thus concocted with the adjustments:

$$\text{sim}(s_i, w_j) = \frac{wf_{w_j} \times isf_{w_j}}{\sum_{w \in s_i} wf_w \times isf_w} \tag{3}$$

where, for each word w_j in sentence s_i, wf_w is the word frequency in the sentence; i.e., relates to the number of times

w_j appears in s_i; and isf_w is the inverse sentence frequency of the word; i.e., relates to the number of sentences w_j appears in (akin to inverse document frequency in the TF-IDF method but for sentences instead of documents).

By definition,

$$wf_{w_j} = \frac{\text{number of times } w_j \text{ appears in } s_i}{\text{total number of words in } s_i} \tag{4}$$

$$isf_{w_j} = \log\left(\frac{\text{total number of sentences}}{\text{number of sentences containing } w_j + 1}\right) \tag{5}$$

The results of the equation (3), i.e., weights of the edges in the sentence-to-word graph, is then saved in the matrix $W_{u\times v}$. Similarly, W is also normalized to $\tilde{W}$. For the purpose of evaluating the importance of words in the iterative calculation section, we also need to normalize the transpose of W to $\hat{W}$.

3.5 Graph Construction

The elements of the aforementioned three matrices are used to create three different graphs. $Q_{u\times u}$ represents the edges in the sentence-to-sentence graph, $R_{v\times v}$ represents the edges in the word-to-word graph, and $W_{u\times v}$ represents the edges of the sentence-to-word graph.

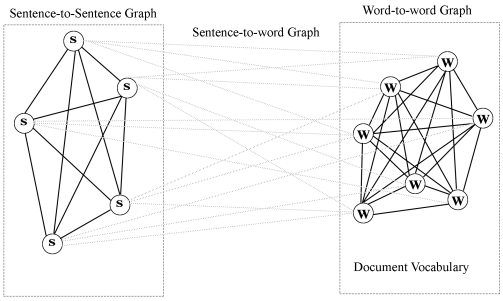

Figure 2: A Simplistic Representation of All Three Graphs in the Extraction Phase

In figure 2, the block on the left is the sentence-to-sentence graph, named G_{SS}, with each edge (indicated by solid lines) representing the similarity between the connected sentences before normalization, saved in $Q_{u\times u}$. Similarly, the block on the right is the word-to-word graph, named G_{ww}, with the edges representing the similarity between the connected sentences before normalization, saved in $R_{v\times v}$. If we remove all the solid line edges from the entire graph in the figure, we obtain the word-sentence graph, named G_{ws}, with the dotted lines representing the relationship between the connected word and the respective sentence before normalization saved in $W_{u\times v}$.

3.6 Iterative Importance Calculator

Calculation of the importance of every word in the document after pre-processing is performed by encoding the relationship between sentences and its effects on the importance of words so that the semantic co-dependencies are accounted for. The use of state-of-the-art embedding technique in vectorization of words and sentences, and the evaluation of the three similarity matrices and their corresponding graphs, ensure that the semantic and syntactic information within the document are sufficiently captured. Using these results of quantification, the importance of each word and simultaneously each sentence is calculated based on two established hypotheses brought on by Wan et al. for keyword generation [6].

- A word is important if it is connected to other important words, and a sentence is important if it is connected to other important sentences.
- A word must be important if it is present in important sentences, and a sentence is important if it has important words.

For all intent and purposes, the importance of the sentences and words are saved in two column vectors $\vec{s}_{u\times 1}$ and $\vec{w}_{v\times 1}$ respectively, with the initial value of all the elements in the two vectors set to 1, in accordance with the assumptions made by Wan et al. The two statements mentioned above now directly gives us the following expressions about the importance of words and sentences:

$$w_j \propto \sum_i \tilde{R}_{ij} w_i \text{ and } s_i \propto \sum_j \tilde{Q}_{ji} s_j \tag{6}$$

$$w_j \propto \sum_i \hat{W}_{ji} s_i \text{ and } s_i \propto \sum_j \tilde{W}_{ij} w_j \tag{7}$$

The iterative forms for the importance of words and sentences are therefore obtained as,

$$w_j = \alpha \sum_i \tilde{R}_{ij} w_i + \beta \sum_i \hat{W}_{ji} s_i \tag{8}$$

$$s_i = \alpha \sum_j \tilde{Q}_{ji} s_j + \beta \sum_j \tilde{W}_{ij} w_j \tag{9}$$

where α and β are the relative contributions of the words and sentences to the importance of specific and respective words and sentences such that $\alpha+\beta=1$. Equal contribution of words and sentences to the importance values are assumed so that $\alpha=\beta=0.5$. Equations (8) and (9) are then applied in iterations to improve the score of the sentences or the words, whose convergence is determined by comparing the previous value with the new value. If the difference between the importance values in two consecutive iterations for every single word and sentence is below a threshold of 0.0001, the iteration stops to obtain the final score of the word or sentence saved in $\vec{s}_{u\times 1}$ and $\vec{w}_{v\times 1}$ respectively.

3.7 LDA Topic Allocation and Keyword Selection

Going beyond the basic keyword extraction method stage, LDA is modified to work in a single document level instead of a corpus level. The corpus fed into the LDA model is sent as a collection of sentences that form a document instead of a collection of documents that form a corpus in order to enhance post-processing of the candidate keyword rankings

and identify latent topic words in a single document and select keywords that cover the most number of them. The LDA model is specified to assign the words in the document within ten unlabeled random topics it generates from a Dirichlet distribution by associating a probability value for each word in each of ten topics to create word-topic probability distributions using a Bayesian inference algorithm. Each word in the document is labeled with a whole number by the model in alphabetical order to form a dictionary. Each sentence is then composed of a set of tuples in the form of (word label, corresponding importance value) for each unique pre-processed word it contains with the importance value obtained from above phases of keyword extraction replacing the word-frequency in the sentence that a true LDA model uses. The corpus fed into the model is now a collection of sentences or bags of words in the aforementioned format. The basic algorithm to evaluate the probability distribution is as follows:

- Go through each sentence 's' and randomly assign each word in the sentence to one of n=10 topics.
- For each sentence 's', go through each word 'w' and calculate:
  - p(topic t | sentence s) = aggregate proportion of importance values of words in sentence 's' that the current word belongs to that are assigned to topic 't' except that word. (If a lot of words in sentence 's' belong to topic 't', it is possible that the word 'w' belongs to topic 't'.)
  - p(word w| topic t) = proportion of importance values of assignments to topic 't' that come from this word 'w' (to try to capture how many sentences are in topic 't' because of word 'w').
- Update probability of word 'w' belonging to topic 't' using:

$$\begin{aligned} \text{p(word w in topic t)} &= \text{p(topic t | sentence s)} \\ &\times \quad \text{p(word w | topic t)} \end{aligned} \tag{10}$$

The model can generate word-topic distribution for every word to belong in every single topic because there is a probability value associated for each of the scenario. A word can belong to one topic with high probability making it significant to the topic while it could have very low probability to belong to another topic and hence can be discarded from even belonging to the latter topic at all. Additionally, a word can belong to more than one different topics with high probability to be considered relevant to those topics and hence is very important to our task of generating topic-wide keywords and should be preserved. To cull low word-topic probability associations, the LDA model is tasked in the input phase with a parameter specifying number of words to assign in each topic to only produce result of word-probability distributions in each topic containing 10% of total unique words with the highest values. This way, the words that belong to multiple topics with high probabilities can be easily visualized from which the topic-wide keywords can be picked out by counting the words that are assigned to the most number of topics. 10% of such words are selected to be the output as the final topic-wide keywords required.

4. Experiments and Results

The above methodology was experimented on Nepali News Dataset with each individual document producing 10% of its total unique words as topic-wide keywords. A total of 24,262 keywords were generated from the 4000 documents fed into the model. The output was compared against human-assigned keywords obtained by requesting individuals in Nepali language based teaching departments from various fields specified to the task with the same parameters like the number of keywords from each document (i.e., close to 10% of total words). To compare the keywords, the human-generated keywords were also pre-processed.

4.1 Evaluation Metrics

For evaluation of the model's performance, precision-based parameters are used to determine how well the generated output reflects the human-assigned keywords, as is often used my majority of the researchers in keyword generation. These are precision, recall, and F-measure, given by:

$$\text{precision} = \frac{n(KW_{\text{common}})}{n(KW_{\text{extracted}})} \tag{11}$$

$$\text{recall} = \frac{n(KW_{\text{common}})}{n(KW_{human-assigned})} \tag{12}$$

$$\text{f-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{13}$$

where $n(KW_{\text{common}})$ is the number of keywords that is common between the model-generated and human-assigned keywords, $n(KW_{\text{extracted}})$ is the total number of keywords produced by the model, and $n(KW_{\text{human-assigned}})$ is the total number of human-assigned keywords. Precision defines how well the human-assigned results are projected into the model-generated results. Similarly, recall shows how well the output of the model represents the original or human-assigned results.

4.2 Evaluation results

The results of topic-wide keyword generation are evaluated against three base architectures in the current research space: the TF-LDA approach, TextRank extraction, and Ying et al. Here's a short introduction to each extraction methods:

TF-LDA Instead of using the importance values in the LDA model corpus, the simplest form of keyword extraction using LDA uses frequency of word in each sentence directly without the use of any graph-based extraction phase. The word-topic distributions are generated with the same algorithm mentioned in section 3.7, using proportion of word frequencies instead of the aggregate proportion of their importance values.

TextRank This method projects the document into a single undirected word-to-word graph with the nodes representing the words and the edge between two nodes representing the semantic connection between the connected words. It then uses PageRank on the graph by considering it a directed graph with equal in-degree and out-degree for each vertex. Thus, the importance of each word $WI(w_i)$ depends on the importance of all the words that connect to it and is calculated using:

$$WI(w_i) = (1-d) + d \times \sum_{w_j \in Adj(w_i)} \frac{w_{ji}}{\sum_{w_k \in Adj(w_j)} w_{jk}} WI(w_j) \tag{14}$$

where w_{ji} is the semantic relationship between words represented by w_i and w_j which is determined by the co-occurrences in a sliding window of various sizes. $Adj(w_i)$ is the set of nodes that connect to word w_i, and d is the damping factor that is set to be 0.85 as is standard since PageRank.

Ying et al.[13] In an effort to cover all the topics covered in the document, this paper applies K-means clustering on the graphs after the iterative importance calculations converge. A set of clusters of words belonging to each supposed topic is obtained, and the keywords are selected as the centroid word for each cluster with the importance values in consideration. The value of k is kept 10 in accordance to the one used during evaluation in their paper for one of their datasets.

The evaluation results against all three approaches are displayed in table 1.

Table 1: Comparing Results on Nepali News Dataset

System	Precision	Recall	F-measure
TF-LDA	0.395	0.447	0.412
TextRank	0.347	0.527	0.415
Ying et al.	0.408	**0.537**	0.460
This Method	**0.461**	0.462	**0.462**

As seen from the table, there is a significant improvement in the results from the baseline approaches in terms of precision and f-measure. Precision is the most important factor where it is preferred to have as much of the human-assigned keywords to be reflected on the topic-wide keywords as possible. Recall is the factor that skews significantly due to the variation in the number of keywords generated by each of the other methods. Ying et al produced 20% of the total words as the output and thus had a higher chance of including as many of the human-assigned keywords in the output as possible. Similarly with TextRank, it generated output based on the number of total words in the document after pre-processing while counting repeated words as well, instead of using only unique words mentioned in this paper. This also creates a higher chance of getting more words from the human-assigned keyword lists.

5. Conclusion and Future Works

This paper introduces LDA in addition to graph-based methods of keyword generation to produce topic-wide keywords i.e., keywords that reflect on as many underlying topics in the document as possible. As many state-of-the-art approaches were used as possible in each stage like La-BSE word and sentence embedding, three-graph approach as opposed to one, and inclusion of importance of sentences in calculation of word importance as well. Everything was done on a document level so that it does not depend on reference to any other document or labels to evaluate keyword importance. The results show better representation of topic-wide keywords.

Since documents needed to interpret are generally lengthy in this paper, LDA might not be the go-to topic modeling approach for smaller texts like social media posts and product reviews generally containing less than 10 sentences. Non-probabilistic approaches like Top2Vec and BERTopic are powerful tools for short text that do not require the number of topics that can be used on top of graph-based approach to meet the same need in the future. For longer texts, other probabilistic approaches like NMF (non-negative Matrix Factorization) can be used as an alternative for word-topic distribution.

Acknowledgments

This paper was supported by the Electronics and Computer Engineering Department at Pashchimanchal Campus, IOE, TU, Nepal, with direct and indirect guidelines. Credit goes to all the Nepali news portals from where the documents were extracted for provision of a large assortment of data. All the teachers involved in producing human-assigned keywords are much appreciated for their time and effort. All the papers and tools referenced are highly praised for their research into their respective fields.

References

[1] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

[2] Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd, et al. The pagerank citation ranking: Bringing order to the web. 1999.

[3] Tej Bahadur Shahi and Ashok Kumar Pant. Nepali news classification using naive bayes, support vector machines and neural networks. In *2018 international conference on communication information and computing technology (iccict)*, pages 1–5. IEEE, 2018.

[4] Sushil Ghimire. Sushil79g/nepali_nlp: A python based library for nlp in nepali language, 2020.

[5] Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic bert sentence embedding. *arXiv preprint arXiv:2007.01852*, 2020.

[6] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 552–559, 2007.

[7] Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. Simple

unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470*, 2018.

[8] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507*, 2017.

[9] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[10] Ashkan Kazemi, Verónica Pérez-Rosas, and Rada Mihalcea. Biased textrank: Unsupervised graph-based content extraction. *arXiv preprint arXiv:2011.01026*, 2020.

[11] Gu Yijun and Xia Tian. Study on keyword extraction with lda and textrank combination. *Data Analysis and Knowledge Discovery*, 30(7):41–47, 2014.

[12] Hao Sun, Bing Li, and Bo Han. A novel keyphrase extraction method by combining fp-growth and lda. In *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 1764–1768. IEEE, 2017.

[13] Yan Ying, Tan Qingping, Xie Qinzheng, Zeng Ping, and Li Panpan. A graph-based approach of automatic keyphrase extraction. *Procedia Computer Science*, 107:248–255, 2017.

[14] Hung Du, Srikanth Thudumu, Antonio Giardina, Rajesh Vasa, Kon Mouzakis, Li Jiang, John Chisholm, and Sanat Bista. Contextual topic discovery using unsupervised keyphrase extraction and hierarchical semantic graph model. *Journal of Big Data*, 10(1):156, 2023.