Expert Systems With Applications 140 (2020) 112876

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Complex Network based Supervised Keyword Extractor

Swagata Duari*, Vasudha Bhatnagar

Department of Computer Science, University of Delhi, New Delhi 110007, India

ARTICLE INFO

Article history:
Received 23 April 2019
Revised 5 July 2019
Accepted 15 August 2019
Available online 16 August 2019

Keywords:
Supervised keyword extraction
Complex network
Graph-theoretic node properties
Text graph.

ABSTRACT

In this paper, we present a supervised framework for automatic keyword extraction from single document. We model the text as complex network, and construct the feature set by extracting select node properties from it. Several node properties have been exploited by unsupervised, graph-based keyword extraction methods to discriminate keywords from non-keywords. We exploit the complex interplay of node properties to design a supervised keyword extraction method.

The training set is created from the feature set by assigning a label to each candidate keyword depending on whether the candidate is listed as a gold-standard keyword or not. Since the number of keywords in a document is much less than non-keywords, the curated training set is naturally imbalanced. We train a binary classifier to predict keywords after balancing the training set.

The model is trained using two public datasets from scientific domain and tested using three unseen scientific corpora and one news corpus. Comparative study of the results with several recent keyword and keyphrase extraction methods establishes that the proposed method performs better in most cases. This substantiates our claim that graph-theoretic properties of words are effective discriminators between keywords and non-keywords. We support our argument by showing that the improved performance of the proposed method is statistically significant for all datasets. We also evaluate the effectiveness of the pre-trained model on Hindi and Assamese language documents. We observe that the model performs equally well for the cross-language text even though it was trained only on English language documents. This shows that the proposed method is independent of the domain, collection, and language of the training corpora.

1. Introduction

Keywords are special words that are typically embedded in documents and provide a compact and precise representation of the document content. Author-specified keywords for research articles and blogs not only convey the topics that the document covers, but are also used by search engines and document databases to efficiently locate information. *Keywords* are also used for categorizing and clustering stories in news industry, document summarization, and recommendations. Keywords can also aid in constructing titles for articles, assigning tags to blogs, and so on.

Not all documents on the Web, however, are accompanied by keywords assigned by authors, in which case important and relevant terms have to be extracted from the document itself. Inundated with the massive volume of digital documents available on the Internet, it is in-feasible to manually extract keywords. Consequently, NLP researchers continually strive towards improving automated methods for *keyword extraction*(KE). Keyphrase extraction is considered as an extension of the keyword extraction task (Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015).

Arising from the problem of automated index generation (Luhn, 1957), earliest keyword extraction techniques used statistical methods (Herrera & Pury, 2008; Ortuno, Carpena, Bernaola-Galván, Muñoz, & Somoza, 2002), which begot the advantage of language and domain independence. With recent popularity of machine learning approaches, supervised and unsupervised methods for keyword extraction have been in forefront of the research arena (Boudin, 2013; Bulgarov & Caragea, 2015; Mothe, Ramiandrisoa, & Rasolomanana, 2018). Supervised learning methods basically *identify* the keywords (or keyphrases) by modeling the problem as binary classification task, while unsupervised methods *extract* keywords by quantifying and ranking the words' *embedded-ness* in text.

Though enrichment of features in supervised approaches and growing sophistication in techniques have achieved enhanced performance, inadvertently the methods have promoted fixation for

* Corresponding author.
E-mail addresses: sduari@cs.du.ac.in (S. Duari), vbhatnagar@cs.du.ac.in (V. Bhatnagar).

https://doi.org/10.1016/j.eswa.2019.112876
0957-4174/

document structure, language, domain, and collection. Several state-of-the-art supervised algorithms for keyword extraction fail to accommodate a generic design because of one of the following three reasons. First, they require linguistic knowledge and hence are dependent on the language tools (for example the works of Chuang, Manning, & Heer, 2012; Hulth, 2003; Nguyen & Kan, 2007; Zhang, 2008). These methods generate language-dependent features that are specific to the language of the training set.

Second, most of the existing supervised algorithms are domain dependent (Caragea, Bulgarov, Godea, & Gollapalli, 2014; Kim, Medelyan, Kan, & Baldwin, 2010; Nguyen & Kan, 2007). For example, citations-enhanced keyword extraction (Caragea et al., 2014) works only when citation information is available. Thus, such techniques work well for scientific domain, but are not suitable for a generic domain that contains texts from news articles, blog articles, meeting transcripts, etc. Nguyen and Kan (2007) extracted keyphrases from scientific papers by enriching the feature set with morphological information found in scientific text, which is also an example of domain-dependent keyphrase extraction.

Third, existing supervised KE methods are collection-dependent because they use statistical features that are derived from the document collection (Caragea et al., 2014; Chuang et al., 2012; Nguyen & Kan, 2007; Sterckx, Demeester, Develder, & Caragea, 2016). Frequency-based statistical features like tf-idf, positions of occurrence, etc. are collection sensitive, and they change drastically with a slight alteration of the training set.

In addition to above three primary reasons, some algorithms require external information from sources like Wikipedia (Medelyan, Frank, & Witten, 2009; Zhang et al., 2017) or expert knowledge in the form of label-distribution to incorporate hints (e.g. a noun word occurring in the title) (Gollapalli, Li, & Yang, 2017). This leaves a research gap for generic keyword extractor that can be applied on any text without considering its language, domain, or corpora. Recognizing this gap, we investigate the feasibility of designing a keyphrase prediction model that is domain-, language-, and collection-independent.

Graph-based unsupervised KE methods represent text as graph,[1] and rely on the node properties to discriminate between keywords and non-keywords (Florescu & Caragea, 2017; Litvak, Last, Aizenman, Gobits, & Kandel, 2011; Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015). These methods process one document at a time and are autonomous, which makes them collection and domain agnostic. However, these methods are dependent on the language tools as they perform POS tagging[2] for identifying candidate keywords (nouns and adjectives) (Florescu & Caragea, 2017; Litvak et al., 2011; Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015; Tixier, Malliaros, & Vazirgiannis, 2016). Because of this reason, graph based KE methods are not pliable for texts in resource-poor languages. It is noteworthy that unsupervised methods often report lower performance as compared to their supervised counterparts.

In this research, we aim to bolster performance of supervised learning approach with the advantages of graph-based keyword extraction methods, sans the bias towards domain or collection underlying the training data. The idea is inspired by consistent success of graph-based KE methods, which are typically unsupervised and weak performers compared to their supervised counterparts. We build over the domain and collection independence of graph-based KE methods and use graph-based node properties as features to develop a supervised model with improved performance. Additionally, we eliminate the language dependency by using statistical properties to filter candidate keywords from the text. Specific contributions of our research are listed below.

1. We devise supervised learning approach for automatic keyword extraction using graph-theoretic feature set (Sections 3–6).
2. We empirically validate our claim that the method is domain-, and collection-independent (Sections 7 and 8).
3. Post keyword extraction, we generate keyphrases from the predicted keywords and demonstrate that our method performs comparably with the state-of-the-art supervised keyphrase extraction approaches (Section 8.3).
4. We evaluate the performance of our proposed method on texts from two India languages to establish language independence of the model (Section 8.4).

We do not delve into sophisticated deep learning based methods due to the limited volume of training set we have, and time required for training the model. We proceed with classic and simple classifiers as a proof of concept, and believe that use of deep learning techniques will enhance the performance of the predictive model.

2. Related works

Existing supervised methods for automatic keyword extraction tackle the problem as a phrase-based binary classification task, where keyphrases (*n*-grams) are extracted from the documents (Caragea et al., 2014; Hulth, 2003; Nguyen & Kan, 2007; Sterckx et al., 2016; Turney, 1999; Witten, Paynter, Frank, Gutwin, & Nevill-Manning, 1999). These methods first create a labelled training set by constructing features for candidate phrases (or words) in the text and designate each phrase as either positive (keywords) or negative (non-keywords) by consulting the associated gold-standard list. The training set thus created is used to induce a predictive model, which predicts word (or phrase) from unseen documents as keyword or non-keyword. Several algorithms for inducing a predictive model have been explored, including CRF and SVM (Zhang, 2008), Bagged decision tree (Medelyan et al., 2009), Naïve Bayes (Caragea et al., 2014; Sterckx et al., 2016), etc.

Since eliciting good quality features is crucial for performance of the trained model, feature construction is recognized as the focal task in creation of training set for supervised KE approaches. Wide variety of features have been proposed to obtain high quality training set for inducing well performing models, e.g., tf-idf, POS tags, n-gram features, etc. Hulth (2003) reported that adding certain linguistic knowledge (e.g., syntactic features) to the training set improves performance of the automatic keyword extractor as compared to relying only on statistics-based features such as, term frequency, n-grams, etc. Nguyen and Kan (2007) used morphological features of text in the training set in addition to simple statistics-based features, and designed a keyword extractor for scientific articles. Medelyan et al. (2009) incorporated information from external sources like Wikipedia to improve the training set. In addition to these, structural features of the document (Lopez & Romary, 2010a), knowledge about domain and collection (Caragea et al., 2014; Nguyen & Kan, 2007), citation-information (Caragea et al., 2014), incorporating expert knowledge (Gollapalli et al., 2017), and multidimensional information (Zhang et al., 2017) are some popular methods for enriching the training set.

Unsupervised KE techniques largely comprise graph-based methods, which transform the text into a graph (complex network) and use graph-theoretic properties to rank keywords. These methods are largely word-based (i.e. unigrams are extracted) (Duari & Bhatnagar, 2019; Rousseau & Vazirgiannis, 2015; Tixier et al., 2016), with a few being phrase-based (i.e. *n*-grams are extracted)

[1] Alternatively, complex network. We use the terms 'graph' and 'network' interchangeably.

[2] Part-of-speech tagging.

(Florescu & Caragea, 2017; Mihalcea & Tarau, 2004). Node properties like PageRank (Mihalcea & Tarau, 2004), PageRank along with position of the word in text (Florescu & Caragea, 2017), degree centrality (Litvak et al., 2011), coreness (Rousseau & Vazirgiannis, 2015), etc. have been studied extensively in the past. Network representation of the text leverages unsupervised keyword extraction methods because of their independence from the influence of domain of the document or corpus. We aim to overcome the domain and collection dependence of supervised KE methods by using graph-based node properties as features for training. Furthermore, we also overcome the problem of language dependency by using a statistical filter for candidate selection while maintaining the efficacy of supervised KE methods.

3. Methodology

Graph-based approaches for keyword extraction established that keywords possess certain properties, which impart special character to them. We hypothesize that succinct properties of keywords are revealed when the text is presented as graph. These properties are effective signals to discriminate between keywords and non-keywords. Accordingly, we employ node properties in the graph representation of text as features to fortify against dependence on linguistic, domain, collection, or structural features of the document. We propose a supervised framework to extract keywords from single document, which consists of the following steps.

1. Select candidate keywords from each document, and construct the corresponding graph-of-text (Section 4).
2. Extract select node properties as features from each graph-of-text (Section 5).
3. Prepare the training set and induce a predictive model (Section 6).

Steps (i) and (ii) harbour innovative approaches that are detailed below. We use the model induced in step (iii) to predict keywords from unseen documents.

4. Modeling text as complex network

Text is modeled as a complex system, where the basic units, i.e. words, interact among each other to bring out the ideas that the author intends to communicate. The interaction between words can be mapped using various relationships, such as statistical, semantic, syntactic, discourse, cognitive, etc. (Blanco & Lioma, 2012). The most frequently used relation for automatic KE systems is co-occurrence based statistical relation (Florescu & Caragea, 2017; Litvak et al., 2011; Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015; Tixier et al., 2016).

We use a parameter-free and language-agnostic approach for creating complex networks from text as proposed in our previous work (Duari & Bhatnagar, 2019). The network representation of text is created by - (i) selecting a subset of words from the text as candidates (Section 4.1) and (ii) using these candidate keywords as nodes, and forging relationships between nodes to create edges (Section 4.2). We briefly describe the method below.

4.1. Selecting candidate keywords

In order to reduce the search space for possible keywords, we first eliminate frequently used non-content bearing words from processing. To do this, we perform *stopword removal* using a standard English stop words list[3] For non-English texts, a custom-curated stopwords list can be adopted to suit the requirement. We then apply a filter to identify candidate keywords from the remaining words. We use σ-index (Ortuno et al., 2002) as a statistical filter to perform this task.

The σ-index of a word computes normalized standard deviation of the word's spacing distribution in successive occurrences, with higher values of σ-index indicating higher term relevance (Ortuno et al., 2002). We adopt Herrera and Pury (2008) implementation of σ-index, where the σ-index of a word w in a document D is defined as below.

Let, $N = |D|$ be the document length, n be the number of occurrences of w, and p_i be the position of ith occurrence of w. Note that $p_0 = 0$ and $p_{n+1} = N + 1$. Then $\sigma(w)$ is computed as:

$$\sigma(w) = \frac{s(w)}{\mu(w)}, \tag{1}$$

where $\mu(w) = \frac{\sum_{i=0}^{n}(p_{i+1}-p_i)}{n+1} = \frac{N+1}{n+1}$ is the average distance between successive occurrences of w and $s(w) = \sqrt{\frac{\sum_{i=0}^{n}((p_{i+1}-p_i)-\mu(w))^2}{n-1}}$ is the standard deviation of word occurrences. We retain top-33% words ranked by σ-index as candidate keywords, which drastically reduces the search space to one-third of the original length.

Conventional graph-based keyword extraction methods use part-of-speech taggers and select nouns and adjectives as candidate keywords using linguistic tools (Florescu & Caragea, 2017; Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015). This makes these approaches dependent on the linguistic tools and inefficacious for resource-poor languages. The use of statistical filter, σ-index, in our proposed approach imparts language-independence to this phase, and thus makes the approach language agnostic.

Please note that the computation of σ-index requires a word to occur at least twice in the document. This does not conflict with our goal because a word that occurs exactly once is unlikely to be a keyword. Furthermore, as words in short texts do not occur frequently, we omit the computation of σ-index for documents with less than 100 unique words excluding stopwords. In such situation, each word is considered as a candidate keyword.

4.2. Network construction

We model text as a weighted, undirected graph $G = (V, E, W)$, where V is the set of vertices that comprises the candidate keywords, $E \in V \times V$ is the set of edges, and W is the corresponding weighted adjacency matrix. We use the conventional relationship of "co-occurrence" of words to define edges between the nodes. Two nodes (candidate words) are linked if they co-occur in a sliding window of user specified size (Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015). In order to eliminate the user parameter (window size), we slide the window over two consecutive sentences (Duari & Bhatnagar, 2019). This strategy begets advantages of capturing coherence in the flow of ideas that a sentence carries from its previous sentence. The links are weighted by the number of times the adjacent nodes (words) co-occur in the original text, and isolated nodes are ignored.

It is noteworthy that short texts (1–3 sentences) result into highly dense networks which are often complete graphs. Network density decreases as the number of sentences increases. Fig. 1 shows network of a short text containing three sentences.

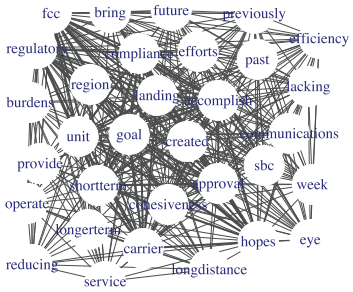
5. Extracting properties of keywords

Centrality of nodes in a network is a popular estimate of node importance. According to Boudin (2013), degree centrality is conceptually the simplest and computationally most efficient centrality measure. However, in the context of weighted graph-of-text, it

[3] http://www.lextek.com/manuals/onix/stopwords2.html.

SBC gets more serious on regulatory compliance
With one eye on the past and the other on its
future, SBC Communications last week created a
unit it hopes will bring a cohesiveness and
efficiency to its regulatory compliance
efforts that previously had been lacking. The
carrier also hopes the new regulatory
compliance unit will help it accomplish its
short-term goal of landing FCC approval. to
provide long-distance service throughout its
region, and its longer-term, goal of reducing
the regulatory burdens under which it and
currently operate.

(a) Sample text.



(b) Complex network of text.

Fig. 1. Network created from sample text in Fig. 1(a) (document id 6 from Hulth2003 dataset).

is more appropriate to use weighted degree (strength) as a measure of node importance (Barrat, Barthelemy, Pastor-Satorras, & Vespignani, 2004). Strength in our setting captures how often the words co-occur with each other in adjacent sentences.

Though strength effectively captures node importance, however, probability density distribution of strength for keywords and non-keywords for the training set prepared during our study clearly shows overlapping areas near high strength values (Fig. 2(a)). The overlap indicates that strength alone is not an accurate discriminator between keywords and non-keywords. Next two plots (Fig. 2(b) and (c)) show similar observations for Eigenvector centrality and PageRank. This impels exploration of other node properties - Coreness, PositionRank, and Clustering Coefficient - which would aid improvement in the quality of extracted keywords. It is noteworthy that we avoid centrality measures that require expensive all-pair shortest path computations. This maintains the frugality of feature extraction phase, enabling its potential for online usage.

All these properties, except Clustering Coefficient, have been individually vetted by state-of-the-art unsupervised graph-based keyword extraction methods. Our goal in this work is to investigate the complex interplay of these properties, which to the best of authors' knowledge, has not been explored for discriminating between keywords and non-keywords. We describe each of the node properties below.

5.1. Strength of a node

Strength (weighted degree) of a node measures its *embeddedness* at local level. For node v_i in a weighted network G, it is computed as (Barrat et al., 2004):

$$strength(v_i) = \sum_j w_{ij} = \sum_j w_{ji}$$

Here, w_{ij} is the corresponding entry in the weight matrix W for edge (v_i, v_j). High strength is associated with a node being more central or important in the network. The indulging intuition is that a word which is co-occurring with many other words (i.e., has high degree/strength) is important and is likely to be a keyword.

5.2. Eigenvector centrality

Eigenvector centrality or *Prestige* of vertex v_i quantifies its *embedded-ness* in the network while recursively taking into account the prestige of its neighbors. Starting with initial prestige vector $\vec{\boldsymbol{p}}_{\boldsymbol{0}}$ where all nodes (words) are assigned equal *prestige*, $\vec{\boldsymbol{p}}_{\boldsymbol{k}}$ is computed recursively as follows till convergence is achieved (Zaki, Meira Jr, & Meira, 2014).

$$\vec{\boldsymbol{p}}_{\boldsymbol{k}} = W^T \vec{\boldsymbol{p}}_{\boldsymbol{k-1}} = (W^k)^T \vec{\boldsymbol{p}}_{\boldsymbol{0}}$$

According to this computation, a word is important if it co-occurs with other important words. Nodes with higher eigenvector centrality are perceived as more important. Effectively, prestige of a word measures its influence over the entire document.

5.3. PageRank

PageRank computes prestige in the context of 'random surfer model' of Web search. A *damping factor*, which is the probability of the surfer making *random jump*, is used here. In case of text documents, this can be interpreted as events like the change of discourse in the document, beginning of a new chapter in a book, etc. We adopt the computation of word score (WS) from TextRank algorithm (Mihalcea & Tarau, 2004), as given below.

$$WS(v_i) = (1-d) + d * \sum_{v_j \in N_i} \left(\frac{w_{ji}}{\sum_{v_k \in N_j} w_{jk}} WS(v_j) \right)$$

Here, damping factor d is set to 0.85 by the algorithm, which is the probability of random jump in context of the random surfing model. N_i and N_j are the sets of adjacent nodes of node v_i and v_j, respectively. Mihalcea and Tarau (2004) expressed that the damping factor associated with random surfer model can be interpreted as text cohesion or "knitting" of discourse together.

5.4. PositionRank

PositionRank is an extension of PageRank that is based on the intuition that keywords are likely to occur towards the beginning of the text rather than towards the end. PositionRank favors words occurring at the beginning of the document as keywords by using a position-biased weight for each candidate (Florescu & Caragea, 2017). Node $v_i \in V$ is assigned a weight based on its positional information by taking the inverse of the sum of its positions of occurrences in the text. Subsequently, PageRank computation is performed on the weighted nodes of the network to yield PositionRank scores for the candidate words. Formally, the PositionRank score of a node v_i is computed as follows.

$$S(v_i) = (1-\alpha).\tilde{p}_i + \alpha. \sum_{v_j \in N_i} \left(\frac{w_{ji}}{\sum_{v_k \in N_j} w_{jk}} S(v_j) \right)$$

Here, α is set as 0.85 by the algorithm, $\tilde{p}_i = \frac{p_i}{\sum_{j=1}^{|V|} p_j}$ is the normalized positional weight of v_i, N_i is the set of adjacent nodes of v_i, and w_{ji} is the weight of edge e_{ji}. The bias vector $\tilde{p}_i$ ensures that

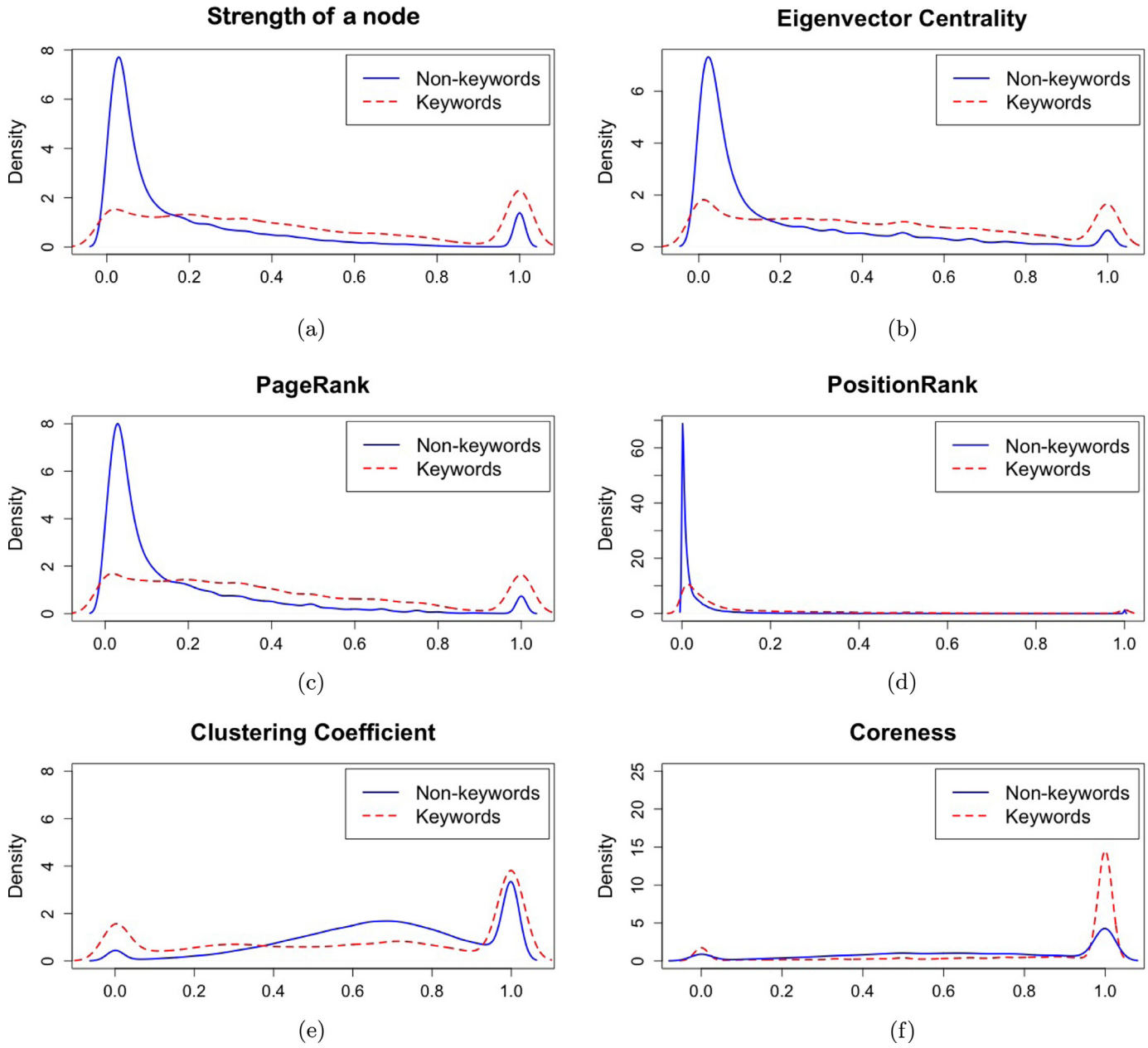


Fig. 2. Density distributions of graph node properties for keywords and non-keywords using the SMOTE-balanced training set.

words occurring towards the beginning are preferred as keywords by the system.

5.5. Coreness

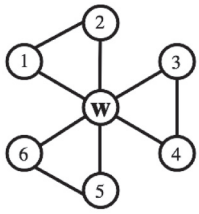
Coreness is a network degeneracy property that decomposes network G into a set of *maximal connected* subgraphs G_k (k denotes the core), such that nodes in G_k have degree at least k within the subgraph and $G_k \subseteq G_{k+1}$ (Seidman, 1983). *Coreness* of a node is the highest core to which it belongs. Rousseau and Vazirgiannis (2015) presume that words in the main (highest) core of the network are keywords due to their dense connections. Though our findings differ where we have empirically observed that main core typically consists of fewer keyword-quality nodes that results in increasing precision and dropping recall (Duari & Bhatnagar, 2019), we are convinced that keywords tend to lie in higher cores. Hence, we choose to include *coreness* as a discriminating property.

5.6. Clustering Coefficient

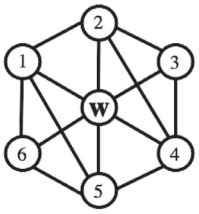
Clustering Coefficient (CC) of a node indicates edge density in its neighbourhood. It is a local property and is computed for node v_i as the ratio of actual number of edges in the sub-graph induced by v_i (excluding itself) to the total number of possible edges in that subgraph (Zaki et al., 2014). A node v_i having high clustering coefficient implies that the neighbors of v_i are densely connected to each other, and can convey a context effectively without involving node v_i. For an undirected graph G, clustering coefficient of node $v_i \in G$ is computed as below.

$$CC(v_i) = \frac{2|e_{jk} : v_j, v_k \in N_i, e_{jk} \in E|}{|N_i|(|N_i| - 1)}$$

Here, N_i is the set of nodes adjacent to v_i. We conjecture that nodes (words) with low clustering coefficient connect diverse con-



(a) Three semantically unrelated contexts, glued together by vertex w. CC for w is 0.20 here.



(b) Semantically related contexts, causing higher clustering coefficient for vertex w. CC for w is 0.53 here.

Fig. 3. Effect of semantically related and unrelated contexts on Clustering Coefficient.

texts together, and thus are likely to be important words. We elaborate the idea below.

A closed triad is formed in a graph of text when three words co-occur either in the same sentence or in adjacent sentences. Semantically, the words in the triad share a context. If a word w shares many unrelated contexts with several words (Fig. 3(a)), then w attains importance because it glues several independent contexts. On the other hand, if the contexts in which w participates are linked as shown in Fig. 3(b) (e.g., contexts formed by vertices 1,2,3 and vertices 1,4,5 are connected via an edge between vertices 2 and 4), then the word may not be important.

For unweighted networks, the above definition of topological clustering coefficient (CC) holds. However, for weighted networks, Barrat et al. (2004) define weighted clustering coefficient (WCC) that incorporates edge weights into computation. Since our networks are weighted, we empirically evaluated the effect of WCC against CC in distinguishing keywords from non-keywords. However, though WCC is apparently a better option, in our case the performance of the models degraded when using WCC in place of CC. This is because incorporating edge weights sometimes increases the clustering coefficient for the node, which negatively correlates with the importance of the node. Thus, we decided to use topological CC instead of WCC as a network property in our experiments.

Overlapping of densities of the six node property values in Fig. 2 indicate high number of false positives and likely poor performance. However, an intricate coaction of all six properties produces desirable effect of improving prediction performance, which has been established in Section 8.

6. Inducing the model

The construction of training set is guided by our conjecture that the distribution of network-centric properties of the keywords are similar irrespective of the language, domain, or collection of the document. Accordingly, we combine short scientific abstracts from Hulth2003 dataset and long scientific articles from SemEval2010 collection to create the training collection. The objective is to predict keywords from documents belonging to different collections of scientific papers, news articles, and non-English texts using the same predictive model.

For each document in the training collection, we consult the corresponding gold-standard keywords list and assign the class label as 'positive' or 'negative' to the candidate words depending on whether they are listed as a gold-standard keyword (as unigram) or not. Corresponding feature values for the candidate keywords, as described in Section 5, are range normalized to remove the bias due to document length. The feature set along with the labels constitute the training set for our empirical evaluations.

The curated training set naturally has high imbalance of class distribution because keywords are much lesser in number than other words. Longer documents in the training set contribute more to imbalance than shorter ones. Our pragmatism of using judicious mix of long and short text paid-off, resulting into the training set exhibiting an imbalance ratio of 1:5 (keywords:non keywords). We use Weka implementation of SMOTE (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) to balance[4] the training set.

We use Naïve Bayes (NB) and XGBoost classifiers to train the model following their success as reported in earlier works. NB has been used for predicting keywords and keyphrases in various earlier studies (Caragea et al., 2014; Kim et al., 2010; Medelyan et al., 2009; Nguyen & Kan, 2007). We decided to use Naïve Bayes classifier because of its simplicity, interpretability, and fast execution time. We additionally use a gradient boosted decision tree implemented in the XGBoost package (XGBoost classifier) following its success in predicting keyphrases as reported by Sterckx et al., who note that XGBoost classifier outperforms NB and linear classifiers in their study (Sterckx et al., 2016).

We attempt to reduce the high bias factor of NB classifier by balancing the training set using artificially generated data to oversample the minority class. We additionally experiment with Bagging and Boosting ensembles of NB classifier to inspect for improvement in performance due to ensemble learning. Use of classical classifiers shows a marked improvement in performance in our experiments. We envisage that the performance will be further boosted by use of deep learning methods if sufficiently large dataset and efficient computation power is available.

7. Experimental setup and objectives

The proposed framework is implemented using R (version 3.3.1) and relevant packages[5] (`igraph, tm, RWeka, caret and pROC`). We use six publicly available collections that have been used in similar studies. Each document in these collections is accompanied by an associated gold-standard keywords list, which is used as ground truth for testing the classifier performance. In the following sections, we briefly describe the datasets used in this study (Section 7.1) and the objective and design of our experiments (Section 7.2).

7.1. Datasets

We use six publicly available datasets in our experiments. The datasets are described in detail below.

[4] We set 'percentage' parameter to 200% for SMOTE filter.

[5] https://cran.r-project.org/web/packages/.

Table 1
Overview of the experimental data collections. |D|: Number of docs, L_{avg}: average doc length, N_{avg}: average gold-standard keywords per doc, K_{avg}: average gold-standard keyphrases per doc, KP_{avg}: average percentage of keywords present in the text, *ngram%*: average %age of 1/2/3/3+ -gram distribution.

Collection	|D|	L_{avg}	N_{avg}	K_{avg}	KP_{avg}	ngram%
Hulth2003 (Hulth, 2003)	1500	129	23	10	90.07	16/52/24/8
WWW (Caragea et al., 2014)	1248	174	9	5	64.97	31/51/16/1
KDD (Caragea et al., 2014)	704	204	8	4	68.12	25/58/16/1
Marujo2012 (Marujo et al., 2012)	450	427	69	48	99.31	75/17/5/2
Krapivin2009 (Krapivin et al., 2009)	2304	7961	11	5	96.91	19/63/16/2
SemEval2010 (Kim et al., 2010)	244	8085	34	16	95.89	21/55/20/4

1. Hulth2003 (Hulth, 2003) consists of 2000 scientific abstracts from *Inspec* dataset, which are further divided into training (1000 articles), test (500 articles), and validation (500 articles). Each article in Hulth2003 dataset is accompanied with two gold-standard lists - one is controlled and uses a thesaurus, and the other is uncontrolled. We combine the training and test collections from Hulth2003 (a total of 1500 articles) to form a part of the training set for our experiments, and consult the uncontrolled keywords list as a gold-standard.
2. WWW and KDD (Caragea et al., 2014) are two collections of abstracts from computer science articles published in the two well known conferences by the respective names. For both these collections, we consider only those articles which contains at least two sentences, and are accompanied by at least one gold-standard keyword.
3. Marujo2012 is a collection of 500 online news articles, which is grouped under training collection (450 articles) and test collection (50 articles). Each article is accompanied by a list of keywords assigned by human annotators through a HIT in Amazon Mechanical Turk (Marujo, Gershman, Carbonell, Frederking, & Neto, 2012). From this dataset, we use the articles under training collection (a set of 450 articles) as an *unseen test set* for our experiments.
4. Krapivin2009 (Krapivin, Autaeu, & Marchese, 2009) and SemEval2010 (Kim et al., 2010) are two datasets which contains full scientific articles from ACM. The Krapivin2009 dataset consists of 2304 articles and associated keywords lists. SemEval2010 consists of 284 articles, out of which 144 are grouped as training, 100 as test, and 40 as validation sets. Each document in SemEval2010 dataset is accompanied by three sets of keyword list - author-assigned, reader-assigned, and author-and-reader-assigned. We use the combined collection of 244 articles (training and test) as a part of the training set for our experiments, and we consult the author-and-reader-assigned keywords list as gold-standard.

Table 1 presents the datasets along with relevant statistics about the data. As mentioned in Section 6, we create the training set for our experiments by combining the articles from Hulth2003 (1500 articles) and SemEval2010 (244 articles) datasets.

7.2. Objectives and experimental design

We perform experimental evaluations to answer the following research questions.

1. *Which model performs best for automatic keyword extraction task?*
   We perform 10-fold cross-validation on the training set using XGBoost, Naïve Bayes, and Bagging and Adaboost ensembles of Naïve Bayes. To reduce the bias, we balance the training set using Weka implementation of SMOTE filter with percentage parameter set to 200%. Details of experiment and results are discussed in Section 8.1.
2. *How well do the graph-theoretic properties discriminate between the keywords and non-keywords over cross-collection and cross-domain datasets?*
   We use the best model trained in experiment 1 for all subsequent experiments. We perform cross-collection and cross-domain analysis of the trained model using three scientific datasets and one news corpus. Experimental results are discussed in Section 8.2.
3. *How does the quality of extracted keyphrases compare with those extracted by state-of-the-art supervised and unsupervised keyphrase extraction methods?*
   We generate keyphrases from predicted keywords as a post processing step, and compare the quality with those extracted by state-of-the-art supervised and unsupervised keyphrase extraction methods. Comparative evaluation of the methods are presented for each dataset in Section 8.3.
4. *How well does the model perform on cross-language documents?*
   To substantiate our claim of the model being language-independent, we use the model trained in experiment 1 to extract keywords from documents written in Indian languages. Section 8.4 presents a detailed discussion on this experiment.

Evaluation Metrics. We use precision, recall, and F1-score as performance evaluation metrics for all experiments pertaining to the above research questions. All three metrics are widely used in literature (Caragea et al., 2014; Hulth, 2003; Medelyan et al., 2009; Mihalcea & Tarau, 2004; Rousseau & Vazirgiannis, 2015; Sterckx et al., 2016; Zhang et al., 2017). Except for 10-fold cross-validation results in Table 2, all results presented in subsequent sections are macro-averaged at the dataset level.

8. Results and discussion

In this section, we present the results for our experiments as highlighted in Section 7.2. Each subsection is devoted to one task, and we support our claim with empirical evidences.

8.1. Establishing the best performing model

We trained four models on the SMOTE balanced training set each, using XGBoost (Chen & Guestrin, 2016), Naïve Bayes (NB), and Bagging and Adaboost ensembles of NB. We present 10-fold cross validation results in Table 2. Bold values represent best performance across all models in terms of the 'positive' class,[6] and values in italics represent second-best results for the same. Among all models, XGBoost trained on the balanced training set turns out

[6] Positive class is of interest to us, which represents keywords.

Table 2
Cross-validated classifier performance. XGB: model trained using XGBoost classifier, NB: model trained using NB classifier.

Models	P	R	F1
XGB	**75.39**	**79.93**	**77.59**
NB	72.4	49.71	58.95
NB+Bagging (NB-B)	*72.41*	49.71	58.95
NB+Adaboost (NB-A)	72.20	*53.42*	*61.41*

Table 3
Macro-averaged results of model performances on test sets. Models are trained on the SMOTE balanced training set.

Models	Hulth2003 Test			Semeval2010		
	P	R	F1	P	R	F1
XGB	49.8	**83.5**	**60.7**	*46.2*	**49**	**46.4**
NB	**52.8**	60.6	50.1	**46.4**	36.5	39.5
NB-B	**52.8**	60.5	50.1	**46.4**	36.5	39.5
NB-A	*52.4*	*63.5*	*51.8*	45.2	*39.2*	*40.6*

Table 4
Macro-averaged results for XGB and NB-A on unseen cross-collection datasets.

Test Sets	XGB			NB-A		
	P	R	F1	P	R	F1
Krapivin2009	21.6	**66.5**	30.9	**26.2**	61.7	**34.9**
WWW	14	**81.8**	23.1	**24**	66.3	**33.1**
KDD	13.6	**78.1**	22.3	**24.3**	70.6	**33.8**

to be the best model and Adaboost ensemble of NB is the second best.

Next, we test the performance of trained models on test sets from Hulth2003 and SemEval2010 collections. Table 3 shows macro-averaged results on the test sets. Bold values indicate best performance for corresponding test sets and values in italics indicate second-best results. We observe that XGB classifier performs best in terms of recall and F1-score for both test sets, whereas best performance in terms of precision is achieved by NB and NB-B classifiers. NB-A performs second-best in term of recall and F1-score, following XGB. Since the performance gap in precision between NB, NB-B, and NB-A are insignificant, we decided to retain NB-A model along with XGB for all further experiments.

8.2. Establishing domain and collection independence

With an aim to validate the claim of domain- and collection-independence, we evaluate XGB and NB-A on three unseen scientific datasets, i.e. Krapivin2009, KDD, and WWW, and one news corpus, i.e. Marujo2012. Recall that both models are trained on combined datasets from Hulth2003 and Semeval2010. Sections 8.2.1 and 8.2.2 present our results for establishing collection- and domain-independence, respectively.

8.2.1. Result on cross-collection datasets

Table 4 shows macro-averaged results for the models on the three cross-collection scientific datasets. We observe that the models are able to recall the keywords reasonably well from unseen documents across corpora. To the best of our knowledge, no earlier work on supervised KE has performed cross-collection investigation for keyword extraction. Hence we are unable to compare the performance.

Table 5
Macro-averaged results for XGB and NB-A on unseen cross-domain datasets.

Test Sets	XGB			NB-A		
	P	R	F1	P	R	F1
Marujo2012	58.3	**42**	**45.2**	**67.4**	29.8	37

Low precision for these datasets is due to the relatively less number of gold-standard keywords assigned per document (See column N_{avg} in Table 1). The models recall most of these keywords along with some false positives, which drops precision. NB-A outperforms XGB for these three datasets in terms of precision and F1-score, however with lower values for recall.

8.2.2. Result on cross-domain dataset

We perform experiments to establish domain-independence of the trained models by evaluating their performance on an unseen, cross-domain dataset of news articles (Marujo2012). We present our empirical observations in Table 5. It is evident from the results that the models are able to perform sufficiently on the cross-domain dataset, which establishes that the models are indeed applicable on documents from any domain. Again, we can't compare with any baseline due to reason stated in Section 8.2.1.

Interestingly, for the cross-domain dataset (i.e., Marujo2012), the models are able to extract keywords with high precision, albeit with a drop in recall. This is because of the relatively higher number of keywords assigned per document ($N_{avg} = 69$ in Table 1) for this dataset. The models tend to extract fewer but correct keywords, thus increasing precision and lowering recall in this case. XGB outperforms NB-A for this dataset in terms of recall and F1-score, whereas NB-A reports better precision.

8.3. Comparison with keyphrase extraction algorithms

State-of-the-art supervised KE methods are *phrase*-based extractors, whereas the unsupervised graph-based methods are *word*-based extractors. Several earlier works suggest that keyphrase extraction should be treated as an extension of keyword extraction, and not as a separate task (Mihalcea & Tarau, 2004; Papagiannopoulou & Tsoumakas, 2018; Rousseau & Vazirgiannis, 2015). Following this viewpoint, we generate candidate keyphrases from the text as a post-processing step considering only those words which are predicted as keywords by our model.

We pre-process the text to remove stopwords, and split at punctuation marks to get phrases. All unique phrases that are not sub-strings of other phrases are extracted as keyphrases. We apply stemming[7] using Porter stemmer[8] to both the gold-standard keyphrases and the extracted keyphrases to improve performance of the keyphrase extractor.

For all datasets except Marujo2012, we extract top-5, -10, and -15 keyphrases based on our observation in column K_{avg} in Table 1. For Marujo2012, we extract top-5 to top-30 keyphrases (in increments of 5) to account for the higher number of average keywords assigned per document.

We compare the performance of our models with the best in literature for each dataset. We report our observations for each dataset separately, because (i) results of all methods are not easily reproducible as their implementations are not publicly available, (ii) there is a diversity in choice of datasets for which the authors base their claims, and (iii) all state-of-the-art methods are not applicable across domain and corpora.

[7] This is an optional step and can be skipped if stemmer is not available.
[8] http://snowball.tartarus.org/algorithms/porter/stemmer.html.

Table 6
Performance evaluation for Keyphrase Extraction on Hulth2003 Test dataset. @*k*: evaluation results for top-*k* keyphrases.

Model	P	R	F1
XGB@10	**52.5**	**65.1**	**54.7**
NB-A@10	49.3	60.6	51.1
n-gram w. tag (Hulth, 2003)	25.2	51.7	33.9
TextRank (Mihalcea & Tarau, 2004)	31.2	43.1	36.2

We present our results in subsequent sections (8.3.1–8.3.5), comparing best performance of our models with select state-of-the-art methods evaluated on the datasets that we are using. We briefly explain these methods in subsequent sections and present comparative evaluation in the form of Tables. We also test the statistical significance of the improved performance of our algorithms over the baselines for each dataset (Section 8.3.6).

8.3.1. Result on Hulth2003 Test dataset

In this section, we evaluate the performance of our KE models on Hulth2003 dataset with the works of Hulth (2003) and Mihalcea and Tarau (2004). Hulth2003 dataset was curated by Hulth (2003) for her study of effect of linguistic properties in improving performance of keyword extractors. Later, this dataset has been mostly used by unsupervised keyword extraction methods (Duari & Bhatnagar, 2019; Rousseau & Vazirgiannis, 2015).

Hulth's work is supervised machine learning based, which uses linguistic information to improve performance. The method explores three term selection strategies - *n*-gram, noun-phrase (NP) chunk, and POS tag sequence, and evaluates the model performance on feature sets with and without POS tag information. Best result is obtained on POS tag based feature sets in comparison to their counterparts, and best F1-score is obtained with *n*-gram approach with POS tags as features.

Mihalcea and Tarau (2004) proposed an unsupervised approach, called TextRank, to extract keywords. The method is based on graph representation of text, where nouns and adjectives constitute the vertices set, and edges are formed between two vertices if they co-occur within a window of size *w*. Edges are undirected, and are weighted by the co-occurrence frequency of the adjacent vertices (words). PageRank (Brin & Page, 1998) computation is performed on the graph representation of text to rank the vertices in order of their keyword-ness, with high PageRank score associated with being more likely to be a keyword. The system then selects top one-third candidates as keywords.

We report our results in Table 6. It is clearly evident from the table that both XGB and NB-A outperform the baseline methods with large margin, with XGB leading in terms of precision, recall, and F1-score. Best result for both these models is obtained when we extract top-10 keyphrases, and XGB dominates NB-A on this dataset. It is noteworthy that the number of extracted keyphrases, i.e., 10 for Hulth2003 dataset, corresponds to the average number of keyphrases for the dataset as presented in Column K_{avg} of Table 1.

8.3.2. Result on KDD and WWW dataset

KDD and WWW datasets were curated by Caragea et al. (2014) to study the effectiveness of citation information in improving the keyword extraction task. Since the study by Caragea et al. (2014) uses citation information, the method is inefficacious for generic documents outside academic or scientific literature that do not have citation information. We evaluate the performance of XGB and NB-A models on KDD and WWW datasets, and compare them with two supervised baselines - CeKE (Caragea et al., 2014) and MIKE (Zhang et al., 2017).

Table 7
Performance evaluation for Keyphrase Extraction on KDD and WWW datasets. @*k*: evaluation results for top-*k* keyphrases.

Model	KDD			WWW		
	P	R	F1	P	R	F1
XGB@5	26.9	49.7	33.3	**30.3**	**52.3**	**36.6**
NB-A@5	**27.5**	**50.9**	**34.1**	30.3	52	36.5
MIKE@5 (Zhang et al., 2017)	14.01	17.33	15.49	14.8	15.05	14.92
CeKE* (Caragea et al., 2014)	21.3	41.3	28.0	22.7	38.6	28.4

* Results are averaged at document-level for 10-fold cross validation.

As mentioned above, CeKE enhances the feature set by using citation information along with statistical (tf-idf, position of occurrence, etc.) and linguistic (part-of-speech tags) information. The approach uses Naïve Bayes classifier to build a predictive model to identify keywords. On the other hand, MIKE uses multidimensional information (e.g., topical information) to enhance the feature set. It uses gradient-descent algorithm to build the predictive model.

Table 7 shows that XGB and NB-A outperform the two baselines with large margins in terms of precision, recall and F1-score. Performance of both XGB and NB-A models is comparable for the two datasets, with no (statistically) significant difference in performance of the models. Specifically, NB-A performs best for KDD dataset when we extract top-5 keyphrases, and XGB performs best on WWW dataset for the same number of keyphrases. The number of extracted keyphrases for both these models (i.e., 5 in this case) corresponds to the average number of keyphrases for both these datasets (column K_{avg} in Table 1).

8.3.3. Result on Krapivin2009 dataset

We evaluate the performance of XGB and NB-A models on Krapivin2009 dataset, and compare with one unsupervised baseline. The unsupervised baseline is a recent work by Papagiannopoulou and Tsoumakas (2018), which uses GloVe to encode local word embeddings for the terms in title and abstract of a scientific publication. A mean *reference* vector is computed from the vectors trained from the full-text, and keyphrases are extracted by ranking all terms on the basis of their cosine similarity to the reference vector. Reference vector represents the semantics of the complete document, and words closer to it are considered keyphrases. RVA (Reference Vector Algorithm from abstracts) with 50-dimensional vector representation reports best result in terms of F1-score.

We present our experimental results on Krapivin2009 dataset in Table 8. We observe that RVA performs best for this dataset in terms of F1-score. This shows the effectiveness of word embeddings in determining keyphrases. Blank entries ('-') in the table mean unavailability of results in relevant literature.

However, it is noteworthy that the evaluation of the baseline and our models is not same. The baseline is evaluated for top one-third keyphrases, whereas our models are evaluated for top-5 predicted keyphrases. This makes it difficult for us to perform an unbiased comparison of the methods. Among XGB and NB models, XGB performs best when we extract top-5 keyphrases. The number of keywords extracted (i.e., 5 in this case) correlates with the aver-

Table 8
Performance evaluation for Keyphrase Extraction on Krapivin2009 dataset. @*k*: evaluation results for top-*k* keyphrases.

Model	P	R	F1
XGB@5	28.1	29.8	27.7
NB-A@5	27.2	28.6	26.7
RVA* (Papagiannopoulou & Tsoumakas, 2018)	–	–	**32.06**

* The algorithm is evaluated for top one-third keyphrases.

Table 9
Performance evaluation for Keyphrase Extraction on SemEval2010 dataset. @*k*: evaluation results for top-*k* keyphrases.

Model	P	R	F1
XGB@10	**38.5**	**25.6**	**30.3**
NB-A@10	36	24	28.3
HUMB@10 (Lopez & Romary, 2010b)	32.0	21.8	26.0
Boudin@10 (Boudin, 2018)	–	–	14.5
XGB@15	30	29.9	29.5
NB-A@15	28.6	28.4	28.1
HUMB@15	27.2	27.8	27.5

age number of keyphrases per document for Krapivin2009 dataset ($K_{avg} = 5$ in Table 1).

8.3.4. Result on SemEval2010 dataset

SemEval2010 dataset was curated for Task 5 of the Workshop for Semantic Evaluation, 2010. 21 teams participated in the task, and HUMB (Lopez & Romary, 2010b) performed best for author-and-reader-assigned keywords (Kim et al., 2010).

We compare our XGB and NB-A models with HUMB (Lopez & Romary, 2010b) and Boudin's algorithm (Boudin, 2018) as baselines. HUMB is a supervised method that identifies keyphrases using a predictive model trained on a feature set of document structure (e.g. section and position), content (e.g. tf-idf), and external information (GRISP terminology and Wikipedia). The model is initially trained using a bagged decision tree, and candidates are further re-ranked using a probabilistic model to improve their ranking (Lopez & Romary, 2010b). Boudin's algorithm is unsupervised, which uses a multipartite graph representation of the text to encode keyphrase candidates and topics in a single graph. Candidates are ranked using TextRank computation for weighted graphs.

Table 9 presents the experimental results for the proposed models and the two baselines. We observe that XGB model outperforms all models in terms of precision, recall, and F1-score when we extract top-10 keyphrases. We also show the results of our models for top-15 keyphrases ($K_{avg} = 16$ in Table 1 for SemEval2010 dataset). However, we only show results of one baseline, HUMB, as Boudin's algorithm do not report results for top-15 keyphrases. The difference in performance of our models (i.e., XGB and NB) for top-10 and top-15 keyphrases is insignificant, with a slightly better performance for top-10 keyphrases.

8.3.5. Result on Marujo2012 dataset

Marujo2012 dataset (Marujo et al., 2012) is a cross-domain dataset that we adopted to establish domain-independence of our proposed method. The dataset consists of news articles. To compare the performance of XGB and NB-A models, we consider as baseline Boudin's algorithm (Boudin, 2018), which has already been briefed in Section 8.3.4.

We present our experimental results in Table 10. We observe that our models outperformed the baseline by a huge margin and shows impressive performance for a cross-domain keyphrase extraction model. Specifically, best precision is achieved when we extract top-10 keyphrases using NB-A model, and best recall and F1-score is achieved when we extract top-30 keyphrases using the XGB model. High precision and comparatively low recall is due to the high number of gold-standard keyphrases assigned for this dataset (Table 1, column K_{avg}). Our models predicted lesser number of keyphrases as in the gold-standard list, out of which most are correctly extracted (high precision) but a few correct keyphrases are missed (low recall).

Table 10
Performance evaluation for Keyphrase Extraction on Marujo2010 dataset. @*k*: evaluation results for top-*k* keyphrases. *: Evaluated only for top-5 and top-10 keyphrases.

Model	P	R	F1
XGB@30	83.4	**43.1**	**53.8**
NB-A@30	80.81	33.36	44.64
XGB@10	92.86	25.62	38.33
NB-A@10	**92.91**	25.55	38.21
Boudin@10* (Boudin, 2018)	–	–	18.2

Table 11
Mean (μ) and standard deviation (σ) of distributions shown in Fig. 5.

Datasets	Mean	Standard deviation
Hulth2003	0.547325	0.007697
WWW	0.365530	0.006432
KDD	0.332905	0.008523
SemEval2010	0.303430	0.008322
Krapivin2009	0.276609	0.004156
Marujo2012	0.595079	0.008096

8.3.6. Statistical significance testing

Our next goal is to examine if the performance of our algorithm is (statistically) significantly better than that of the corresponding baselines for each dataset. Since we know only the macro-averaged metrics for the baselines, we can't use traditional statistical significance testing approaches. Therefore, we follow the approach recommended by Berg-Kirkpatrick, Burkett, and Klein (2012) and Dror, Baumer, Shlomov, and Reichart (2018).

Let O be our algorithm and B be the baseline algorithm. We test the null hypothesis, H0: the performance of O is no better than the performance of B, against the alternative, H1: the performance of O is significantly better than B. We compare our method with the corresponding baselines for each dataset. The performance difference, $\delta(x)$, is the difference in performance metric of O minus B for the dataset x.

For each dataset, we generate one million bootstrap samples from the document-level F1-score vectors for our algorithm.[9] Following the algorithm recommended by Berg-Kirkpatrick et al. (2012) (Fig. 4), we estimate the p-value as the ratio of number of times our algorithm beats the baseline by twice the margin[10]($2\delta(x)$) on the bootstrap samples, to the total number of samples. For p-value < 0.05, we reject the null hypothesis.

For all datasets except Krapivin2009, low p-value (< 0.05) led to rejection of H0. This is a strong evidence that the superior performance of the proposed method is not due to chance. As evident in Table 8, performance of our method is weaker than the competing method for Krapivin2009 corpus. The same is confirmed by the statistical test. We show the distribution of F1-scores for one million bootstrap samples for each dataset (for XGB model) in Fig. 5. Each plot is paired with the corresponding quantiles of standard normal distribution. Distribution of F1-scores is found to be good normal fit (Fig. 5(a)–(f)) for all datasets including Krapivin2009. The mean and standard deviation for each of these distributions is shown in Table 11. Low standard deviation values establish consistency of the proposed method.

At this point in time, we are unable to explain consistently low performance of our method on Krapivin2009 dataset for keyphrase

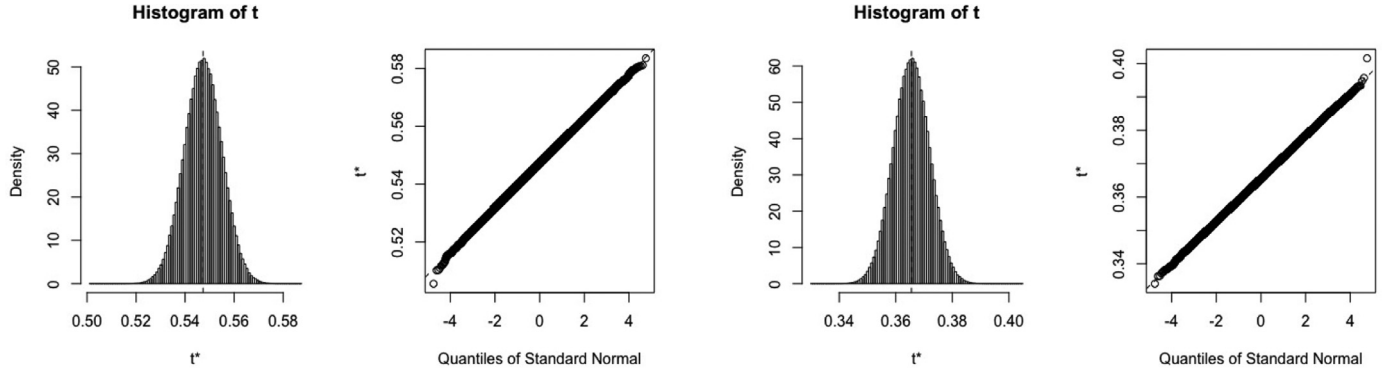
[9] We use `boot` package in R (https://cran.r-project.org/web/packages/boot/boot.pdf) to generate the bootstrap samples.

[10] Please refer to Section 2.2 of Berg-Kirkpatrick et al. (2012) for a detailed discussion.

Input: The original metric vector x of size n, and b (performance metric of baseline)
Output: The estimated *p-value*

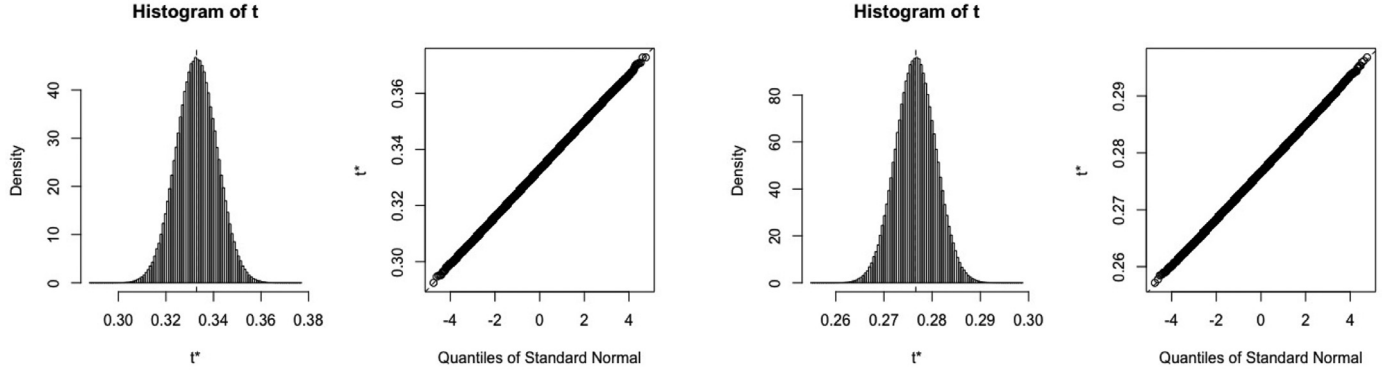
1. Draw R bootstrap samples $x^{(i)}$ of size n from x.
2. Initialize $s = 0$.
3. For each sample $x^{(i)}$ do:
   3.1. Compute $\delta(x^{(i)}) = \text{mean}(x^{(i)}) - b$
   3.2. Increment s if $\delta(x^{(i)}) > 2\delta(x)$
4. Estimate $p\text{-value}(x) \approx s/R$

Fig. 4. Pseudocode for estimating p-value (Berg-Kirkpatrick et al., 2012).



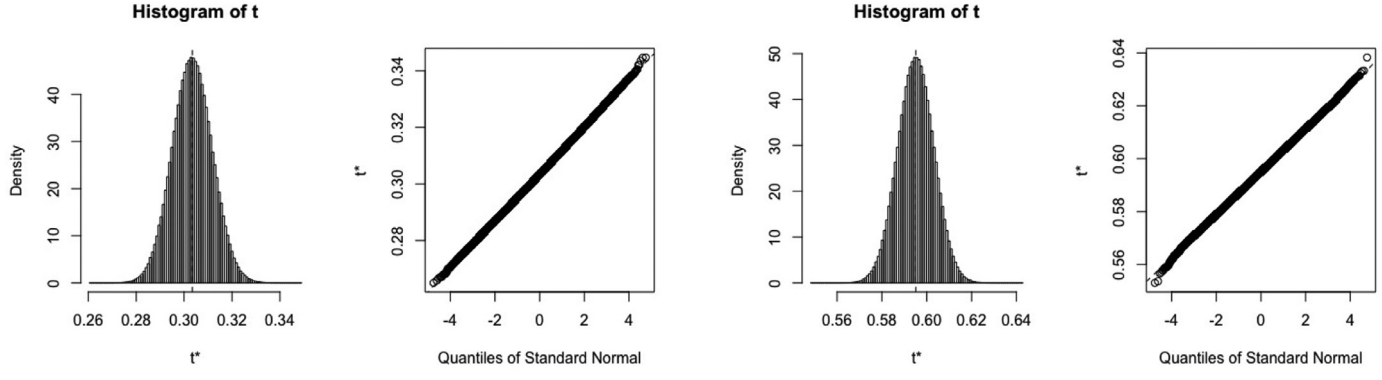
(a) Hulth2003, XGB@10

(b) WWW, XGB@5



(c) KDD, XGB@5

(d) Krapivin2009, XGB@5



(e) SemEval2010, XGB@10

(f) Marujo2012, XGB@30

Fig. 5. Distribution of average F1-scores for $R = 10^6$ bootstrap samples drawn from each dataset, along with corresponding quantiles of standard normal distribution. t: Mean F1-score for bootstrap sample.

Table 12
Keywords predicted from the original English and translated Hindi text using the pre-trained XGB model. '-' in translated keywords mean the corresponding Hindi word should be a stopword.

Item	Hindi (H)	English (E)
Predicted Keywords	**अच्छे**, अपडेट, **अराजक**, आखिरी, आनंद, आपका, आपको, उपयोग, **एलिस**, **कपटी**, कमीकमार, कम, **क्लासिक**, गई, **चरमराती**, चीज, जिसकी, **जेम्स**, **झटके**, **झटकों**, टीआई, **ठंड**, **डाल्टन**, **तेज**, **थ्रिलर**, **दरवाज़े**, **दिग्गज**, देखते, पुल, **पैट्रिक**, **प्रेतवाधित**, **फिल्में**, **बकवास**, बचाता, बच्चों, **बायरन**, बारे, बेहतर, **मूल**, रूप, **रोज**, लेखक, **लेह**, लोगों, शुरू, **शोर**, समझाता, समीक्षा, **संयमित**, **सिम्पकिंस**, **सॉ**, स्कूल, **स्लैम**, हमें, **हॉरर**	bridges, called, **chaotic**, **chills**, **classic**, **cryptically**, **deathly**, **delivers**, **domestic**, effect, **elise**, expect, explains, **finale**, **franchise**, **gravely**, guys, **haunted**, **hauntedhouse**, **horror**, house, **insidious**, josh, **launched**, **lin**, move, **movie**, **movies**, **nononsense**, **psychic**, real, realm, renai, **restrained**, review, **screen**, **shaye**, **shocks**, starting, steals, tale, thing, **unexpected**, **veteran**, **wan**, **whannell**, world
Translation in English	**good**, update, **chaotic**, last, bliss, -, -, use, **Elise**, **insidious**, occasional, shortlived, **classic**, -, **creaking**, thing, -, **James**, **shocks**, **shocks**, ty, **chills**, **dalton**, **fast**, **thriller**, **doors**, **veteran**, see, bridges, **patrick**, **haunted**, **movies**, **nonsense**, save, children, **Byrne**, about, better, **original**, type, **rose**, writter, **leigh**, guys, starting, **noises**, explains, review, **restrained**, **simpkins**, **saw**, school, **slam**, -, **horror**	-
H ∩ E (20)	अराजक (chaotic), एलिस (elise), कपटी (insidious), क्लासिक (classic), चीज (thing), झटके (shocks), झटकों (shocks), ठंड (chills), दिग्गज (veteran), पुल (bridges), प्रेतवाधित (haunted), फिल्में (movies, movie), बकवास (nononsense), लोगों (guys), शुरू (starting), समझाता (explains), समीक्षा (review), संयमित (restrained), हॉरर (horror)	
Match with Gold (G)	29	29

extraction (lower by ≈5%). Deeper investigation about the nature of Krapivin documents is pending for future.

8.4. Keyword extraction from Indian language documents

India is a country with 23 official languages, including English. According to Census of India of 2011, India has 121 major languages with more than 10,000 speakers for each language.[11] With such a wide variety of written and spoken languages, there is a huge collection of literature available in the country. Since digital texts are increasing day by day, automatic analysis of such documents needs to be addressed. However, due to unavailability of sophisticated NLP tools, documents written in Indian regional languages, which are grossly under-resourced, remain poorly analyzed.

We demonstrate the language-agnostic character of the proposed method by using the XGB model trained on English language documents to predict keywords from text documents written in two Indian languages. We establish the effectiveness of the proposed method in two phases. In the first phase, we choose an English document and predict the keywords. We Google translate[12] the same document to Hindi and compare the keywords predicted from the translation with keywords predicted from the English document. We choose to translate an English document to Hindi over an article originally written in Hindi so that the quality of predicted Hindi keywords can be compared with the English gold-standard. In the second phase, we apply the same XGB model on five Assamese language documents. Below, we describe in detail the experiments and the observations.

The sample English text is a randomly chosen document from Marujo2012 dataset (id "art_and_culture-20925876.txt"), which is translated to Hindi. We combine[13] two publicly available Hindi stopwords lists[14] to create an expanded stop-list. Table 12 presents a detailed analysis of the results. The columns correspond to results for Hindi and English text, respectively. First row of the table lists predicted keywords using the XGB model. Based on the English gold-standard keywords list, we highlight each recalled keyword in bold. For Hindi keywords, we highlight the words whose English translation is present in the gold-standard list. Next row presents English translation for every Hindi keyword predicted by the model. The '-' in the translation denotes that the corresponding word is semantically a Hindi stopword but is not included in the stop-list. Third row lists twenty keywords that are predicted from both Hindi and English versions, with the translations given in parenthesis. Out of twenty-nine total predicted keywords (last row), twenty common keywords indicate fairly good performance of the model on the Hindi document although it was trained on English corpora. We are confident that human translated Hindi text of the English document will yield improved performance. We clarify here that the same number of Hindi and English keywords matching with the gold-standard is incidental.

In the second phase, we experiment to evaluate the model for Assamese[15] language texts. To perform this experiment, we collected five Assamese articles from Assamese Wikipedia.[16] The topics of the documents and the keywords predicted from each of the documents are shown in Table 13. We are unable to objectively assess the performance of our method due to unavailability of gold-standard keywords for these documents. We provide English translation for the corresponding predicted keywords to enable rational assessment of the performance of our method. Keywords relevant to the topic are marked in bold. Last column shows the ratio of

[11] http://censusindia.gov.in/2011Census/C-16_25062018_NEW.pdf.

[12] https://translate.google.com/. The text, stopword list, and associated codes are available at https://github.com/SDuari/Supervised-Keyword-Extraction.

[13] We are aware that the stopword lists are not perfect, and they missed a few stopwords. However, we do not improve on the stopword list as it is out of scope for this study.

[14] https://github.com/stopwords-iso/stopwords-hi and https://www.ranks.nl/stopwords/hindi.

[15] Assamese is the regional language of Assam, the most populous North-Eastern state of India. It is spoken by more than 15 million people, and is the mother tongue of the first author of this paper.

[16] https://as.wikipedia.org/. The collection along with the stopword list is available at https://github.com/SDuari/Supervised-Keyword-Extraction.

Table 13
Keywords predicted from the Assamese documents using the pre-trained XGB model. '-' in translated keywords mean the corresponding Assamese word is semantically a stopword. R/E: Number of relevant keywords/number of predicted keywords excluding '-'.

Topic	Assamese Keywords	English Translations	R/E
Animation	ইংৰাজী, এটা, এনিমেছন, এনিমেছনৰ, এনিমেশ্যন, ছবি, হৈছিল, দিয়া, নির্মাণ, হয়, কৰিলে, কৰোঁতে, ক্ৰম, গতিৰ, চলমান, চিত্ৰ, চিত্ৰৰ, দুটা, দ্বিবিমীয়, পাছত, প্ৰক্ৰিয়াৰে, ভ্ৰম, যাক, লক্ষ্য, সৃষ্টি, স্থিৰ, হয়, সহায়েৰে	english, -, **animation**, (of) **animation**, **animation** (different spelling), **image**, -, -, creation, -, -, -, **sequence**, (of) speed, **moving**, **picture**, (of) **picture**, -, **two dimensional**, -, technique, **illusion**, -, built, target, **still** (as in still image), -, (with) help	11/18
Capitalism	অৰ্থনৈতিক, আহৰণ, ইয়াৰ, ইংৰাজী, উপাদানসমূহ, এক, কল্যাণকামী, ঠাই, পৰে, পাৰে, পুঁজিবাদ, পুঁজিবাদৰ, পুঁজিবাদী, প্ৰকাৰৰ, প্ৰতিযোগিতাৰ, প্ৰতিষ্ঠানমূলক, ফেয়াৰ, বিভিন্ন, ব্যক্তিগত, ব্যৱসায়িক, ব্যৱস্থা, ভিন্ন, মাত্ৰা, মালিকীস্বত্ব, মুক্ত, য'ত, ৰাজনৈতিক, লেইছেজ, সম্পত্তি, হ'ব	**economic**, extract, -, english, elements, -, **welfare**, place, -, -, **capitalism**, **capitalist**, categories, (of) **competition**, **institutional**, **Faire,** several, **personal**, **commercial**, system, vary, degree, **ownership**, **open**, -, **poltical**, **Laissez**, **wealth**, -	14/23
Computer	অংশই, আধুনিক, এক, এটা, কম্পিউটাৰ, কাম, কার্যপ্রণালী, হৈছিল, দি, নামৰ, পাৰে, প্ৰচেছিং, নিয়ন্ত্ৰণ, যাক, সঁজুলি, সঁজুলিৰ, সমাধা, স্বয়ংক্ৰিয়, গাণিতিক	part, **modern**, -, -, **computer**, work, **operations**, -, -, (of) names, -, **processing**, **control**, -, **equipment**, (of) **equipment**, **solve**, **automatic**, **mathematical**	10/13
Movie	ইংৰাজী, একেসময়তে, এখন, এটা, কথাছবি, ক্ৰম, ক্ৰমিক, গতিৰ, চমু, অকল, চলচ্চিত্ৰ, চিত্ৰৰ, চিনেমা, ছবি, তুলি, দৰ্শনি, দেখা, ফিল্ম, বিশেষ, বোলছবি, যাক, নির্মান, স্থিৰ, হয়	english, simultaneously, -, -, **motion picture**, **sequence**, **sequencial**, (of) **speed**, short, -, **film**, (of) **images**, **cinema**, **picture**, capture, **visualize**, see, **film**, **special**, **motion picture**, -, create, **still** (as in still image), -	13/19
Solar System	অৱস্থান, অসংখ্য, আছে, আজিৰ, আণৱিক, ইউৰেনাছ, ইয়াক, উপগ্ৰহসমূহ, একেলগে, কেন্দ্ৰ, ক্ৰমে, খুন্দাৰ, গঠিত, গ্ৰহ, গ্ৰহবোৰৰ, গ্ৰহাণু, চাৰিওফালে, হৈছিল, টা, ডাৱৰৰ, নেপচুন, পদার্থক, পৃথিৱী, প্ৰদক্ষিণ, প্ৰায়, ফলত, বছৰৰ, বলত, বাক্ক, বিলিয়ন, বুধ, বৃহস্পতি, মঙ্গল, মহাকর্ষণ, মহাজাগতিক, যাক, লগত, লগতে, লগেতে, শক্তিৰ, শনি, শুক্ৰ, সকলো, সিহঁতৰ, সূর্য, সূর্যাৰ, সৃষ্টি, সৌৰজগত, হয়	locate, several, -, -, **molecular, Uranus**, -, **satellites**, together, **center**, sequentially, (of) **gravitational collapse**, composed of, **planet**, (of) **planets**, **dwarf planet**, around, -, -, (of) **cloud**, **Neptune**, objects, **Earth**, **orbits**, nearly, -, years, force, bound, billion, **Mercury**, **Jupiter**, **Mars**, **gravitational**, **intersteller**, -, -,-,-, (of) force, **Saturn**, **Venus**, -, -, **Sun**, (of) **Sun**, **formation**, **solar system**, -	23/36

the relevant keywords to the total number of predicted keywords (barring "-").

Some noise is evident in the predicted keywords (e.g. 'help' in *Animation*, 'several' in *Capitalism* and *Solar system*, 'part in *Computer*, 'see' in *Movie*). Interestingly, the term 'english' occurs in 3/5 topics. This is because English translations of some words are preceded by the term 'english' in the Assamese text. Morphologically inflected words with different endings (translated with semantics/context in parenthesis) manifest as repetitions. For example, in *Animation*, words 'image', 'picture', '(of) picture', indicate to an Assamese reader that *image* and *picture* are keywords.

This substantiates our claim that the proposed method is applicable to any language outside the training corpus, and can perform reasonably well without using any linguistic tools. However, morphological idiosyncrasies of languages in general may have somewhat blunting effect on the potential of the proposed method. Introducing a human in the loop can quickly resolve such issues to aid automatic indexing of documents in language specific digital libraries and repositories.

9. Conclusion

We presented a supervised framework for automatic keyword extraction using graph-theoretic properties of words in text. The framework is domain-, collection, and language-independent. We explored six graph node properties to distinguish keywords from non-keywords - degree centrality (strength of a node), eigenvector centrality, PageRank, PositionRank, coreness, and clustering coefficient. Using training set from a mixed collection of short and long scientific texts, we trained classification models on SMOTE-balanced training set using XGBoost, Naïve Bayes, and bagging and boosting ensembles of Naïve Bayes. The induced models are then tested on four unseen collections, out of which one is from a different domain. Experimental results show that XGBoost (XGB) outperforms others in terms of F1-score, while Adabbost ensemble of Naïve Bayes (NB-A) closely follows. We also empirically affirm that our approach is domain- and collection-independent. Furthermore, to validate the claim of language-independence, we evaluated our models on unseen Indian language texts (Hindi and Assamese). Experimental results for keyphrase extraction show that the proposed models (XGB and NB-A) are able to outperform established keyphrase extraction models for all datasets except Krapivin2009.

Top-5 keyphrases extracted from this paper[17] using XGB model are - "supervised keyword extraction", "complex network", "extract node properties", "graph-based node properties", and "keyword extraction techniques", which basically sums up the work presented here.

In future, we plan to apply the proposed approach over documents written in various Indian languages. We also intend to make our model a benchmark for cross-lingual studies, on the basis of which future keyword extraction algorithms for Indian languages could be evaluated.

Declaration of Competing Interest

The authors certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies,

[17] Excluding title, keywords, Conclusion, References, footnotes, and tables and figures along with their captions.

stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

Credit authorship contribution statement

Swagata Duari: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft, Visualization. **Vasudha Bhatnagar:** Conceptualization, Methodology, Validation, Formal analysis, Investigation, Writing - review & editing, Supervision.

References

Barrat, A., Barthelemy, M., Pastor-Satorras, R., & Vespignani, A. (2004). The architecture of complex weighted networks. *Proceedings of the National Academy of Sciences, 101*(11), 3747–3752.

Berg-Kirkpatrick, T., Burkett, D., & Klein, D. (2012). An empirical investigation of statistical significance in nlp. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 995–1005). Association for Computational Linguistics.

Blanco, R., & Lioma, C. (2012). Graph-based Term Weighting for Information Retrieval. *Information Retrieval, 15*(1), 54–92.

Boudin, F. (2013). A comparison of centrality measures for graph-based keyphrase extraction. In *Proceedings of the sixth international joint conference on natural language processing* (pp. 834–838).

Boudin, F. (2018). Unsupervised keyphrase extraction with multipartite graphs. In *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 2 (short papers): 2* (pp. 667–672).

Brin, S., & Page, L. (1998). The Anatomy of a Large-scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems, 30*(1–7), 107–117.

Bulgarov, F., & Caragea, C. (2015). A comparison of supervised keyphrase extraction models. In *Proceedings of the 24th international conference on world wide web* (pp. 13–14). ACM.

Caragea, C., Bulgarov, F. A., Godea, A., & Gollapalli, S. D. (2014). Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1435–1446).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321–357.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794). ACM.

Chuang, J., Manning, C. D., & Heer, J. (2012). Without the clutter of unimportant words: Descriptive keyphrases for text visualization. *ACM Transactions on Computer-Human Interaction (TOCHI), 19*(3), 19.

Dror, R., Baumer, G., Shlomov, S., & Reichart, R. (2018). The hitchhikers guide to testing statistical significance in natural language processing. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1383–1392).

Duari, S., & Bhatnagar, V. (2019). sCAKE: Semantic Connectivity Aware Keyword Extraction. *Information Sciences, 477*, 100–117.

Florescu, C., & Caragea, C. (2017). A position-biased pagerank algorithm for keyphrase extraction. In *Proceedings of thirty-first aaai conference on artificial intelligence* (pp. 4923–4924).

Gollapalli, S. D., Li, X.-L., & Yang, P. (2017). Incorporating expert knowledge into keyphrase extraction. In *Thirty-first aaai conference on artificial intelligence* (pp. 3180–3187).

Herrera, J. P., & Pury, P. A. (2008). Statistical Keyword Detection in Literary Corpora. *The European Physical Journal B, 63*(1), 135–146.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on empirical methods in natural language processing* (pp. 216–223). Association for Computational Linguistics.

Kim, S. N., Medelyan, O., Kan, M.-Y., & Baldwin, T. (2010). Semeval-2010 Task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 21–26). Association for Computational Linguistic.

Krapivin, M., Autaeu, A., & Marchese, M. (2009). Large Dataset for Keyphrases Extraction. *Technical Report DISI-09-055*.

Litvak, M., Last, M., Aizenman, H., Gobits, I., & Kandel, A. (2011). DegExtA language-independent graph-based keyphrase extractor. In *Advances in intelligent web mastering–3* (pp. 121–130). Springer.

Lopez, P., & Romary, L. (2010a). GRISP: A massive multilingual terminological database for scientific and technical domains. In *Proceedings of the seventh conference on international language resources and evaluation (lrec'10)*.

Lopez, P., & Romary, L. (2010b). HUMB: Automatic key term extraction from scientific articles in GROBID. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 248–251). Association for Computational Linguistics.

Luhn, H. P. (1957). A Statistical Approach to Mechanized Encoding and Searching of Literary Information. *IBM Journal of R&D, 1*(4), 309–317.

Marujo, L., Gershman, A., Carbonell, J., Frederking, R., & Neto, J. P. (2012). Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. In *Proceedings of the eighth international conference on language resources and evaluation (lrec-2012)*.

Medelyan, O., Frank, E., & Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 3: 3* (pp. 1318–1327). Association for Computational Linguistics.

Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

Mothe, J., Ramiandrisoa, F., & Rasolomanana, M. (2018). Automatic keyphrase extraction using graph-based methods. In *Proceedings of the 33rd annual acm symposium on applied computing* (pp. 728–730). ACM.

Nguyen, T. D., & Kan, M.-Y. (2007). Keyphrase extraction in scientific publications. In *International conference on asian digital libraries* (pp. 317–326). Springer.

Ortuno, M., Carpena, P., Bernaola-Galván, P., Muñoz, E., & Somoza, A. (2002). Keyword Detection in Natural Languages and DNA. *EPL (Europhysics Letters), 57*(5), 759.

Papagiannopoulou, E., & Tsoumakas, G. (2018). Local word vectors guiding keyphrase extraction. *Information Processing & Management, 54*(6), 888–902.

Rousseau, F., & Vazirgiannis, M. (2015). Main core retention on graph-of-words for single-document keyword extraction. In *European conference on information retrieval* (pp. 382–393). Springer.

Seidman, S. B. (1983). Network Structure and Minimum Degree. *Social Networks, 5*(3), 269–287.

Sterckx, L., Demeester, T., Develder, C., & Caragea, C. (2016). Supervised keyphrase extraction as positive unlabeled learning. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 1–6).

Tixier, A., Malliaros, F., & Vazirgiannis, M. (2016). A graph degeneracy-based approach to keyword extraction. In *Proceedings of the 2016 conference on empirical methods in natural language processing6* (pp. 1860–1870).

Turney, P. D. (1999). Learning to Extract Keyphrases from Text. *Technical Report, National Research Council of Canada*.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (1999). KEA: Practical automatic keyphrase extraction. In *Proceedings of the fourth acm conference on digital libraries* (pp. 254–255). ACM.

Zaki, M. J., Meira Jr, W., & Meira, W. (2014). *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press.

Zhang, C. (2008). Automatic Keyword Extraction from Documents using Conditional Random Fields. *Journal of Computational Information Systems, 4*(3), 1169–1180.

Zhang, Y., Chang, Y., Liu, X., Gollapalli, S. D., Li, X., & Xiao, C. (2017). MIKE: Keyphrase extraction by integrating multidimensional information. In *Proceedings of the 2017 acm on conference on information and knowledge management* (pp. 1349–1358). ACM.