Domain-Specific Keyword Extraction Using Joint Modeling of Local and Global Contextual Semantics

MUHAMMAD ABULAISH and MOHD FAZIL, South Asian University
MOHAMMED J. ZAKI, Rensselaer Polytechnic Institute

Domain-specific keyword extraction is a vital task in the field of text mining. There are various research tasks, such as spam e-mail classification, abusive language detection, sentiment analysis, and emotion mining, where a set of domain-specific keywords (aka lexicon) is highly effective. Existing works for keyword extraction list all keywords rather than *domain-specific* keywords from a document corpus. Moreover, most of the existing approaches perform well on formal document corpuses but fail on noisy and informal user-generated content in online social media. In this article, we present a hybrid approach by jointly modeling the local and global contextual semantics of words, utilizing the strength of distributional word representation and contrasting-domain corpus for domain-specific keyword extraction. Starting with a seed set of a few domain-specific keywords, we model the text corpus as a weighted word-graph. In this graph, the initial weight of a node (word) represents its semantic association with the target domain calculated as a linear combination of three semantic association metrics, and the weight of an edge connecting a pair of nodes represents the co-occurrence count of the respective words. Thereafter, a modified PageRank method is applied to the word-graph to identify the most relevant words for expanding the initial set of domain-specific keywords. We evaluate our method over both formal and informal text corpuses (comprising six datasets), and show that it performs significantly better in comparison to state-of-the-art methods. Furthermore, we generalize our approach to handle the language-agnostic case, and show that it outperforms existing language-agnostic approaches.

CCS Concepts: • **Keyword Extraction → Lexicon Generation**; *Online social network analysis*; Contextual semantics;

Additional Key Words and Phrases: Text mining, information extraction, domain-specific keyword extraction, language-agnostic keyword extraction

ACM Reference format:
Muhammad Abulaish, Mohd Fazil, and Mohammed J. Zaki. 2022. Domain-Specific Keyword Extraction Using Joint Modeling of Local and Global Contextual Semantics. *ACM Trans. Knowl. Discov. Data.* 16, 4, Article 70 (January 2022), 30 pages.
https://doi.org/10.1145/3494560

Authors' addresses: M. Abulaish and M. Fazil, South Asian University, Chanakyapuri, New Delhi, India; emails: abulaish@sau.ac.in, mohdfazil.jmi@gmail.com; M. J. Zaki, FIEEE, Rensselaer Polytechnic Institute, Troy, NY; email: zaki@cs.rpi.edu.

1556-4681/2022/01-ART70 $15.00
https://doi.org/10.1145/3494560

1 INTRODUCTION

In the real-world, people can be grouped into different communities based on their profession, interests, geographic region, ethnicity, and so on. Each group typically has its preferred or domain-specific lexicon, including jargon and slang, that is normally used in its routine discourse. For example, computer scientists employ many computing technology-related terms, such as *algorithm*, *artificial intelligence*, *machine learning*, *cloud computing*, *GPUs*, and so on. Besides representing routine discourse, domain-specific keywords can also be used to model the personality, attitude, or behavior of a person based on his/her language content. Automatic keyword extraction from user-generated content plays a vital role in the deep understanding of a domain of discourse and personality traits. With the popularity of online social media, a huge amount of data is generated every moment, which is being exploited for information and knowledge acquisition in the form of topic modeling [9], text summarization [2], event detection [1], and ontology creation [12]. Besides these, keyword (keyphrase) extraction is another fundamental task for data and knowledge acquisition from textual data [62].

1.1 Keyword Extraction Methods and Their Limitations

Keyword extraction is an important task where relevant words from a text corpus are extracted describing the subjects within the (possibly domain-specific) corpus. There are several approaches for keyword extraction for text classification and summarization, opinion mining and sentiment analysis, and various other text information processing tasks [2, 31, 43, 51]. However, it is still an open problem, and researchers continue to devise efficient methods for effective keyword extraction [4]. Most of the existing approaches for keyword extraction are graph-based, modeling an initial set of candidate words as a graph [31, 43]. Furthermore, the existing literature presents approaches for keyword extraction from formal document corpuses spanning research and news articles. However, with the growth of social media and blogging sites, researchers are now evaluating methods to extract keywords from **online social networks** (**OSNs**) [6, 8]. Keyword extraction from OSN data is complicated and challenging due to the presence of noise, inconsistency, slang, and casual writing style. Most approaches for keywords extraction suffer from two basic limitations [68]. The first limitation is that they are generally domain and dataset dependent, and perform poorly on informal and noisy texts from OSNs. The second limitation is that most existing approaches overlook the contextual semantics, and generally rely on simple statistical measures, such as frequency count, co-occurrence count, and measures like **term-frequency inverse document frequency** (**tf-idf**). As a result, keyword extraction approaches generally extract all the keywords from a document corpus irrespective of the domain. For example, if a corpus consists of tweets related to topics like hate, finance, and politics, and we have some hate-related seed words and want to extract more hate-related keywords from the corpus then we need an approach that can extract only hate-centric keywords.

1.2 Why Domain-Specific? Background and Motivation

Whereas existing state-of-the-art keyword extraction approaches like [8, 31, 43] identify statistically important words from a dataset, they ignore the domain semantics. As a result, a keyword set created using these approaches contains all the important words independent of the conceptualizing domain. On the other hand, a domain-specific keyword set is different from a simple keyword set because it contains only the words most related to a particular domain of interest. For example, when we use an initial lexicon of three hate-related seed words (*nigga*, *fuck*, and *bitch*), our domain-specific approach extracts a number of other hate-related words like *idiot*, *hell*, *ass*, *disgusting*, and so on that are associated closely with hate. However, if we use a domain-independent

approach, the resulting keywords are diverse and do not particularly relate to hate, as we detail in Section 4.4.4.

A domain-specific keyword set is also called a lexicon for that domain. Henceforth, we use *domain-specific keyword set* and *lexicon* interchangeably. Researchers have extensively used lexicon-based approaches in various text information processing tasks, such as text classification and summarization, hate language monitoring, and e-mail spam classification [3, 14, 56]. As a result, there are well-established benchmark lexicons related to different domains. For example, Hatebase[1] is a multilingual lexicon containing hate words related to religion, ethnicity, race, gender, and sexual discrimination, which is used in classification systems for offensive language detection, and gender prediction. Likewise, SocialSent[2] is a lexicon containing words related to different categories of sentiments, which is used in sentiment analysis systems. However, most of the existing lexicons are manually curated through crowd-annotation [46, 57], which has three main limitations: (i) it is time-consuming and expensive, (ii) accuracy of such a corpus depends on domain knowledge and suffers from human biases, and (iii) it is not incremental because every lexicon expansion requires another round of manual annotation. Although there are some standard domain-specific lexicons, they are not sufficient to cover every domain of interest. For example, if one needs the keyword set of radical/extremist words used in the South Asian region for monitoring or classification of extremist content, then there is no such lexicon. Nevertheless, we may initially start with a few radical keywords, such as *kashmirfreedom*, *gazwaehind*, and *khalistan*, and then use our domain-specific keyword extraction technique to discover other relevant keywords relating to extremism in South Asia, as we demonstrate in Section 4.

Domain-specific keyword extraction approaches mostly use the concept of contrasting corpora and graph-based modeling [37, 49, 55]. Contrasting corpora-based approaches like [37, 49] are based on global statistics like frequency counts and inverse document frequency counts, which ignore the local contextual semantics-based similarity between the words. On the other hand, graph-based keyword extraction approaches initially construct a graph using a set of candidate keywords, wherein nodes and edges are assigned weights using some statistical metrics. The node and edge weights show the relative importance and inter-node relationship, respectively. In the existing literature, authors have introduced various metrics based on statistical and structural information for node and edge weight assignment, as shown in Table 1. The existing metrics for node weight assignment do not consider the contextual semantics of the underlying word within a domain. To this end, we introduce three metrics to find the contextual inclination of nodes with an initial lexicon of seed words, representing the conceptualizing domain. Although the work in [55] utilizes an initial set of seed words to bias the modeling of candidate words for extracting domain-specific lexical units, it does not incorporate the contextual semantics of the words or the strength of the graph-based modeling used in our approach. To the best of our knowledge, no work utilizes the strength of both global contextual semantics (based on contrasting domain corpora) and local contextual semantics (obtained through word representations and conditional probabilities) for lexicon generation, as done in our proposed approach.

Our Contributions

In this article, we present a hybrid approach called **DSKE** (for **D**omain-**S**pecific **K**eyword **E**xtraction) that extracts contextually similar terms from a text corpus, starting from an initial set of domain-specific seed words. The main contributions of our work are as follows:

[1] https://hatebase.org/.
[2] https://nlp.stanford.edu/projects/socialsent/.

- We jointly model the local (obtained through the distributional representation of words and co-occurrence-based contextual probability) and global contexts (based on word probability in contrasting-domain corpora) for domain-specific keyword extraction.
- We utilize an initial set of seed words to bias the modeling of candidate words for the extraction of domain-specific lexical units.
- Unlike existing approaches, which are evaluated on scientific documents rather than online social media texts, DSKE is evaluated over both formal and informal documents and outperforms existing methods on both types of datasets.
- We extend DSKE to the language-agnostic scenario. This extension, called LA-DSKE, outperforms state-of-the-art language-agnostic approaches.
- We provide a concrete case study where we generate a lexicon of radical terms used by the sympathizers of the *khalistan* and *kashmir* movements to incite users via an online social discourse on Twitter. The generation of such a lexicon is important because most of the extremist and insurgent activities in South Asia, especially in India, are related to these two issues.[3] A recent study reports that fake and inciting social media content is the major reason for the violent mobilization and radicalization of youth in the conflict relating to the Indian state of Jammu and Kashmir [60]. Beyond this specific case study, our approach can provide valuable insights into other movements relating to political and social unrest, as well as social justice.

For reproducibility, we have released both the code for our implementation as well as the datasets at https://github.com/iammohdfazil/DSKE.

2 RELATED WORK

Domain-specific keyword extraction is closely related to keyword extraction. In keyword/keyphrase extraction, relevant and important keywords are extracted from a text-corpus without domain coherence, whereas domain-specific keywords extraction identifies relevant words from a corpus that conceptualizes a specific domain of interest. Therefore, before presenting a review of the literature on domain-specific keyword extraction, we present a brief review of the literature on keyword extraction.

2.1 Keyword Extraction

Keyword extraction is a well-explored and fundamental problem in the field of information retrieval and knowledge discovery, and existing solutions follow four main approaches. The first category includes statistical methods for keyword extraction using different variants of term frequency [35] and word distribution [10, 29, 32, 69]. For example, Matsuo and Ishizuka [41] utilize the frequent terms from a document to compute the word co-occurrence probabilities. Ercan and Cicekli [21] present a lexical chain-based approach for keyword extraction, wherein they convert documents into lexical chains to extract features. Wan et al. [62] present a single document-based approach for keyphrase extraction utilizing the local information from the document and global information from a set of similar documents, called neighboring documents. The researcher have also presented a number of word association measures and utilized statistical significance analysis such as *t-test* [16], z-test [19], and mutual information [17], to extract relevant keywords from a text corpus.

The second category exploits machine learning methods for keyword extraction from large text [25, 28, 58]. For example, [58] uses *statistics*-, *structure*-, and *semantics*-based features to train a

[3] https://www.orfonline.org/expert-speak/42391-cyber-jihad-biggest-challenge-kashmir/.

gradient-boosted decision tree classifier to classify the keyphrases. In another approach, [28] uses *word-*, *orthographic-*, *stopword-*, *parse-tree*, and *title*-based features. Furthermore, they augment the features with three sets of expert features and train a conditional random field-based keyphrase tagger to label keyphrases. To avoid the intensive task of feature engineering, Florescu and Jin [25] model the document as a graph and exploit its structure to represent the documents as a numeric vector. They further use the document vectors to train classifiers to segregate the keyphrases from non-keyphrases.

The third category relies on deep learning for keyword and keyphrase extraction [5, 42, 67]. Meng et al. [42] present an RNN encoder-decoder framework for keyphrase prediction. In another approach, Basaldella et al. [5] describe an RNN architecture based on bidirectional LSTM for keyphrase extraction. Zhang and Xiao [67] presented a seq2seq model utilizing attention, copy, and coverage mechanisms to retrieve keyphrases from a text corpus, which is also capable of generating keyphrases that are even not present in the corpus.

The fourth and most popular category for keyword extraction is graph-based [8, 23, 43]. Graph-based keyword extraction approaches generally perform three steps: (i) identification of candidate words, (ii) construction of a graph where nodes represent the words and edges represent the relationship among words, and finally (iii) ranking of nodes based on some graph-theoretic measure. Furthermore, they can be grouped into two categories—supervised and unsupervised [30]. In their seminal work, Mihalcea and Tarau [43] proposed TextRank for keywords extraction by modeling a text document as a word co-occurrence graph, and then applying the PageRank algorithm [47] to rank and select the relevant keywords. Following this work, researchers have devoted efforts in developing more efficient graph-based methods for keyword extraction. Wan and Xiao [62] extended TextRank to ExpandRank—they first find a set of documents (neighbor documents) similar to the target document using cosine similarity and model the final set of documents as a word co-occurrence graph. Finally, PageRank is applied to the graph to assign a relevance score to words for keyword extraction. Litvak et al. [39] proposed DegExt, a degree distribution-based method for keyword extraction. Florescu and Caragea [23] presented `PositionRank`, a word position-based approach to assign weights to the candidate words, which are further modeled as a word co-occurrence graph and `PageRank` is applied to select the relevant keywords. Biswas et al. [8] modeled the corpus words as a graph and defined various graph-based metrics to assign weights to words before applying `PageRank`. Most existing methods extract keywords from a single document using local structural information ignoring the global information of the corpus.

Recent graph-based methods use the semantic similarity between words based on distributional word representations or word embeddings in graph-based approaches. For example, [63] models a corpus as a word-word graph and presents an embedding-based metric, attraction score, to assign weights to edges. Furthermore, they apply PageRank to the graph to rank the words for keywords selection. Papagiannopoulou and Tsoumakas [48] presented the reference vector algorithm to extract keywords from scientific repositories. The algorithm first calculates a reference vector for each document based on the average word vectors of lexical units of title and abstract. The weight of a candidate word is the similarity between that word and the reference vector of the underlying document. Finally, PageRank is applied to rank the words for keywords extraction. Mahata et al. [40] use phrase embeddings to extract keyphrases from scientific documents. Zhang et al. [68] use embedding-based semantic similarity, but rather than presenting an approach they evaluate its efficacy on already existing term extraction methods. Zhang et al. [65] model a text corpus as a heterogeneous text graph embedding model incorporating word-word, word-topic, and topic-topic graphs. Furthermore, they learn the embeddings from the heterogeneous graphs and use it to rank the candidate words. Although these methods use word embeddings for keyword/keyphrase extraction, they are not designed for domain-specific keyword extraction. Furthermore, all the

Table 1. Node and Edge Weight Measures Used in Existing Graph-Based Approaches

Approach	Node weight	Edge weight	OSN dataset
TextRank [43]	1	Edge is created if the words co-occur within a fixed window (2 to 10). The edge weight is equal to the co-occurrence count within the specified window.	No
ExpandRank [62]	1	Edge is created if the words co-occur within a fixed window (2 to 20). The edge weight is calculated as follows: $E(w_i, w_j) = \sum_{d_p \in D} sim(d_0, d_p) \times count_{d_p}(w_i, w_j)$	No
DegExt [39]	No weight	Edge is created if two words are adjacent. The edges are labelled with the IDs of sentences containing the words.	No
Corpus-Independent [63]	1	Edge is created between two words if they co-occur within a sentence. The edge weight is calculated as follows: $E(w_i, w_j) = dice(w_i, w_j) \times \left(\frac{f(w_i)+f(w_j)}{d}\right)$	No
Sanra and Bhatia [55]	No weight	Edge between every pair of words is created based on conditional probability and pruned based on chi-square test.	Yes
PositionRank [23]	Inverse of the sum of all positions of a word occurrence in the document	Edge is created based on word co-occurrence within a fixed window (2 to 10) and weight equals the co-occurrence count within the underlying window.	No
CNW [8]	Node weight is sum of distance from central word, selectivity centrality, importance of neighboring words, word position, and word frequency.	Edge between two words is created if they are adjacent. The edge weight is calculated as follows: $E(w_i, w_j) = \frac{f(w_i, w_j)}{f(w_i)+f(w_j)-f(w_i, w_j)}$	Yes
Key2Vec [40]	Phrase weight is equal to its cosine distance with the theme vector of that document.	An edge between two phrases is created if they co-occur within a window of 5 words. The edge weight is calculated as follows: $E(p_i, p_j) = \left(\frac{1}{1-cos(p_i, p_j)}\right) \times PMI(p_i, p_j)$	No
NamedKeys [27]	Word/phrase weight is the average of its similarity to the document vector and phrase quality.	Two words/phrases are connected if they co-occur in a sentence. The edges weight is assigned 1.	No

discussed approaches have been evaluated only over scientific documents rather than social media content.

In Table 1, we summarize the strategies used by various state-of-the-art graph-based approaches for node and edge weight assignments. We also note whether an approach has been evaluated on OSN datasets or not. We can observe that only two approaches have been evaluated over OSN datasets—Sarna and Bhatia [55] neither used embeddings to bias the initial weight of nodes nor incorporated any structural information in the keyword extraction process, and while CNW [8] biases the initial word score using a set of structural measures; it neither uses a lexicon of seed words to bias the initial weight nor uses local (through word embedding and co-occurrence) or global (contrasting corpora) contextual semantics. Thus, none of the existing approaches have been evaluated over both formal and informal datasets. Furthermore, except Sarna and Bhatia [55], no approach uses seed words to assign the initial node/word weight.

2.2 Domain-Specific Keywords Extraction

Existing approaches for domain-specific automatic keyword extraction can be categorized as (i) crowd-sourcing based approaches, (ii) contrasting-corpora based approaches, and (iii) graph-based approaches.

2.2.1 Crowd-Sourcing Based Approaches. The several existing methods for domain-specific lexicon generation are based on the manual curation of a set of words through crowd annotation. In an early approach, Strapparava and Valitutti [59] constructed the `WordNet-Affect` lexicon which consists of various categories of emotion representing terms. They developed the `WordNet-Affect` in two layers, wherein the first layer generates the core part of the lexicon through manual curation of emotion-bearing terms from a dictionary. The core lexicon is further extended in the second layer using various lexical and semantic relationships with the `WordNet` words [45]. Similarly, Esuli and Sebastiani [22] presented a semi-supervised approach for sentiment-related keyword extraction. It assigns three polarity scores—*positive*, *negative*, and *objective* to each word. Mohammad and Turney [46] used `Amazon Mechanical Turk`[4] to create a lexicon of 10, 170 emotion-bearing words along with their polarity. Additionally, each word is associated with eight different categories of emotions. Similarly, Staiano and Guerini [57] crawled a reader annotated dataset of news corpus from a news portal[5] and applied term-frequency and tf-idf to find the emotion score of a word.

2.2.2 Contrasting-Corpora Based Approaches. Determining which keyword is relevant to a particular domain of interest is not a trivial task. Apart from the domain of interest, information from other domains also plays an important role. For example, several approaches calculate a term's relevance based on its distribution in the corpora of other domains, and present variants of tf-idf to observe the relevance of a term in a corpus of a particular domain [18, 34, 52] relative to the contrasting domain. In an early approach, Chung [15] exploited the idea of contrasting corpora to retrieve the keywords relevant to a particular domain. Park et al. [49] presented the *domain specificity* index to observe the relevance of a term within a domain-specific corpus relative to the contrasting corpus. They define the *domain specificity* as the ratio of occurrence probability of a term t in a domain-specific corpus c_d to the occurrence probability of t in a generic corpus c_g. The basic idea behind the index is that a term specific to a domain will occur more frequently in the domain-specific corpus rather than in a generic corpus. Kit and Liu [37] proposed the *termhood* index based on the ranking of terms in a domain-specific corpus against their ranking in contrasting (general) corpus. Based on the idea of tf-idf, Kim et al. [36] proposed another index to calculate the relevance of a term specific to a domain by considering the set of all the documents of a corpus as a single document, but this method can be inefficient (Witten et al. [64]). Bordea et al. [11] presented a point-wise mutual information-based *domain coherence index*, which is based on domain modeling of the corpus to find the term relevance.

2.2.3 Graph-Based Approaches. In terms of graph-based domain-specific keyword extraction, existing literature has two major limitations. First, to the best of our knowledge, all existing approaches except [55] are for general keyword/keyphrase extraction rather than domain-specific keyword extraction. Sarna and Bhatia [55] present a chi-square based probabilistic approach for domain-specific keyword extraction, but they do not incorporate the contextual semantics-based similarity of seed words with the corpus words. Second, using word embeddings to integrate multiple kinds of useful information into the graph-based random-walk model for domain-specific keywords extraction is not studied in the existing literature. On the other hand, our DSKE approach uses an initial lexicon of seed terms to compute the semantic relatedness score of corpus words with the seed words that will ultimately bias the relevance score of corpus words while recalculating the score using personalized `PageRank` depending on their vicinity with the target domain. Secondly, no existing approach combines the strengths of contrasting-corpora and

[4] https://www.mturk.com/.
[5] https://www.rappler.com/.

graph-based approaches, incorporating local and global semantic-coherence property for expansion of the initial lexicon of seed words using domain-specific keyword extraction.

3 DSKE APPROACH: DOMAIN-SPECIFIC KEYWORD EXTRACTION

We now present details of DSKE, our state-of-the-art method for domain-specific keyword extraction. The input to DSKE consists of (i) a corpus of tweets $D = \{t_1, t_2, \ldots, t_n\}$ (in our case a document is equivalent to a tweet) from which candidate words are extracted, and (ii) an initial lexicon $\mathcal{L}$ of (a few) seed words (such as *nigga*, *bitch*, *free*, *idiot*) that will be used to generate an expanded lexicon. We will present DSKE in the context of online social media in form of tweets, but the approach can also be applied to longer or more formal textual documents. The DSKE approach for lexicon generation consists of three main steps: (i) *candidate word* extraction, (ii) *contextual semantics-aware graph* construction, and (iii) finally *word ranking and lexicon* generation, which are further described in detail in the following sub-sections. Table 2 presents a list of symbols used in this article along with their brief descriptions.

3.1 Candidate Word Extraction

The first step is the extraction of an initial set of candidate words from the text-corpus D, which will be further processed through graph modeling for lexicon generation. The selection of candidate words from the corpus is based on their probabilistic relevance of being keywords. This is an important step because the final words for lexicon generation are selected from the initial set of candidate words. Therefore, the quality of candidate words has a defining effect on the overall performance of the approach. The candidate word extraction process consists of the following steps.

3.1.1 Noise Filteration. OSNs are conversational platforms where users use very informal and noisy language, rather than the formal writing style used in documents and news corpora. As a result, user-generated content on social media consists of special symbols including URLs, which are not content-bearing words relevant for lexicon generation. We perform extensive preprocessing to filter unnecessary and non-content bearing words from the corpus. In Twitter, retweets are prefixed with "RT", and users are tagged using "@" symbol, which are both filtered out. We also filter the URLs and hashtag symbol "#" as they do not contain keyword-relevant information. Additionally, we filter all the special symbols and numeric characters and convert the tweets to lower case to avoid any ambiguity.

3.1.2 POS Tagging and Word Selection. The noise-filtered tweets are passed through a **part-of-speech** (**POS**) tagger to assign a label for each word in a tweet (we use the Spacy[6] POS tagger). The important lexical units, i.e., keywords, in user-generated content are generally *noun* and *adjective* phrases [43, 61]. Therefore, we consider only the lemmatized form of the *noun* and *adjective* words for candidate word extraction. The other POS tags related to *pronoun*, *verb*, and *adverb* generally do not pertain to important keywords, and thus are filtered [6]. Finally, after removing duplicate words we obtain a set of candidate words for the corpus.

3.2 Contextual Semantics-Aware Graph Construction

Graph-based approaches like [55] are generally based on word co-occurrence counts and other simple graph measures to assign initial node weights, ignoring both local and global domain-specific contextual semantics between the words. However, a word in two different documents may have different contextual interpretations depending on its surrounding words [54]. For example, the

[6] https://spacy.io/.

Table 2. Symbols and Their Descriptions

Symbol	Description
D	Corpus of tweets
t_i	i^{th} tweet of corpus D
$\mathcal{L}$	Initial lexicon of seed words
$l \in \mathcal{L}$	A seed word in the Initial lexicon
$G = \langle W, E \rangle$	A graph with sets of nodes (words) W and edges E
N	Number of nodes in graph G
$w \in W$	A node (word) of W
$\mathcal{S}(w)$	Embedding-based similarity of word w
e_w	Embedding representation of word w
$\mathcal{P}(w)$	Co-occurrence-based contextual probability of a word w
$\mathcal{D}(w)$	Domain relevance of a word w
c_d	Domain-specific corpus
c_g	General corpus
$c(w)$	Frequency count of w
$c(l, w)$	Co-occurrence count of l and w
N_d	Number of terms in domain-specific corpus c_d
N_g	Number of terms in generic corpus c_g
$tf_d(w)$	Frequency count of w in c_d
$tf_g(w)$	Fequency count of w in one of the corpus $g \in c_g$
$\mathcal{V}(w)$	Vertex score of word w
$\mathcal{E}(w_i, w_j)$	Edge-weight between a pair of words w_i and w_j
B_d	Set of bigram keyphrases in c_d
f_b	Frequency of a bigram keyphrase $b \in B_d$
T	Set of trigram keyphrases
f_t	Frequency of a trigram keyphrase $t \in T$

word *Jihad*, when used in the context of the *Kashmir movement* will co-occur with words like *Pakistan*, *Kashmir*, and *fight*, whereas when used in the global context, it will co-occur with a different set of words like *terrorism*, *hamas*, *sharia*, as shown later in Figure 5 in Section 4.4.2. Most state-of-the-art approaches for domain-specific keyword extraction do not incorporate the contextual semantics of words as shown in Table 1; rather they simply use the co-occurrence of words [37, 49, 55]. However, some approaches utilize the context while calculating either the word score or edge score [48, 63], but existing approaches do not utilize word embeddings with other contextual semantics-based measures to compute the association between the words and conceptualizing domain. DSKE introduces three metrics to incorporate both local and global contexts for computing the initial vertex score between a set of seed words and corpus words, representing the strength of their contextual association.

The aim of our work is to identify the set of words/phrases from the extracted set of candidate words that are contextually associated and domain coherent with the seed words. We model the extracted candidate words as a contextual word co-occurrence graph, where the initial vertex score of each word represents the context-aware association between the word and the target domain, and the edge weights represent the co-occurrence frequency between every pair of words within

the tweets. We model the extracted set of candidate words as a word co-occurrence graph $G = \langle W, E \rangle$, where W is the set of nodes representing the candidate words and $E \subseteq W \times W$ is the set of edges connecting the nodes (words). An edge between a pair of nodes is created only if they co-occur in the corpus in at least one document (i.e., in some tweet). The weight assigned to a node (word) $w \in W$ is the average of three local and global context-based association metrics: (i) semantic similarity of w with the initial seed word set $\mathcal{L}$, (ii) probability of occurrence of seed words with w, and finally (iii) domain relevance of w with respect to a set of contrasting corpora. We describe the details next.

3.2.1 Local-Context Based Vertex Relevance. The initial weight assignment to the nodes of the word co-occurrence graph is an important step for domain-specific lexicon generation because it represents the context-based semantic relatedness of a corpus word with the domain of seed set $\mathcal{L}$. The local context of a word/phrase/sentence is based on its surrounding words/phrases/sentences. We use two *local context*-based similarity measures to compute the relevance score of lexical units representing the vertices.

Embedding-Based Semantic Similarity: The distributional representation or embedding of a word/concept/phrase is a numeric vector incorporating its contextual semantics in the corpus. The vector representation of contents can be generated at different levels of granularity—*word*, *phrase*, *sentence*, and *paragraph* [38]. We exploit distributional representations to find the contextual inclination of words with the seed words, computing the dense vector-based cosine similarity between them. The existing literature has many neural network-based methods to learn the vector representation of words. Bengio et al. [7] presented a feed-forward neural network model that jointly learns the word representation and statistical language model. Mikolov et al. [44] presented the computationally efficient and widely used word2vec approach to train word embeddings from an unlabeled corpus. They proposed two models: (i) a **continuous bag of words** (**CBOW**) model and (ii) skip-gram model. We use the skip-gram model over the noise-filtered corpus to train a word embedding model that maps each word of the corpus into the vector space of latent concepts. Each dimension of the word embedding represents a latent concept based on co-occurrence with other words in the context. We use the trained word-embedding vectors to measure the contextual semantics-based similarity between the lexicon of seed words and every candidate word $w \in W$ of the graph. The *embedding-based similarity* $\mathcal{S}(w)$ of each word w is the average of the cosine similarity of w with each seed word $l \in \mathcal{L}$ given as

$$\mathcal{S}(w) = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \cos(e_w, e_l), \tag{1}$$

where e_w and e_l represent the embedding vectors of words w and l, respectively.

Co-Occurrence-Based Contextual Proximity: In DSKE, we use an initial lexicon of seed words representing the conceptualizing domain. The frequent occurrence of a word with the seed words reflects its contextual proximity with them. For example, the word *jihad* generally occurs with terrorism-related words; however, in different corpuses it may occur with a different set of words depending on the conflicting zone discussed in each corpus like Palestine, Kashmir, and Somalia. To incorporate this contextual closeness, we define another metric, called co-occurrence-based contextual proximity ($\mathcal{P}$), to capture the co-occurrence of the word and seed words. We call it *local-context based proximity measure* because it captures the context between words based on their occurrence within a corpus rather than across the corpuses. For a word w, it is defined as the

average of the conditional probability of w with each seed word $l \in \mathcal{L}$, given as

$$\mathcal{P}(w) = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} p(l|w), \tag{2}$$

$$p(l|w) = \frac{c(l,w)}{c(w)}, \tag{3}$$

where $p(l|w)$ represents the conditional probability of l given w, $c(l,w)$ represents the co-occurrence count of l and w, and $c(w)$ represents the frequency count of w in the corpus.

3.2.2 Global-Context Based Vertex Relevance. Existing work uses the context of a word based on its surrounding words in the input corpus, but ignores its context relative to other corpora. To observe the context of words across corpora, we define a global context-based metric called *domain relevance*, which is a generalization of domain specificity [49].

Domain Relevance: As discussed earlier, a word may occur with different set of contextual words in different corpuses depending on the conceptualizing domain. This contextual information across corpora is not captured in the local context-based metrics. The existing literature has approaches that contrast a domain-specific corpus against a generic corpus or corpora to determine a word's relevance but does not incorporate local contextual semantics. As a result, they do not perform well in isolation on informal and noisy datasets [37, 49]. Therefore, we utilize the contrasting corpora-based domain relevance of words along with embedding-based semantic similarity and co-occurrence based proximity to observe their semantic proximity with the domain of $\mathcal{L}$. In DSKE, the *domain relevance* $\mathcal{D}$ of a word w is defined as the ratio of the occurrence probability of w in the domain-specific corpus to the average of its occurrence probability over the general corpora. If the domain-specific corpus is c_d and general corpora c_g, then the domain relevance $\mathcal{D}(w)$ of w is defined as

$$\mathcal{D}(w) = \frac{p_d(w)}{p_g(w)} = \frac{tf_d(w)/N_d}{\frac{1}{|c_d|}\sum_{g \in c_g} tf_g(w)/N_g}, \tag{4}$$

where $P_d(w)$ represents the occurrence probability of w in domain-specific corpus c_d, $P_g(w)$ represents the average of the occurrence probability of w in contrasting corpora c_g, N_d represents the number of terms in c_d, N_g represents the number of terms in one of the generic corpus g of c_g, $tf_d(w)$ represents the frequency count of w in c_d, and $tf_g(w)$ represents the frequency count of w in one of the corpus $g \in c_g$.

Finally, the vertex score $\mathcal{V}(w)$ of a word w, representing the contextual semantics-based similarity of w with the seed words, is the average of the three contextual semantics-based similarity metrics $\mathcal{S}(w)$, $\mathcal{D}(w)$ and $\mathcal{P}(w)$, given as

$$\mathcal{V}(w) = \frac{1}{3}(\mathcal{S}(w) + \mathcal{D}(w) + \mathcal{P}(w)), \tag{5}$$

3.2.3 Edge Score. In addition to the vertex score, we also capture the contextual semantics-aware association between every pair of vertices $(w_i, w_j) \in G$. Existing approaches have utilized the co-occurrence count of two words over windows of different sizes to assign the edge weight [8, 20, 43]. We define the edge-weight between a pair of words (w_i, w_j) as the number of tweets in which they co-occur; we do not use window-based co-occurrence count because tweet-level co-occurrence incorporates better context. The edge score $\mathcal{E}(w_i, w_j)$ is defined as

$$\mathcal{E}(w_i, w_j) = \sum_{t \in D} I_t(w_i, w_j), \tag{6}$$

$$I_t(w_i, w_j) = \begin{cases} 1 & if\ w_i, w_j \in t \\ 0 & otherwise \end{cases}, \tag{7}$$

where $I_t(w_i, w_j)$ is an identity function which is one when both the words co-occur in a sentence, otherwise zero.

Finally, we normalize the edge weights of G by dividing by E_{max} that represents the largest weight among all the edges, given as

$$\mathcal{E}(w_i, w_j) = \frac{\mathcal{E}(w_i, w_j)}{E_{max}}, \tag{8}$$

3.3 Words Ranking and Lexicon Generation

Given the weighted graph G with node weights $\mathcal{V}(w)$ and edge weights $\mathcal{E}(w_i, w_j)$, we apply PageRank to rank the nodes. In DSKE, weights having higher contextual semantics-based similarity with seed words are assigned higher weights emulating personalized PageRank [47] with the non-uniform node distribution serving as a *personalisation vector*. The non-uniform distribution of weights biases the computation toward specific nodes in the recursive update procedure allowing the nodes of the graph to spread their importance to other nodes depending on their weights. This spread of a node score is also affected by the weights of its adjacent edges such that for two nodes that are strongly connected, the flow of the weight score of one node will be higher than the other node. The final weight of a node is not only based on its contextual semantics with the initial lexicon of seed words but also its co-occurrence with the seed words.

To identify the most relevant words for expansion of the initial set of lexical units, we apply a modified PageRank algorithm [31] on the weighted undirected graph G with initial weights on both the nodes and edges. The importance score of a word is updated as follows:

$$\mathcal{V}^{'}(w_i) = \frac{1-d}{N} + C \sum_{w_j \in Adj(w_i)} \frac{\mathcal{E}(w_j, w_i) \times \mathcal{V}(w_j)}{|Adj(w_j)|}, \tag{9}$$

where $\mathcal{V}^{'}(w)$ represents the updated score of $w \in W$, d is a damping constant, C is a scaling constant, N represents the total number of words (nodes) in the graph, and $Adj(w_j)$ represents the adjacent nodes of word w_j. The damping constant d represents the probability of a random jump from one node of the graph to another. The modified PageRank is applied to update the score of each word until convergence. Finally, words are sorted based on their converged scores and we select the highly ranked words for domain-specific keyword extraction.

3.4 KeyPhrase Extraction

We use the extracted keywords to construct multi-word keyphrases—bigrams and trigrams—to expand the lexicon. Unlike existing works that typically extract bigram keyphrases based on the co-occurrence of underlying pair of extracted keywords [43], we define a threshold to filter out the keyphrases which might have occurred by chance.

Bigram Phrases: To filter out rare and insignificant bigrams, we define the *Average Bigram Frequency* (ABF) threshold as the ratio of the sum of the frequency count of all the bigrams generated from the corpus (excluding stopwords) to the number of bigrams, given as

$$ABF = \left\lfloor \sum_{b \in B_d} f_b / |B_d| \right\rfloor, \tag{10}$$

where B_d represents the set of all the bigram keyphrases generated from the corpus c_d, and f_b represents the frequency of one of the bigram keyphrase $b \in B_d$. Finally, a bigram keyphrase b is selected, if $f_b > ABF$.

Trigram Phrases: We also extract trigram keyphrases based on the expanded lexicon of single words. We first generate the list of candidate trigram keyphrases T from the corpus using a sequence of three words, such that the first and third words are connected by some stopword.[7] For example, in the preprocessed tweet—*Dedicated to our warriors of Ghazwaehind*, the set of candidate trigram keyphrases contain only one keyphrase—{*warriors of Ghazwaehind*}. We also experimented with the general trigram extraction approach of using all trigrams. However, we found that most of the trigrams are not very useful and introduce noise, and using stopword connected trigrams shows better performance. Finally, similar to the bigram threshold *ABF*, we also define a threshold for trigram keyphrases called *Average Trigram Frequency* (ATF). Finally, DSKE selects a candidate trigram keyphrase $t \in T$ if it has matching first and third words in the expanded lexicon of single words and its occurrence frequency $f_t > ATF$.

3.5 Language-Agnostic Keyword Extraction Approach

The language dependency of a keyword/keyphrase extraction technique is based on the availability of language parsers to select the potential candidate words in the initial phase of the extraction. An approach is called *language-agnostic* if it does not require a language tool during the keyword extraction process. We extend DSKE towards a language-agnostic domain-specific keyword extraction approach called LA-DSKE, which can be useful for languages with deficient language parsing tools. LA-DSKE does not require a language parsing tool for candidate word extraction, however, it has one extra component for irrelevant and stopword identification, as detailed below.

3.5.1 Global Stopword Index. In existing state-of-the-art approaches, *noun* and *adjective* words are generally considered as informative words that make up the candidate words. LA-DSKE follows the intuition that if we can identify the list of irrelevant words from a corpus, then its complement will represent the candidate words. Stopwords and other irrelevant words occur in every corpus irrespective of the domain of a corpus and do not bear information regarding the domain knowledge.

LA-DSKE curates a list of stopwords using a statistical filter from a corpora. Like the termhood-index [37], we first define a statistical index called *normalized stopword index* based on the intuition that a stop or irrelevant word frequently occurs in every corpus irrespective of the domain. Moreover, it is defined using the ranking of words rather than their absolute frequency. The *normalized stopword index* $\mathcal{N}_{C_i}(w)$, of a word w in a corpus C_i is calculated as

$$\mathcal{N}_{C_i}(w) = \frac{r_{C_i}(w)}{|V_{C_i}|}, \tag{11}$$

where $r_{C_i}(w)$ represents the rank of w in C_i. If V_{C_i} denotes the vocabulary for corpus C_i, then $r_{C_i}(w)$ is equal to $|V_{C_i}|$ for the most frequent word, $|V_{C_i}|-1$ for the second most frequent word and so on, such that for least frequent word $r_{C_i}(w) = 1$. The rank of a word is divided by the vocabulary size to normalize its value in the interval $(0, 1]$ and to avoid the effect of corpus size. For a stopword, if the value of $\mathcal{N}$ is calculated over multiple corpuses then its value will be high across the corpora. In contrast, the normalized stopword index value for relevant and content-bearing words will be high within the corpuses of their respective domain, and low in contrasting

[7] We use the stopwords maintained by *nltk*.

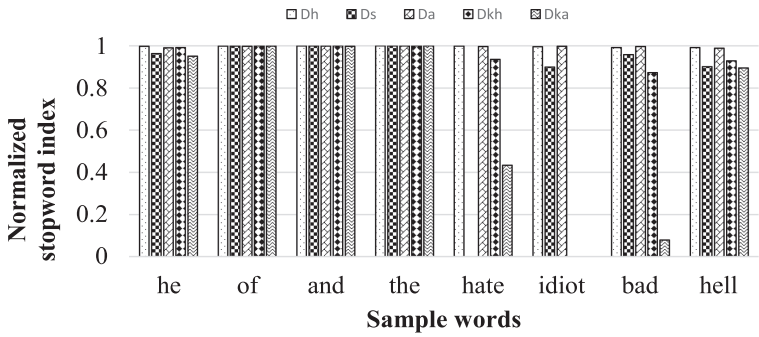


Fig. 1. $\mathcal{N}$ value of example words over five datsets. Details of the datasets are given in Section 4.

corpuses. For example, Figure 1 shows $\mathcal{N}$ values of a sample of four stopwords and four hate-bearing words. We can observe that the $\mathcal{N}_{C_i}(w)$ values of the four stopwords (he, of, and, the) are almost 1 and identical in all the five datasets. In contrast, values of the four hate-bearing words are comparable to stopwords in the first and third datasets because they are hate-bounded datasets but significantly lower for the remaining three datasets (even 0 in a few cases due to the absence of that word in the underlying dataset).

To incorporate the variation in the normalized stopword index value of a word across the corpuses and further favor a word having lower variation, we define the *global stopword index* for word w. Given a set of corpuses $C = \{C_1, C_2, \ldots\}$, in order to determine the global stopword index $\mathcal{G}_w$ of a word w, it is first ranked across each corpus C_i to compute the weighted *normalized stopword index*, with the weight given as the inverse of the adjustment factor δ_w that incorporates the deviation in the average ranking of w around the maximum ranking across the corpuses in an incremental manner, given as follows:

$$\mathcal{G}_w = \sum_{i=1}^{|C|} \mathcal{N}_{C_i}(w) \times \frac{1}{\delta_w}, \tag{12}$$

$$\delta_w = \sum_{i=1}^{|C|} \left(\left(\frac{\sum_{j=1}^{i} \mathcal{N}_{C_j}(w)}{i} \right) - \max\{\mathcal{N}_{C_1}(w), \ldots, \mathcal{N}_{C_i}(w)\} \right), \tag{13}$$

where δ_w is the adjustment factor incorporating the variation in the $\mathcal{N}$ values of the word. For example, if C has five corpuses, then to compute the global index value for each word, we start with two corpuses to find the average and maximum of $\mathcal{N}$ values, respectively, and compute the difference of these two values. Furthermove, we add a new dataset, compute the average and maximum of $\mathcal{N}$ values in three corpuses and again the difference between the two. We repeat this process to compute the difference until all the datasets are added. Finally, all the differences are added to compute the adjustment factor.

Rather than defining the adjustment factor as a simple average-based deviation, we define it incrementally because average-based deviation cannot incorporate the impact of individual datasets on the deviation. Due to increment-based deviation, for a fluctuating set of values, the adjustment factor δ_w will be higher in comparison to the average-based deviation for content-bearing words, accordingly $\mathcal{G}_w$ will be low. On the other hand, for stop and irrelevant words having similar values across the corpuses, δ_w will be low and accordingly $\mathcal{G}_w$ will be high. Finally, when $\mathcal{G}_w$ values are sorted, stopwords will be among the top of the sorted list, content-bearing words frequently occurring in certain domain-specific corpuses will be generally in the middle of the sorted list, and

Table 3. Global Stopword Index Value of Example Words Over Multiple Datasets

#Dataset	Global Stopword Index ($\mathcal{G}$)							
	Stopword				Content-bearing Words			
	he	**of**	**and**	**the**	**hate**	**idiot**	**bad**	**hell**
2	35.06	1,881.04	2,186.681	27,024	1.0	24.48	72.80	16.74
3	20.64	546.637	853.47	10,063.69	0.79	14.82	36.91	9.93
4	15.69	342.57	520.19	4,568.57	0.64	2.17	13.62	6.57
5	11.76	262.01	361.55	2,257.01	0.45	0.77	2.69	4.66
6	9.77	208.65	278.33	1,451.02	0.39	0.38	1.50	2.64

very rarely occurring words will be in tail end of the sorted list. Table 3 presents the $\mathcal{G}_w$ values for four content bearing and four stopwords when a dataset is added to the corpora C. We can observe that the $\mathcal{G}_w$ value for content-bearing words is very low in comparison to stop-words. Another interesting observation is that initially (i.e., for two datasets), the $\mathcal{G}_w$ value of some stop-words is equivalent to the content-bearing words. However, as the number of datasets is added into C, the difference in the $\mathcal{G}_w$ value of stop and content-bearing words widens due to the deviation and adjustment factor. We also analyzed the impact of the order of corpuses and found that it has an insignificant affect on the value of the adjustment factor.

3.5.2 Stopwords and Candidate Words Selection. Following the calculation of $\mathcal{G}_w$ for each word of the corpus, we generate a list of stopwords. We analyzed the distribution of $\mathcal{G}_w$ for annotated keywords and found that most of the stopwords are ranked at the top of the list as expected. We rank the words based on their $\mathcal{G}_w$ value and select the top 1% as the final set of stopwords, and all the remaining words are treated as candidate words.[8] Finally, we model the identified words as a word co-occurrence graph and apply the modified PageRank method to rank and select the domain-specific keywords (as defined in Sections 3.2 and 3.3, respectively).

4 EXPERIMENTAL SETUP AND RESULTS

We evaluate DSKE over both informal and formal text to establish its efficacy. However, our main focus is on informal social media content from Twitter. The informal text comprises two main datasets—D_1 and D_2, which are used to create five different subsets. The D_1 data is a benchmark dataset of 80,000 tweets related to three categories of offensive language—*hateful*, *spam*, and *abusive* [26]. It also includes *normal* tweets. Since the authors of D_1 have released only tweet ids due to privacy, we crawled Twitter data and collected 64,963 tweets and their related metadata information to construct the dataset D_1 (15,037 of the originally listed tweet ids were suspended or deleted).

For qualitative evaluation, we randomly select 1,000 tweets each from *hateful*, *spam*, and *abusive* categories, and these subsets are denoted as D_h, D_s, and D_a, respectively. The reason to select a smaller subset is that it is extremely hard to manually determine the ground truth keywords for the full set of 64,963 tweets; instead, we perform evaluation over the random set of 1,000 tweets from each category. We also show results on the subsets of randomly selected 500 tweets.

[8]We also experimented with top 2% and 3%, but 1% worked the best since relevant keywords are generally ranked higher in the ranking list. Whereas some relevant keywords are also present at lower ranks, the majority of low ranked words are irrelevant.

Table 4. Dataset Statistics of D_2

Dataset	Keyphrases for crawling	#tweets retrieved	#unique tweets
Khalistan movement	free Khalistan, Referendum2020, KhalistanMovement, #freedomofkhalistan, Khalistan2020, SikhHardliners, FreeJaggiNow	27,779	3,888
Kashmir terrorism	time for jihad, Call for Jihad, Gazwa e Hindh, jihad for kashmir, kashmir mujahideen, Ghazwaehind, mujahidin zindabad	5,717	560

We use **Natural Language Understanding**[9] (**NLU**) to extract the initial set of candidate keywords because it is a state-of-the-art tool for keywords extraction. As a result, we obtain 286,343, and 264 keywords, respectively, for the three datasets. Furthermore, to filter out useless and unimportant keywords, we use human annotators to label the ground truth keywords. In this direction, we asked five annotators to rate the extracted *hateful*, *spam*, and *abusive* keywords on an 11-point scale from 0 to 10, where a keyword is assigned 0 when the annotator is 100% confident that the keyword does not belong to a particular category, and it is assigned 10 when the annotator is 100% confident that the keyword belongs to a particular category. The five annotators were doctoral students of our lab working in the field of OSN analysis, and who are well aware of *hateful*, *abusive*, and *spam* language problems in social media. We take more annotators to limit the biasness in ground truth labels. Finally, we take the average of the rating scores of the five annotators and label a keyword as *hateful*, *spam*, or *abusive* if the average rating score is greater than half of the scale range, that is 5. Following this process, we obtain an annotated set of 72 hate words (A_h), 128 spam words (A_s), and 99 abusive words (A_a).

To further evaluate DSKE, we crawled another dataset D_2 of Tweets, based on 14 radical keyphrases related to *Khalistan* and *Kashmir* movements inciting inter-religious conflict. In the South Asian region, Kashmir and Khalistan issues are the prime reason of conflict between two nuclear powers and accordingly, the main source of youth radicalization and extremist content in online social media.[10] Following the revocation of *Article 370* in the Indian state of Jammu and Kashmir, there was a rampage of radical tweets from both India and Pakistan. We crawled the dataset related to the *Khalistan movement* and *terrorism in Kashmir* using Twitter stream API based on 14 keyphrases, such as *call for jihad*, *time for jihad*, *free Khalistan*. The complete list of keyphrases is given in the second column of Table 4. During the period from 5^{th} August 2019 to 28^{th} August 2019, we crawled a total of 33,496 tweets out of which 27,779 were related to Khalistan movement and 5,717 were Kashmir related tweets. After removing duplicate tweets, we obtain the second dataset D_2, comprising 4,448 tweets (3,888 tweets were related to Khalistan movement and 560 tweets were related to Kashmir). We trained 100-dimensional embedding vectors on D_2 as discussed above and randomly select 1,000 tweets from the set of Khalistan movement-related tweets called D_{kh} and all the 560 tweets from the set of Kashmir related jihad inciting tweets represented as D_{ka}. Next, using human annotation, we extracted 57 and 84 keywords for Khalistan and Kashmir, denoted as A_{kh} and A_{ka}, respectively.

DSKE is evaluated over these five datasets D_h, D_a, D_s, D_{kh}, and D_{ka} created from D_1 and D_2. Brief statistics of D_2, including the list of keyphrases used for data crawling, are given in Table 4, whereas Table 5 presents the statistics of the five evaluation datasets. DSKE is evaluated using three standard evaluation metrics—*precision*, *recall*, and *F-score* at K. *Precision* at K, denoted $P@K$,

[9] https://www.ibm.com/in-en/cloud/watson-natural-language-understanding.
[10] https://www.orfonline.org/expert-speak/42391-cyber-jihad-biggest-challenge-kashmir/.

Table 5. Statistics of Five Datasets Used for Evaluation

	Benchmark dataset D_1					Crawled dataset D_2		
Category	Abusive (D_a)	Hateful (D_h)	Spam (D_s)	Normal (D_n)	Total	Khalistan (D_{kh})	Kashmir (D_{ka})	Total
#tweets	12,878	2,740	9,048	40,297	64,963	3,888	560	4,448
#evaluation tweets	1,000	1,000	1,000	1,000	4,000	1,000	560	1,560

represents the fraction of correctly identified words from the set of top K extracted words, whereas *recall* at K, $R@K$, represents the fraction of correctly identified words with respect to the set of manually annotated words. Finally, *F-score* at K denoted $F@K$, is the harmonic mean of $P@K$ and $R@K$. In the experimental evaluation, we set the value of damping and scaling constants – d and C at 0.85 and 0.95, respectively [13, 31].

Instead of using the pre-trained word embeddings, we use `word2vec` [44] to train the embeddings from scratch due to two main reasons: (1) Tweets differ substantially from the corpus used to pre-train the word2vec embeddings, and (2) training from scratch allows us to better incorporate the local contextual information. For training word2vec on tweets, we set the *window* and *min_count* parameters as 5 and 1, respectively, where *window* represents the maximum distance between a target and context words and *min_count* is a threshold such that words having occurrence count lower than this threshold are filtered. We learn 100 dimensional embeddings of words from D_1 to incorporate local contexts. The learning was performed on a Linux machine having 16 GB RAM and 500 GB hard disk using Intel Xeon processor. The source code for DSKE and datasets are available at https://github.com/iammohdfazil/DSKE.

4.1 Comparative Evaluation

DSKE is compared with three baselines methods, two contrasting corpora-based approaches [37, 49], six graph-based approaches [8, 24, 43, 53, 55] and one deep learning-based approach [66]. We implemented all the baseline and state-of-the-art approaches that are briefly described below.

Frequency: This baseline approach extracts and ranks the words based on their frequency count in the text corpus.

tf-idf: This baseline ranks and selects words based on their tf-idf value in the corpus.

Embedding: Instead of using pre-trained word embeddings, the `word2vec` [44] method is used to train the embeddings. A detailed description of the embedding training is given in the last paragraph of Section 4. This baseline calculates the embedding-based similarity of words with the lexicon of seed words to extract the top-ranked contextually similar domain-specific words.

TDS [49]: The ***term domain specificity*** (**TDS**) observes the relevance of a term within a domain-specific corpus. It is the ratio of the occurrence probability of a term t in a domain-specific corpus c_d to the occurrence probability of t in a generic corpus c_g defined as

$$tds(t) = \frac{p_d(t)}{p_g(t)} = \frac{tf_d(t)/N_d}{tf_g(t)/N_g}, \tag{14}$$

where $p_d(t)$ and $p_g(t)$ represent the probability of occurrence of term t in c_d and c_g, respectively.

TH [37]: The *termhood* (TH) index is based on the idea that a term specific to a domain will occur frequently in the domain-specific corpus compared to another corpus. Unlike TDS which considers the absolute frequency of a term, the TH index considers the term's rank in

the corpus vocabulary as defined below:

$$th(t) = \frac{r_d(t)}{|V_d|} - \frac{r_g(t)}{|V_g|}, \tag{15}$$

where $|V_d|$ is the vocabulary size of the domain-specific corpus c_d, and $r_d(t)$ represents the rank of term t in c_d which is equal to $|V_d|$ for the most frequent term, and $|V_d| - 1$ for the second most frequent term, and so on.

RAKE [53]: This is an unsupervised and language-independent method for extracting keywords from a document. Based on a set of stopwords, phrase delimiters, and word delimiters, RAKE partitions a document to generate the list of candidate words. Furthermore, a word co-occurrence graph is constructed and a metric based on frequency and degree is defined and computed for each word. Further, a score is assigned to each keyword as the sum of the scores of each constituent word of the keyword. Finally, keywords are sorted based on the computed score and top-ranked keywords are selected as the final keywords.

TextRank [43]: This approach first identifies the nouns and adjectives as candidate words and models them as a word co-occurrence graph. Furthermore, PageRank is applied on the graph to rank and select the relevant keywords. TextRank can be applied to directed/undirected, or weighted/unweighted graphs.

Corpus-Independent [63]: Like [43], this article also considers only nouns and adjectives as the candidate words and models them as a weighted graph. It uses embedding and local statistics to compute *informativeness* and *phraseness* of a word/phrase. Furthermore, these two measures are used to calculate the attraction score between the pairs of words/phrases, representing the edge weight between them. Finally, biased PageRank is applied on the weighted graph and high-ranked words are considered as keyphrases.

PositionRank [24]: This method exploits the positions of a word's occurrences, as well as its frequency, which are used within a biased PageRank method to rank and select the most relevant keywords from a corpus.

CNW [8]: This method models the corpus words as a graph and defines various structure-based metrics, such as *distance from central node*, *selectivity centrality*, and *degree centrality*. It also includes *position of a word within a sentence* and *neighbor-based importance* to assign initial weights to words. Furthermore, the bias PageRank is applied to rank and select the keywords.

SARNA [55]: This method generates an expanded set of domain-specific keywords from an initial set of seed words. It first constructs a conditional probability-based directed word co-occurrence graph consisting of some seed words. Further, it uses chi-square based significance analysis to filter the statistically insignificant edges. Finally, words adjacent to seed words are selected as keywords. It neither uses any initial node score nor graph-based ranking to rank nodes.

TC-LSTM [66]: This is a deep neural network-based approach for keywords extraction. It uses a target center-based LSTM model to learn the representation of a target word using its contextual information in both forward and backward directions. It further applies an attention mechanism to assign higher weights to important words before passing them through a dense network for the classification task.

Table 6 presents the performance evaluation results of DSKE in terms of all the three evaluation metrics in comparison to the baseline and state-of-the-art approaches over the five datasets. In the experimental evaluation of DSKE and comparative approaches, we used a random sample of 1,000 tweets from the normal category (D_n) of benchmark dataset D_1 as the contrasting corpus. We

Table 6. Comparative Performance Evaluation of DSKE with 11 Baseline and State-of-the-art Approaches Over Five Datasets Using an Initial Lexicon of 3 Seed Words

Approach	Datasets														
	D_h			D_s			D_a			D_{kh}			D_{ka}		
	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80
Frequency	0.163	0.181	0.171	0.150	0.094	0.115	0.163	0.133	0.146	0.225	0.40	0.288	0.238	0.226	0.232
tf-idf	0.063	0.069	0.066	0.238	0.148	0.183	0.138	0.112	0.124	0.025	0.044	0.032	0.038	0.036	0.037
Embedding	0.075	0.083	0.079	0.113	0.070	0.086	0.325	0.265	0.292	0.238	0.422	0.304	0.363	0.345	0.354
TDS [49]	0.150	0.167	0.158	0.325	0.203	0.250	0.025	0.020	0.022	0.075	0.133	0.096	0.10	0.095	0.098
TH [37]	0.163	0.181	0.171	0.149	0.085	0.108	0.163	0.133	0.146	0.057	0.067	0.061	0.100	0.071	0.083
RAKE [53]	0.038	0.042	0.040	0.00	0.00	0.00	0.063	0.058	0.059	0.00	0.00	0.00	0.225	0.214	0.219
TextRank [43]	0.238	0.264	0.250	0.30	0.188	0.231	0.325	0.265	0.292	0.275	0.489	0.352	0.30	0.288	0.293
Corpus-Ind. [63]	0.032	0.034	0.036	0.188	0.118	0.147	0.118	0.102	0.106	0.313	0.535	0.397	0.05	0.041	0.036
PositionRank [24]	0.225	0.250	0.237	0.413	0.258	0.317	0.288	0.235	0.258	0.288	0.511	0.368	0.325	0.310	0.317
CNW [8]	0.30	0.333	0.316	0.575	0.359	0.442	0.325	0.265	0.292	0.338	0.60	0.432	0.350	0.333	0.341
SARNA [55]	0.256	0.153	0.191	0.326	0.145	0.191	0.013	0.010	0.011	0.235	0.093	0.125	0.272	0.104	0.141
TC-LSTM [66]	0.325	0.362	0.342	0.537	0.336	0.413	0.40	0.323	0.358	0.363	0.644	0.464	0.438	0.416	0.427
DSKE	**0.338**	**0.375**	**0.355**	**0.725**	**0.453**	**0.558**	**0.488**	**0.398**	**0.438**	**0.425**	**0.756**	**0.544**	**0.513**	**0.488**	**0.499**

The bold entries in these tables represent the best performance evaluation result values among the various methods analyzed therein.

can observe that DSKE outperforms all the methods. The contrasting corpora-based approaches (TDS and TH) show the worst performance, whereas graph-based approaches show better performance, except RAKE and Corpus-Independent. One interesting observation is that among the three baseline approaches—Frequency, tf-idf and Embedding—the keywords extracted using Embedding outperform the other two over the three datasets D_a, D_{kh}, and D_{ka}. On these datasets, Embedding is also competitive with the graph-based approaches. The embedding-based baseline only finds the sum of the similarity between the set of seed words and candidate words, and further ranks the words based on this similarity. The moderate performance of embedding-based baseline points to the overall strength of DSKE, which uses semantic similarity based on the distributional representation of words as one of the metrics of association between the corpus words and the initial set of seed words.

We examine the comparative performance evaluation of DSKE with other graph-based approaches because it is also a graph ranking-based approach. Among the graph-based approaches, RAKE consistently performs poorly over all the five datasets. It is because RAKE ranks a word based on three metrics—(i) degree, (ii) frequency, and (iii) ratio of degree and frequency, wherein degree favor words that occur in longer keyphrases that are generally not used in informal content like OSN text. However, it performs considerably well on formal texts, as shown in Table 12 because formal text generally contains large keyphrases in comparison to informal texts. Corpus-Independent also shows poor performance except for D_{kh}. The initial score of a node in the Corpus-Independent approach is assigned based on *informativeness* and *phraseness*, wherein *pharseness* measures the phrase formation probability of a word with other words. As discussed above, informal texts generally do not contain large phrases; therefore, it performs considerably poorly. However, it shows better performance on format texts. On the other hand, the collective node weight-based approach CNW performs well. We also performed a comparative evaluation with a deep learning approach TC-LSTM, which performs better in comparison to other approaches. However, CNW and TC-LSTM both have lower performance compared to DSKE, which also outperforms the only other domain-specific keyword extraction approach SARNA by up to 40 points in terms of precision at K@80 (e.g., for D_s), up to 66 points in terms of recall at K@80 (e.g., for D_{kh}),

Table 7. Performance Evaluation Results Over the Five Datasets Each Containing 1,000 Tweets Considering Initial Lexicon of 3 Seed Words

Evaluation metric	Dataset D_1			Dataset D_2	
	D_h	D_s	D_a	D_{kh}	D_{ka}
P@80	0.338	**0.725**	0.488	0.425	0.513
R@80	0.375	0.453	0.398	**0.756**	0.488
F@80	0.355	**0.558**	0.438	0.544	0.499

The bold entries in these tables represent the best performance evaluation result values among the various methods analyzed therein.

and up to 42 points in terms of f-score at K@80 (e.g., D_{kh}). Among the other approaches, CNW performs best because it was applied to extract keywords from social media texts, and accordingly the authors designed appropriate graph-based metrics to assign initial weight to nodes. Though it also performs well over formal texts in terms of precision, it shows poor performance in terms of F-score in comparison to other approaches that are specifically designed for keyword extraction from formal texts. The other keyword extraction approaches like TextRank and PositionRank also show reasonable performance, but they are not comparable to DSKE. This is because these approaches are targeted for formal texts wherein metrics used to assign initial node weight do not consider contextual semantics. These comparative evaluation results conclusively show that DSKE outperforms state-of-the-art approaches by a big margin across the informal social media text datasets, showing the effectiveness of combining both the local embedding based contextual semantics and the global context from contrasting domain corpora.

4.2 DSKE Evaluation Results

We now evaluate the efficacy of DSKE at $K = 40$, 60, and 80. We utilize three initial seed words ($|\mathcal{L}| = 3$), and used a randomly chosen 1,000 normal tweets from D_1 as the contrasting corpus C.

Table 7 shows the performance evaluation results at $K = 80$ over the five datasets with each (except D_{ka}) having 1,000 tweets; the Kashmir dataset D_{ka} has only 560 tweets. We can observe that in terms of P@80, the performance of DSKE does not seem encouraging over the hate dataset. This is because the extracted set has many words like *nazi*, *muslim*, *black* that are contextually used in hateful tweets but they were labeled by the annotators as *non-hatred* words. Additionally, many words like *terrorism*, *russia*, *gay*, which are subject to hatred in certain contexts were not extracted by the NLU, and these are missing from the annotated set of words. In the table, values in bold typeface represent the best performance for that metric among all the five datasets.

For better understanding, the top-60 keywords identified by DSKE over the five datasets are shown in Table 8. Words in bold represent the correctly retrieved words by DSKE. In terms of P@80, DSKE performs the best on the spam dataset D_s because it has the highest number of manually annotated ground truth keywords, eventually increasing the precision and hampering the recall. On the other hand, in terms of R@80, DSKE performs best over the Khalistan dataset D_{kh}, as shown in the second row of Table 7. This is because the Khalistan dataset has the lowest number of manually annotated ground truth keywords, and eventually, the highest recall. The comparative performance of DSKE over the different values of $K = 80, 60$, and 40 over the five evaluation datasets is summarized in Figure 2. It can be observed that as we take fewer top-ranked keywords for lexicon generation, precision increases sharply, whereas recall degrades as expected, resulting in lower F-scores as K decreases.

Table 8. Top-60 Keywords Over the 5 Datasets Using an Initial Lexicon of 3 Seed Words

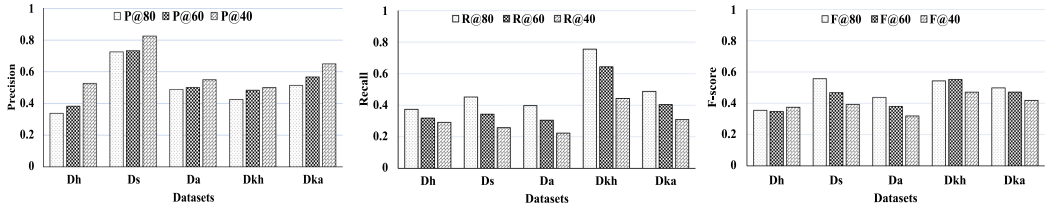
Dataset				
Hate (D_h)	Spam (D_s)	Abusive (D_a)	Khalistan (D_{kh})	Kashmir (D_{ka})
Seed Words				
nigga, fuck, bitch	*free, click, show*	*fuck, idiot, ass*	*khalistan2020, sikh, freejagginow*	*kashmir, jihad, ghazwaehind*
Extracted Words				
idiot, **nigga**, **bitch**, people, **hell**, **ass**, **bad**, trump, **disgusting**, **stupid**, **hate**, **ugly**, **fuck**, **sick**, man, **retard**, woman, tired, country, **evil**, time, syria, day, girl, **isis**, potus, **bloody**, **attack**, crazy, **damn**, thing, **feminist**, parissaxo, everyone, life, iraq, friend, **hoe**, **nasty**, god, black, **bastard**, president, way, muslim, anyone, nazi, religion, word, obama, **republican**, tweet, one, gay, today, htt, war, terrorism, russia, job	**new**, **free**, **video**, **youtube**, **thank**, **today**, melon, day, **game**, **ebay**, **available**, **sale**, **event**, **home**, people, **news**, **set**, time, **app**, **april**, **great**, **book**, **card**, **store**, black, **chance**, **link**, **website**, **show**, **car**, **night**, **fuck**, **weekend**, man, **girl**, **first**, **shoe**, **movie**, **amazon**, **job**, size, **winner**, **giveaway**, **month**, visit, **good**, **daily**, **season**, **thing**, **pic**, white, red, case, look, business, **code**, ticket, full, **music**, way	**idiot**, fucking, **ass**, **bitch**, **bad**, **stupid**, **ugly**, **sick**, **fuck**, people, **hell**, day, **trump**, fuckin, **nasty**, look, **evil**, man, way, **damn**, thing, time, **hate**, jesus, **terrify**, **hoe**, **bloody**, **annoying**, get, today, guy, **disgusting**, **shit**, **insane**, world, andyrichter, sequel, bag, **retard**, someone, **nigga**, **syria**, obama, **girl**, everything, **awful**, **asshole**, one, state, show, everyone, **shitty**, **dick**, video, school, anyone, **terrible**, something, friend, country	world, **referendum2020**, hindustan, **freekashmir**, **khalistan**, **kashmiris**, **kashmiri**, august, government, modi, **freedom**, **movement**, **voice**, imrankhanpti, rsrobin1, turn, one, majorgauravarya, british, **right**, **khalistan2020**, **free**, country, **terrorist**, **sikh**, case, brother, johal, state, **jagtar**, pakistan, kashmirbanaygapakistan, **independence**, people, same, **kashmirunderthreat**, muslim, **kashmir**, geeta_mohan, **citizen**, other, **guru**, narendramodi, army, nagaland, day, time, **kashmirkhalistan_ajointcall**, **kashmirwantsfreedom**, **referendum**, india, **flag**, trial, **freejagginow**, punjab, borisjohnson, **community**, singh, hindu, indian	**ghazwaehind**, **muslim**, **kashmir**, **time**, **jihad**, india, **pakistan**, indian, **army**, officialdgispr, **allah**, **kashmirhamarahai**, **zaidzamanhamid**, **war**, imrankhanpti, **ready**, world, **islam**, **pak**, good, **savekashmirso**, **zionist**, **kashmiri**, modi, people, year, **right**, **kashmirbleed**, **solution**, **peaceforchange**, **final**, **article370**, **pakistani**, rss, **standwithkashmir**, turkey, hindu, pid_gov, full, **taliban**, **decision**, rahulgandhi, **force**, nation, amitshah, **savekashmirforhumanity**, old, politician, reason, **quran**, **battle**, cnn, **kufr**, sir, thing, wrong, democracy, **kaffir**, fawadchaudhry, **allahuackber**

Bold indicates correctly identified keywords. (Disclaimer: offensive words, while abhorrent, are presented as is, without filtering.)

Table 9. Extraction Statistics of Bigram and Trigram Keyphrases

Dataset	Bigram			Trigram		
	#Keyphrase	#Correct keyphrase	Precision	#Keyphrase	#Correct keyphrase	Precision
Hate (D_h)	30	18	60.00	16	12	75.00
Spam (D_s)	66	36	54.54	14	12	**85.71**
Abusive (D_a)	98	86	**87.75**	33	21	63.63
Khalistan (D_{kh})	90	41	45.55	26	18	69.23
Kashmir (D_{ka})	35	10	28.57	12	8	66.67

The bold entries in these tables represent the best performance evaluation result values among the various methods analyzed therein.



(a) Precision (b) Recall (c) F-score

Fig. 2. Performance evaluation results for different values of K over the five datasets, for $K = 80, 60, 40$.

4.2.1 Keyphrase Results. This section presents the performance evaluation results of DSKE for extraction of bigram and trigram keyphrases. Firstly, we extracted the *bigram* and *trigram* keyphrases using the approach discussed in Section 3.4. The extracted keyphrases are further manually labeled by five annotators using the same aforesaid approach that was adopted for single words. For the five datasets, D_h, D_s, D_a, D_{kh}, and D_{ka}, DSKE extracts 30, 66, 98, 90, and 35 bigram

Table 10. List of 20 Bigram Keyphrases Extracted Using DSKE Over Five Datasets

Hate (D_h)	Spam (D_s)	Abusive (D_a)	Khalistan (D_{kh})	Kashmir (D_{ka})
idiot trump, **stupid idiot**, **idiot woman**, nigga bitch, **ass nigga**, **fuck nigga**, **nigga stupid**, **nigga hate**, **ugly nigga**, nigga tired, **ass bitch**, **bitch ass**, bad bitch, ass people, hate people, people hate, black people, **bad ass**, **stupid ass**, **ugly ass**	**new app**, **new video**, new set, free today, **free game**, **free download**, **youtube video**, **music video**, video today, set today, day thank, **win daily**, **ebay store**, sale today, **online store**, business people, great time, **app store**, **amazon music**, **online business**	**stupid idiot**, idiot world, **ugly fuck**, **sick fuck**, bad everything, **damn stupid**, stupid people, **fuckin stupid**, something awful, **fuckin annoying**, **shit nasty**, **stupid shit**, hate anyone, nasty friend, **annoying ass**, **bloody hell**, **evil shit**, one thing, **hoe shit**, **disgusting asshole**	**khalistan khalistan2020**, **kashmir khalistan2020**, **khalistan2020 kashmirwantsfreedom**, **pakistan khalistan2020**, majorgauravarya imrankhanpti, **free khalistan2020**, **jagtar singh**, **freekashmir khalistan2020**, **support referendum**, freedom freekashmir, **khalistan2020 movement**, zaidzamanhamid pakistan, **support khalistan2020**, khalistan sikh, **free khalistan**, **khalistan referendum**, sikh people, punjab india, **freekashmir kashmirkhalistan_ajointcall**, hindu india	**kashmir ghazwaehind**, **pakistan ghazwaehind**, pakistan india, **kashmirhamarahai ghazwaehind**, **kashmir jihad**, **ghazwaehind standwithkashmir**, muslim world, muslim kashmiri, pakistan kashmir, right time, rss india, pakistan turkey, pakistan taliban, **jihad ghazwaehind**, peaceforchange officialdgispr, article370 kashmirhamarahai, **kashmirhamarahai standwithkashmir**, modi rss, final solution, **final battle**

Bold indicates correctly identified keyphrases. (Disclaimer: offensive words, while abhorrent, are presented as is, without filtering.)

Table 11. Trigram Keyphrases Extracted Using DSKE Over Five Datasets

Hate (D_h)	Spam (D_s)	Abusive (D_a)	Khalistan (D_{kh})	Kashmir (D_{ka})
trump an idiot, **fuck that nigga**, **hate this nigga**, **hate a nigga**, nigga just tired, **nigga is crazy**, **bitch i hate**, hell these people, people are disgusting, hate when people, nazi is bad, **ugly as hell**, **crazy as hell**, **god i hate**, **hate the way**, **sick and tired**, iraq and syria, syria or iraq	**new on ebay**, **win a new**, **win the new**, **win a free**, **movie for free**, **online for free**, **download the free**, **win a set**, today and people, **win a home**, **download the app**, **chance to win**, **giveaway to win**, **win an amazon**	**fuck that bitch**, **bad as fuck**, **fuck this stupid**, **stupid as fuck**, **ugly as fuck**, **sick as fuck**, **fuck i hate**, **annoying as fuck**, **fuck that shit**, **fuck that guy**, **bitch so bad**, **thing a bitch**, **bitch i hate**, **hate you bitch**, **ass so bad**, **bad as hell**, **stupid as hell**, **ugly as hell**, sick of people, people are sick, **damn i hate**, **annoying as hell**, hate when people, people are disgusting, trump after show, one this day, **nasty as shit**, **man i hate**, **thing i hate**, **hate this nigga**, **hate the world**, hate my mom, hate this school, nigga have one, shit in school, show the world	**sikh for khalistan2020**, **kashmir and khalistan2020**, **khalistan2020 with kashmir**, **khalistan2020 and freekashmir**, people of khalistan2020, **same for khalistan2020**, **khalistan and kashmir**, **kashmir and khalistan**, khalistan from indian, **freedom of khalistan**, **freedom for khalistan**, **and other khalistan**, **khalistan and kashmiris**, **movement of khalistan**, **community in khalistan**, india and indian, india then pakistan, india is terrorist, government of india, **kashmiri and sikh**, pakistan on kashmir, kashmir and punjab, kashmir and free, people of kashmir, time for pakistan, pakistan should support, the same time, one for freedom, flag on august, flag before august, same with kashmiris, voice of nagaland	**time for ghazwaehind**, **time to ghazwaehind**, **ready for ghazwaehind**, **jihad for kashmir**, kashmir those people, **time for jihad**, **jihad against india**, **ready for jihad**, people of pakistan, **war on islam**, right of decision, wrong or right, **democracy is kufr**

Bold indicates correctly identified keyphrases. (Disclaimer: offensive words, while abhorrent, are presented as is, without filtering.)

keyphrases, respectively (see the statistics given in the second column of Table 9). The third and fourth columns represent the number of correctly identified bigram keyphrases and the respective precision values. It can be observed from the first four columns that DSKE extracts not only the highest number of bigram keyphrases for abusive content but also the highest percentage of correct keyphrases. On the other hand, Kashmir-related bigram keyphrases show the worst performance. Table 10 shows the list of 20 bigram kephrases extracted by DSKE from the five datasets.

Similarly, we applied DSKE for trigram keyphrases extraction, and the fifth, sixth, and seventh columns of Table 9 present the underlying results. We can observe the DSKE is very effective in extracting trigram keyphrases, especially on the spam dataset D_s (85.71% precision). Table 11 shows the extracted trigram keyphrases, wherein trigrams in bold typeface represent the correctly extracted ones. DSKE performs very well on all the five datasets.

4.3 Evaluation on Formal Texts

To further establish the efficacy of DSKE over formal text, we perform a comparative evaluation of DSKE versus the baseline and other methods, over the widely used Hulth benchmark dataset (Hulth [33]), which consists of 2,000 INSPEC abstracts of journal papers from Computer Science and Information Technology. The dataset is divided into training, validation, and testing sets containing 1,000, 500, and 500 abstracts, respectively. We evaluated DSKE over the validation set as done in [43] using an initial set of 3 seed words. However, some existing approaches use the test set. We select the top 3,400 keywords for DSKE and other approaches because it is close to the annotated set which is 3,487, and the corresponding results are presented in Table 12. We can observe from the table that CNW performs best in terms of precision but has a very low recall value.

Table 12. Comparative Performance Evaluation of DSKE on Journal Abstracts

Approach	Huth Dataset		
	precision	recall	f-score
Frequency	0.5373	0.5239	0.5306
Tfidf	0.5376	0.5242	0.5309
Embedding	0.5841	0.5695	0.5767
TDS [49]	0.3903	0.0877	0.1433
TH [37]	0.4286	0.0069	0.0135
RAKE [53]	0.4359	0.4250	0.4304
TextRank [43]	0.5891	0.5744	0.5817
Corpus-Ind. [63]	0.4328	**0.6496**	0.5101
CNW [8]	**0.7139**	0.3378	0.4586
PositionRank [24]	0.6056	0.5905	0.5975
SARNA [55]	0.4732	0.0278	0.0525
TC-LSTM [66]	0.604	0.589	0.597
DSKE	0.6400	0.6240	**0.6319**

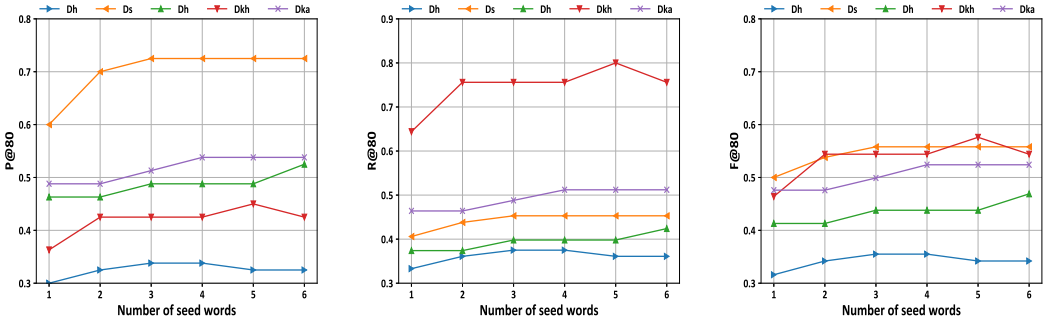
The bold entries in these tables represent the best performance evaluation result values among the various methods analyzed therein.

Similarly, TC-LSTM shows comparative performance, but DSKE outperforms it in term of all the three metrics. The corpus-independent approach has lower precision but the highest recall. Nevertheless, we can observe that DSKE performs best in terms of F-score, providing the best balance between precision and recall.

It can be observed from Tables 6 and 12 that the performance of the DSKE is more significant over the informal texts in comparison to the formal texts. In this regard, one interesting observation is that DSKE significantly outperforms all approaches over social media text. This is mainly due to the fact that formal texts are written using syntactic and semantic rules, and accordingly existing work uses weight assignment metrics based on those rules. However, DSKE still performs considerably better in terms of F-score.

4.4 Discussion

4.4.1 Impact of Seed Words on Evaluation Results. DSKE assigns the initial vertex score of words based on their contextual semantics-based similarity with an initial set of seed words, which also biases the identification of domain-specific lexical units. Therefore, we study the impact of the number of seed words on the final set of domain-specific keywords by varying the number of initial words from 1 to 6. Figure 3 presents the impact of the number of seed words on performance. We observe lower performance when only 1 or 2 seed words are used to compute the initial vertex score of candidate words. As we increase the number of seed words, the performance typically improves. We can observe from Figures 3(a), 3(b), and 3(c) that generally performance stabilizes at 3 seed words that is why all the evaluations in this article use 3 seed words. Another interesting observation is that along with the performance, the ranking of words also changes, as shown in Figure 4, which shows how the ranking of four example words (“hate”, “bastard”, “video”, “ugly”) changes as we vary the seed set size; while there is some change, it is not drastic.



(a) P@80 (b) R@80 (c) F@80

Fig. 3. Impact of number of seed words on three evaluation metrics over five datasets.

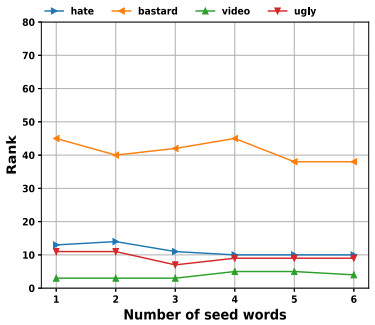


Fig. 4. Impact of number of seed words on the ranking of example words.

4.4.2 Impact of Embeddings on Evaluation Results. In DSKE, one way of incorporating the local context of words is using their distributional representation in the form of word embeddings. As noted before, we trained the word embeddings from scratch. We also evaluate the efficacy of pre-trained and locally-trained embeddings. Since GloVe [50] has pre-trained embeddings from Twitter, we compare our locally-trained embeddings with the GloVe embeddings of 100 dimensions on the Twitter dataset.[11] We did not use pre-trained word2vec embedding because unlike GloVe it does not have any pre-trained embeddings on Twitter corpus. Table 13 shows the comparative evaluation results. The last column of the table represents the loss/gain for GloVe; minus sign (-) represents loss and (+) represents a gain in performance. We can observe that Glove embeddings perform worse than our learned embeddings on three out of the five datasets; the performance is lower on both precision and recall, and consequently also on F-score. In particular, the performance degradation on D_{kh} and D_{ka} can be attributed to the use of regional words and their contexts in these two datasets. Although the words used in these two datasets are generic, in South Asia following the Pulwama attack in 2019, which lead to high tension between India and Pakistan, a number of these words acquired regional associations and contexts. For example, "jihad" in the global context is generally used to address terrorism, however, in South Asia in the current scenario, it is particularly used to radicalize youth on the Kashmir issue. Figures 5(a) and 5(b) show the top-10 words most similar to "jihad" from trained and pre-trained embeddings, respectively. We observe that "jihad" in our dataset is close to various regional words like "Pakistan", "Kashmir",

[11] https://nlp.stanford.edu/projects/glove/.

Table 13. Performance Evaluation Results Over the Five Datasets: Trained vs. Pre-Trained Embeddings

Dataset	Trained embeddings			GloVe embeddings			Loss/Gain in F@80 for Glove
	P@80	R@80	F@80	P@80	R@80	F@80	
D_h	0.338	0.375	0.355	0.338	0.375	0.355	0.0
D_s	0.725	0.453	0.558	0.725	0.453	0.558	0.0
D_a	0.488	0.398	0.438	0.463	0.374	0.412	**−0.026**
D_{kh}	0.425	0.756	0.544	0.407	0.725	0.518	**−0.031**
D_{ka}	0.513	0.488	0.499	0.473	0.468	0.469	**−0.030**

The bold entries in these tables represent the best performance evaluation result values among the various methods analyzed therein.

Table 14. Ablation Results for Different Components of DSKE Over Five Datasets

Approach	Datasets														
	D_h			D_s			D_a			D_{kh}			D_{ka}		
	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80	P@80	R@80	F@80
DSKE	0.338	0.375	0.355	0.725	0.453	0.558	0.488	0.398	0.438	0.425	0.756	0.544	0.513	0.488	0.499
DSKE-$\mathcal{S}(w)$	0.263	0.292	0.276	0.663	0.414	0.509	0.413	0.333	0.369	0.350	0.622	0.448	0.413	0.393	0.402
DSKE-$\mathcal{P}(w)$	0.30	0.333	0.316	0.713	0.445	0.548	0.450	0.364	0.402	0.413	0.733	0.528	0.450	0.429	0.439
DSKE-$\mathcal{D}(w)$	0.325	0.361	0.349	0.725	0.453	0.558	0.450	0.364	0.402	0.375	0.667	0.480	0.463	0.440	0.451

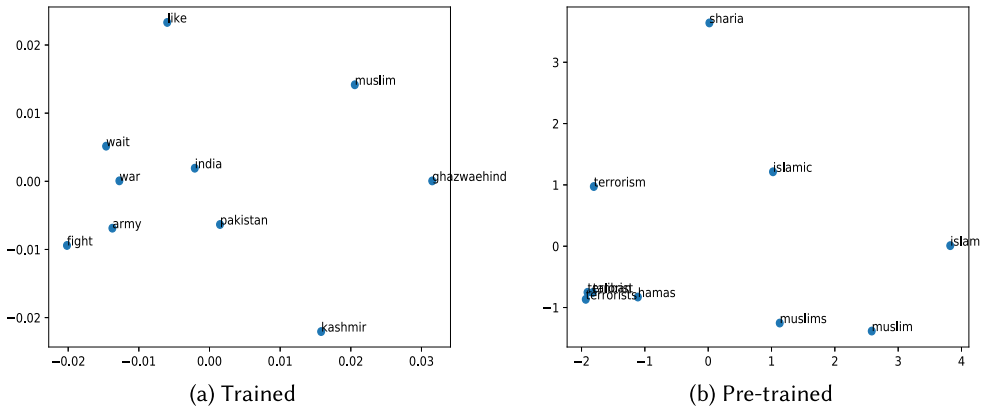


Fig. 5. Scatter plot of the PCA projection of top-10 most similar words to "jihad" using trained and pre-trained embeddings.

and "ghazwaehind", whereas in the global context it is more similar to "terrorism", "sharia", and "islamic".

4.4.3 Ablation Study of DSKE Model Components. DSKE incorporates both local and global contextual semantics using three components as discussed in Sections 3.2.1 and 3.2.2. We evaluate the contribution of each of the three components through *component ablation analysis*, in which a component is excluded from the vertex score assignment to observe its impact on lexicon generation. Table 14 shows the impact of each of the three components on lexicon generation. We use -$\mathcal{S}(w)$ to denote the case when we do not use embedding-based similarity. Similarly, -$\mathcal{P}(w)$ and -$\mathcal{D}(w)$ denote the exclusion of contextual proximity and domain relevance based similarity, respectively. We can observe that embedding-based semantic similarity has the highest impact. Contextual proximity has the next most impact, except on D_{kh}, where contrasting-corpora based

Table 15. Top-30 Keywords on D_h Using General Purpose and Domain-Specific Keyword Extraction Approaches

Hate (D_h)	
General purpose	Domain specific
new, muslim, msnbc, mad, place, **bastard**, thing, **bitch**, good, world, islam, face, wrong, **fuck**, **evil**, one, today, guy, way, trump, america, **retard**, **stupid**, gay, russian, way, people, **hell**, friend, job	**idiot**, **nigga**, **bitch**, people, **hell**, **ass**, **bad**, trump, **disgusting**, **stupid**, **hate**, **ugly**, **fuck**, **sick**, man, **retard**, woman, tired, country, **evil**, time, syria, day, girl, **isis**, potus, **bloody**, **attack**, crazy, **damn**

similarity has a high impact. Nevertheless, all three are important to obtain the best results for the full approach that combines all three components. These results clearly indicate that word embeddings along with a small set of seed words can be used as an effective mechanism to bias the ranking of domain-specific lexical units.

4.4.4 Domain-Specific vs. General Purpose Keyword Extraction. Domain-specific keyword extraction is closely related to keyword extraction. In keyword/keyphrase extraction, relevant and important keywords are extracted from a corpus without domain coherence, whereas domain-specific extraction identifies relevant words from a corpus that conceptualize a specific domain of interest. In DSKE, we bias the ranking of candidate words based on their similarity with the initial lexicon of seed words. However, DSKE can easily be converted to a general purpose keyword extraction approach by simply removing the seed word component, by setting $\mathcal{L} = \phi$ and assigning all the candidate words an initial weight of 1, which makes DSKE equivalent to TextRank. For example, Table 15 shows the list of top-30 keywords identified by the general purpose and domain-specific DSKE over D_h. The general purpose approach extracts keywords like *new*, *muslim*, and *bastard*, which are ranked lower in domain-specific DSKE. At the same time, a number of high ranked domain-specific keywords like *stupid*, *people*, and *hell* are ranked lower by the general purpose approach. On analysis, we found that out of top-30 keywords extracted by both general-purpose approach and DSKE only 8 are in common. Furthermore, the analysis of keywords of the first column of Table 15 shows that though a few of them are hate-related keywords, overall the extracted set of keywords do not clearly adhere to a particular domain of interest.

4.4.5 Statistical Significance Analysis. We also performed statistical significance analysis to answer the question: Is the difference between the ranking scores of keywords and normal words statistically significant? We performed an independent sample t-test on the five given datasets because keyword and normal words are independent sample sets. In independent sample t-test, null and alternative hypothesis are expressed as follows:

$$\begin{aligned} H_0{:}\mu &= \mu_0 \quad (thetwopopulationmeansareequal), \\ H_1{:}\mu &\neq \mu_0 \quad (thetwopopulationmeansarenotequal), \end{aligned} \tag{16}$$

where H_0 represents the null hypothesis with the assumption that there is no significant difference in the population means of ranking scores of keywords and words, whereas the alternative hypothesis H_1 assumes that the means of ranking score of keywords and words differ. We performed significance analysis at 5% level of significance and computed the value of p for each dataset which is given in Table 16. On analysis, we found that the computed value of p is less than 0.05 for each dataset, except D_s. Therefore, the null hypothesis is rejected over the four datasets, and it can be concluded that the mean of the ranking score for keywords and words over these four datasets differ significantly. However, in the case of D_s, we do not have sufficient evidence to conclude that the difference in ranking score between keywords and words is statistically significant, and we fail to reject the null hypothesis.

Table 16. p-value Over the Five Datasets

Dataset	p-value	$H_0 : \mu = \mu_0$
D_h	0.0015	Rejected
D_s	0.09	Fails to reject
D_a	0.014	Rejected
D_{kh}	0.015	Rejected
D_{ka}	0.002	Rejected

Table 17. Language-Agnostic Performance Evaluation Results Over the Five Datasets

Dataset	LA-DSKE			DSKE			Loss/Gain in F@80 for LA-DSKE	CNW [8]			RAKE [53]		
	P@80	R@80	F@80	P@80	R@80	F@80		P@80	R@80	F@80	P@80	R@80	F@80
D_h	0.275	0.305	0.289	0.338	0.375	0.355	−0.066	0.30	0.333	**0.316**	0.038	0.042	0.040
D_s	0.588	0.367	**0.452**	0.725	0.453	0.558	−0.106	0.575	0.359	0.442	0.00	0.00	0.00
D_a	0.350	0.286	**0.315**	0.488	0.398	0.438	−0.123	0.325	0.265	0.292	0.063	0.058	0.059
D_{kh}	0.363	0.644	**0.464**	0.425	0.756	0.544	−0.080	0.338	0.60	0.432	0.00	0.00	0.00
D_{ka}	0.363	0.345	**0.354**	0.513	0.488	0.499	−0.145	0.350	0.333	0.341	0.225	0.214	0.219

Bold results show the best results among language-agnostic approaches.

4.5 Language-Agnostic Results for LA-DSKE

We now present the evaluation results and comparative study of LA-DSKE, the language-agnostic variant of DSKE. We use the same set of five datasets—D_h, D_s, D_a, D_{kh}, and D_{ka}—and follow the approach discussed in Section 3.5 to find the list of potential stopwords. In the stopword identification process, we use 1,000 tweets from each dataset (except D_{ka}, which has 560 tweets). Furthermore, tweets are tokenized and all the words excluding the identified stopwords are selected as the candidate words. Thereafter, we apply the same approach as discussed in Sections 3.2 and 3.3 for keyword extraction using an initial set of 3 seed words. Table 17 presents the performance evaluation results of LA-DSKE over the five datasets. In this table, the values in the eighth column (Loos/Gain in F@80 for LA-DSKE) shows considerable gap in the performance of LA-DSKE in comparison to the language-dependent version, i.e., DSKE, over D_{ka} and D_a. However, it performs considerably better for the remaining three datasets. Considering the language-agnostic feature, the performance of LA-DSKE is acceptable, and it can be a reasonably good solution for languages lacking in terms of parsing tools.

Nevertheless, we need to compare LA-DSKE with other language-agnostic approaches to judge its effectiveness. Therefore, we compare LA-DSKE with CNW [8] and RAKE [53] that are language-agnostic. The last six columns of Table 17 show the results. It can be observed that LA-DSKE outperforms both approaches (except in one case of D_h where CNW has the best performance). The last three columns of this table show that RAKE performs poorly in comparison to LA-DSKE.

5 CONCLUSION AND FUTURE WORK

In this article, we presented a graph-theoretic hybrid approach called DSKE, utilizing the strength of distributional word representations and contrasting-domain corpus for domain-specific keyword extraction from noisy social media text. Unlike existing approaches, the novelty of DSKE lies in using the different measures of contextual semantics-based similarity between an initial set of seed words and corpus words to compute the semantic association of the corpus words with the target domain. DSKE is evaluated over both formal and informal text constituting six different datasets and performs significantly better in comparison to baseline and state-of-the-art approaches. We have also generated a lexicon of radical words used by the sympathizers of the

Khalistan and *Kashmir* movements. Furthermore, we extended DSKE as a language-agnostic approach called LA-DSKE and show that it too performs better than state-of-the-art approaches. At present, we are working to extend the lexicon of radical terms using DSKE, and to release that for the research community. Furthermore, we aim at utilizing the generated lexicon to develop an extremist and hate tweets classification system.

REFERENCES

[1] Muhammad Abulaish, Sielvie Sharma, and Mohd Fazil. 2019. A multi-attributed graph-based approach for text data modeling and event detection in Twitter. In *Proceedings of the 11th International Conference on Communication Systems & Networks*. IEEE Computer Society, 703–708.

[2] Muhammad Abulaish, Md. Imran Hossain Showrov, and Mohd Fazil. 2018. A layered approach for summarization and context learning from microblogging data. In *Proceedings of the 20th International Conference on Information Integration and Web-Based Applications & Services*. ACM, 70–78.

[3] Izzat Alsmadi and Ikdam Alhami. 2015. Clustering and classification of email contents. *Journal of King Saud University–Computer and Information Sciences* 27, 1 (2015), 46–57.

[4] Nikita Astrakhantsev. 2017. ATR4S: Toolkit with state-of-the-art automatic terms recognition methods in Scala. *Language Resources and Evaluation* 52, 3 (2017), 853–872.

[5] Marco Basaldella, Elisa Antolli, Giuseppe Serra, and Carlo Tasso. 2018. Bidirectional LSTM recurrent neural network for keyphrase extraction. In *Proceedings of the Italian Research Conference on Digital Libraries*. Springer, Cham, 180–187.

[6] Abdelghani Bellaachia and Mohammed Al-Dhelaan. 2012. NE-Rank: A novel graph-based keyphrase extraction in Twitter. In *Proceedings of the International Conferences on Web Intelligence and Intelligent Agent Technology*. IEEE Computer Society, 372–379.

[7] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research* 3, 1 (2003), 1137–1155.

[8] Saroj Kr. Biswas, Monali Bordoloi, and Jacob Shreya. 2018. A graph based keyword extraction model using collective node weight. *Expert Systems with Applications* 97, 5 (2018), 51–59.

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3, 1 (2003), 993–1022.

[10] Abraham Bookstein and Don R. Swanson. 1974. Probabilistic models for automatic indexing. *Journal of the American Society for Information Science* 25, 5 (1974), 312–318.

[11] Georgeta Bordea, Paul Buitelaar, and Tamara Polajnar. 2013. Domain-independent term extraction through domain modelling. In *Proceedings of the 10th International Conference on Terminology and Artificial Intelligence*. ICSA, 1–8.

[12] Christopher Brewster, Jose Iria, Ziqi Zhang, Fabio Ciravegna, Louise Guthrie, and Yorick Wilks. 2007. Dynamic iterative ontology learning. In *Proceedings of the 6th International Conference on Recent Advances in Natural Language Processing*. ACL, 1–5.

[13] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems* 30, 11 (1998), 107–117.

[14] Pete Burnap and Matthew L. Williams. 2016. Us and them: Identifying cyber hate onTwitter across multiple protectedcharacteristics. *EPJ Data Science* 5, 11 (2016), 1–15.

[15] Teresa M. Chung. 2003. A corpus comparison approach for terminology extraction. *Terminology* 9, 2 (2003), 221–246.

[16] Kenneth Church, William Gale, Patrick Hanks, and Donald Hindle. 1991. Using statistics in lexical analysis. *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon* 1, 1 (1991), 115–164.

[17] Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics* 16, 1 (1990), 22–29.

[18] W. B. Croft and D. J. Harper. 1979. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation* 35, 4 (1979), 285–295.

[19] Sally F. Dennis. 1964. The construction of a thesaurus automatically from a sample of text. In *Proceedings of the Symposium on Statistical Association Methods For Mechanized Documentation*. ACL, 61–148.

[20] Swagata Duari and Vasudha Bhatnagar. 2018. sCAKE: Semantic connectivity aware keyword extraction. *Information Sciences* 477, 1 (2018), 100–117.

[21] Gonenc Ercan and Ilyas Cicekli. 2007. Using lexical chains for keyword extraction. *Information Processing and Management* 43, 3 (2007), 1705–1714.

[22] Andrea Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A publicly available lexical resourcefor opinion mining. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*. European Language Resources Association, 417–422.

[23] Corina Florescu and Cornelia Caragea. 2017. A position-biased PageRank algorithm for keyphrase extraction. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. AAAI, 4923–4924.
[24] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. ACL, 1105–1115.
[25] Corina Florescu and Wei Jin. 2019. A supervised keyphrase extraction system based on graph representation learning. In *Proceedings of the European Conference on Information Retrieval*. Springer, Cham, 197–212.
[26] Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of Twitter abusive behavior. In *Proceedings of the 12th International Conference on Web and Social Media*. Association for the Advancement of Artificial Intelligence, 491–500.
[27] Zelalem Gero and Joyce C. Ho. 2019. NamedKeys: Unsupervised keyphrase extraction for biomedical documents. In *Proceedings of the 10th International Conference on Bioinformatics, Computational Biology and Health Informatics*. ACM, 328–337.
[28] Sujatha D. Gollapalli, Xiao-Li Li, and Peng Yang. 2017. Incorporating expert knowledge into keyphrase extraction. In *Proceedings of the 31st International Conference on Artificial Intelligence*. AAAI, 3180–3187.
[29] Stephen P. Harter. 1975. A probabilistic approach to automatic keywords indexing. *Journal of the American Society for Information Science* 26, 5 (1975), 197–206.
[30] Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. ACL, 1262–1273.
[31] Samer Hassan, Rada Mihalcea, and Carmen Banea. 2007. Random-walk term weighting for improved text classification. In *Proceedings of the 1st International Conference on Semantic Computing*. IEEE Computer Society, 242–249.
[32] J. P. Herreraa and P. A. Pury. 2008. Statistical keyword detection in literary corpora. *The European Physical Journal B* 63, 1 (2008), 135–146.
[33] Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 216–223.
[34] Kyo Kageura and Bin Umino. 1996. Methods of automatic term recognition: A review. *Terminology* 3, 2 (1996), 259–289.
[35] karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 28, 1 (1972), 11–21.
[36] Su N. Kim, Timothy Baldwin, and Min Y. Kan. 2009. Extracting domain-specific words - a statistical approach. In *Proceedings of the Australasian Language Technology Association Workshop*. ACL, 94–98.
[37] Chunyu Kit and Xiaoyue Liu. 2008. Measuring mono-word termhood by rank difference via corpus comparison. *Terminology* 14, 2 (2008), 204–229.
[38] Ugur Kursuncu, Manas Gaur, Usha Lokala, Krishna prasad Thirunarayan, Amit Sheth, and Budak Arpinar. 2018. *Predictive Analysis on Twitter: Techniques and Applications*. Springer, Cham, Chapter Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining, 67–104.
[39] Marina Litvak, Mark Last, Hen Aizenman, and Inbal Gobits. 2011. DegExt: A language-independent graph-based keyphrase extractor. In *Proceedings of the International Conference on Advances in Intelligent Web Mastering*. Springer,, 121–130.
[40] Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. ACL, 634–639.
[41] Yutaka Matsuo and Mitsuru Ishizuka. 2003. Keyword extraction from a single documentusing word co-occurrence statistical information. In *Proceedings of the 16th International Florida Artificial Intelligence Research Society Conference*. AAAI, 392–396.
[42] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of 55th the Annual Meeting of the Association for Computational Linguistics*. ACL, 582–592.
[43] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the International Conferences Empirical Methods in Natural Language Processing*. ACL, 404–411.
[44] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations inVector space. *Computing Research Repository (CoRR)*. 1–12. https://arxiv.org/abs/1301.3781.
[45] George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM* 38, 11 (1995), 39–41.
[46] Saif Mohammad and Peter Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence* 29, 3 (2013), 436–465.
[47] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report. Stanford InfoLab, 1–17.

[48] Eirini Papagiannopoulou and Grigorios Tsoumakas. 2018. Local word vectors guiding keyphrase extraction. *Information Processing and Management* 54, 6 (2018), 888–902.
[49] Youngja Park, Siddharth Patwardhan, Karthik Visweswariah, and Stephen C. Gates. 2008. An empirical analysis of word error rate and keyword error rate. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association*. ICSA, 270–273.
[50] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.
[51] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2010. Opinion word expansion and target extraction through double propagation. *Computational Linguistics* 37, 1 (2010), 9–27. DOI: https://doi.org/10.1162/coli_a_00034
[52] S. E. Robertson and S. Walker. 1997. On relevance weights with little relevance information. In *Proceedings of the 12th International Conference on Research and Development in Information Retrieval*. ACM, 16–24.
[53] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extractionfrom Individual Documents*. John Wiley & Sons.
[54] Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. ACL, 1668–1678.
[55] Geetika Sarna and M. P. S. Bhatia. 2016. A probalistic approach to automatically extract new words from social media. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*. IEEE Computer Society, 719–725.
[56] Rushdi Shams and Robert E. Mercer. 2013. Classifying spam emails using text and readability features. In *Proceedings of the 13th International Conference on Data Mining*. IEEE Compouter Society, Washington DC, USA, 657–666.
[57] Jacopo Staiano and Rada Marco Guerini. 2014. DepecheMood: A lexicon for emotion analysis from crowd-annotated news. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. ACL, 427–433.
[58] Lucas Sterckx, Cornelia Caragea, Thomas Demeester, and Chris Develder. 2016. Supervised keyphrase extraction as positive unlabeled learning. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing*. ACL, 1924–1929.
[59] Carlo Strapparava and Alessandro Valitutti. 2004. WordNet-Affect: An affective extension of WordNet. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*. European Language Resources Association, 1083–1086.
[60] Kabir Taneja and Kriti M. Shah. 2019. The conflict in Jammu and Kashmir and the convergence of technology and terrorism. *Royal United Services Institute for Defence and Security Studies*, Paper No. 11 (2019), 1–14.
[61] Ming-Feng Tsai, Chuan-Ju Wang, and Pu Chuan Chien. 2016. Discovering finance keywords via continuous-space language models. *ACM Transactions on Management Information Systems* 7, 3 (2016), 1–17.
[62] Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd International Conference on Artificial Intelligence*. AAAI, 855–860.
[63] Rui Wang, Wei Liu, and Chris McDonald. 2015. Corpus-independent generic keyphrase extraction using word embedding vectors. In *Proceedings of the Software Engineering Research Conference*. 1–8.
[64] Ian H. Witten, Alistair Moffat, and Timothy C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann.
[65] Yuxiang Zhang, Huan Liu, Suge Wang, W. H. Ip., Wei Fan, and Chunjing Xiao. 2019. Automatic keyphrase extraction using word embeddings. *Soft Computing* 23, 5 (2019), 1–16.
[66] Yu Zhang, Mingxiang Tuo, Qingyu Yin, Le Qi, Xuxiang Wang, and Ting Liu. 2020. Keywords extraction with deep neural network model. *Neurocomputing* 383, 1 (2020), 113–121.
[67] Yong Zhang and Weidong Xiao. 2018. Keyphrase generation based on deep Seq2seq model. *IEEE Access* 6, 9 (2018), 46047–46057.
[68] Ziqi Zhang, Ziqi Zhang, and Ziqi Zhang. 2018. SemRe-Rank: Improving automatic term extraction by incorporating semantic relatedness with personalised PageRank. *ACM Transactions on Knowledge Discovery from Data* 12, 5 (2018), 1–41.
[69] Hongding Zhou and Gary W. Slater. 2003. A metric to search for relevant words. *Physica A: Statistical Mechanics and its Applications* 329, 1 (2003), 309–327.

Received March 2021; revised August 2021; accepted October 2021