IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 28 February 2023, accepted 30 March 2023, date of publication 5 April 2023, date of current version 12 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3264678

RESEARCH ARTICLE

Advanced Hierarchical Topic Labeling for Short Text

PARAS TIWARI[1], ASHUTOSH TRIPATHI[2], AVANEESH SINGH[3], AND SAWAN RAI[2]
[1]Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, Varanasi, Uttar Pradesh 221005, India
[2]Department of Computer Science and Engineering, PDPM Indian Institute of Information Technology Design and Manufacturing, Jabalpur, Madhya Pradesh 482005, India
[3]Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, Kanpur, Uttar Pradesh 208016, India

Corresponding author: Avaneesh Singh (avaneeshs@iitk.ac.in)

ABSTRACT Hierarchical Topic Modeling is the probabilistic approach for discovering latent topics distributed hierarchically among the documents. The distributed topics are represented with the respective topic terms. An unambiguous conclusion from the topic term distribution is a challenge for readers. The hierarchical topic labeling eases the challenge by facilitating an individual, appropriate label for each topic at every level. In this work, we propose a BERT-embedding inspired methodology for labeling hierarchical topics in short text corpora. The short texts have gained significant popularity on multiple platforms in diverse domains. The limited information available in the short text makes it difficult to deal with. In our work, we have used three diverse short text datasets that include both structured and unstructured instances. Such diversity ensures the broad application scope of this work. Considering the relevancy factor of the labels, the proposed methodology has been compared against both automatic and human annotators. Our proposed methodology outperformed the benchmark with an average score of 0.4185, 49.50, and 49.16 for cosine similarity, exact match, and partial match, respectively.

INDEX TERMS Document categorization, hierarchical topic modeling, hierarchical topic labeling, topic modeling, topic labeling.

I. INTRODUCTION

The popularity of digital usage has led to enormous content generation. A vast corpus of content is accessible in just a few clicks. This is a curse in disguise. Users need refined and relevant information rather than just a pile of content. The relevancy could be related to a specific query, interest, or topic. Topic Modeling (TM) helps to cluster the relevant content from the compilation of content. The applications of TM are not limited to the computer science domain. The TM primarily uses the probabilistic approach [1], which makes it suitable for a broad range of applications. Authors in [2] used the aspect-term-based TL for sentiment analysis of mobile phone reviews available on e-commerce platforms. The authors in [3] used the TM technique to classify the apps. It has also been utilised in various other domains like digital humanities [4], bioinformatics [5], social and cultural studies [6], and so on. Such broad contributions have enhanced the significance of TM.

TM is very suitable for experts in the domain to understand the essence of a text-based corpus. However, the limited information representation in TM is insufficient to satisfy a larger readership. Usually, it is difficult to get the exact impression of the documents by considering only the topic terms. Topic Labeling (TL) overcomes this limitation of TM. TL resolves such issues by labeling the topic and associating the relevancy of topic terms with the topic corpus. TL assigns the appropriate label to the topic rather than any collection of topic terms. Authors in [7] proposed the first dedicated methodology for labeling the topics. TL has also proven to be extremely useful in various domains [8]. Similar to TM, TL also contributes to a broad range of applications. Various interesting techniques have been proposed for both TM and TL.

The associate editor coordinating the review of this manuscript and approving it for publication was Biju Issac.

The topics in TM are not interrelated. Every topic in the TM does not utilise the information available in the other topics. In a large-size corpus, the document may differ at the lower level but may relate at a higher. For example, in a large corpus that contains abstracts of multi-domain articles, at a lower level, the documents about 'sentiment analysis' and 'handwritten digit recognition' may belong to different topics. However, both are related at a higher level, belonging to the topic of 'machine learning.' Similarly, at a higher level, articles belonging to 'machine learning' and 'artificial intelligence' belong to the same topic, 'computer science.' Such related information gets missed in the TM.

TM struggles to limit the frequency of topics while maintaining topic coherence [9]. The problem is more critical when users need niche information but are uncertain about the keywords required to fetch the related, relevant documents. Users hit and try various keywords in their search query to extract the relevant content. Hierarchical Topic Modeling (HTM) and Hierarchical Topic Labeling (HTL) play a crucial role here. With the help of HTL, users can start with a broad topic and filter the documents for the specific sub-topic. It led to the availability of the relevant document based on the niche requirements. It also facilitates recommending the most similar documents that meet the niche requirement as sibling topics. Hierarchical information extraction has been used in a broad range of applications. Authors in [10] proposed a three-step process, i.e., feature extraction, feature clustering, and app clustering, for information extraction. The authors extracted features using greedy hierarchical clustering techniques over n-grams, clustered features using spherical k-means, and clustered apps using agglomerative hierarchical clustering technique. Authors [10] opted for hierarchical clustering to observe information at the granular level. Similarly, in this article, we have also implemented HTL on the article's abstract dataset. The proposed work would assist in retrieving articles ranging from broadly related to the problem statement to specifically related to the problem statement.

However, only a few contributions have been made so far for HTL. HTL gives better insight into the large dataset. Apart from the data volume, even the size of the data instance is an essential factor. The number of short-length text instances has significantly increased [11]. Such text-instance trends have enhanced the importance of contributions for short text data. Generally, in topic modeling domain short text are considered as text instances belonging to the social media platforms such as, Twitter®, Facebook®, and Youtube® comments [64]. Twitter has 280 characters limit. In our work, the average size of token per instance is 127. In our work, even among the short text, we have included both structure and unstructured text instances. Hence, we have contributed HTL for short text instances in this work.

The rest of the paper is as follows: Section II discusses various contributions made so far, dealing with the TL and HTL. The section III discusses various similarities and differences in various dimensions to ensure an unambiguous understanding of HTL. Section IV discusses the proposed architecture at the granular level, its strengths and limitations, and the evaluation strategies with their relevancy to the problem statement. The section V discusses the various sets of experiments that were performed to optimise performance. Section VI discusses the performance comparison with state-of-the-art and the proposed methodology. Section VII concludes the need for HTL, performance factors, and future work.

II. RELATED WORK

In this section, we discuss various contributions to HTL. Since the HTL and TL share some features, we have also discussed a few significant contributions of the TL that have relevance to the HTL.

In the limited contributions for HTL, authors in [12] assume all the entities as sets of concepts in the ontology [13]. For creating the hierarchy of the topics, authors in [13] used multiple similarity measures like cosine, overlap, mutual, dice, Tanimoto, and Jaccard similarities, along with an English thesaurus for TL. Inspired by the term-based similarity authors in [14], the term weight-based statistical ranking method was proposed. The authors generated candidate labels using Ngram Testing and ranked them considering the importance of each term via term weighting schemes and representation via a probability distribution.

As discussed about the impact of contributions to TL over HTL, following are the major approaches that influenced the HTL:

A. STAT-BASED APPROCHES

As with HTL, the majority of TL strategies also consider the frequency of terms to be one of the essential features. Authors in [15] weighted the topic terms based on the averaged pointwise mutual information and conditional probabilities, considering the highest-scoring topic term as the topic label. Authors in [16] generate embedding matrix by doc2vec [17] and word2vec [18] to calculate the similarity between the Wikipedia titles and words in the topics. The topic candidates were further re-ranked, using Support Vector Regression (SVD) [19] of four features, i.e., LetterTrigrams [20], PageRank [21], NumWords and TopicOverlap [22].

Apart from the term frequency, various other aspects have also contributed remarkably to TL. Very interestingly, authors in [23] considered TL as a word-sense disambiguation problem, using Eigenvalue-based measures to choose the most sensible word among different target words. Authors in [20] extracted a summary of documents belonging to a topic as a candidate label and the most similar candidate label vector to the topic vector as a topic label. The authors in [24] used sentiment-based and aspect-based cluster terms to label the tweets related to the COVID-19 pandemic. The contextual information, however, is missed by term-based strategies. Contextual information plays a significant role in various applications. Graphs have been considered an important tool

to utilise contextual information. Hence, graphs have also played important roles in various applications.

B. GRAPH-BASED APPROACHES

Graphs are considered an elementary representation of entities and the relationships among them. Multiple applications utilise the relative information to get the desired outcome. Authors in [25] created a graph-based ranking method for labeling the topic from the terms belonging to the topic and Kullback-Leibler Divergence to select the relevant candidate sentences. The authors created a directed weighted graph using relevance centrality, coverage centrality, and discrimination centrality. Authors in [26] proposed TF-IDF (Term Frequency-Inverse Document Frequency) and BM25 [27] based information extraction methods along with a ranking method inspired by [28] over LDA and NMF topic modeling. Authors in [29] proposed an ensemble learning [30] based methodology over truth discovery algorithm [31] that consists of a graph with topics, words, Wikipedia articles, and candidate terms as nodes, and relationship among them as edge. Authors in [32] applied graph extraction methods over DBpedia[1] concepts. The graphs generally require large computations to get the output. The excessive computational requirement for graph-based strategies limits their scope of applications.

C. ONTOLOGY-BASED APPROACHES

Ontologies [33] has also significantly impacted TL. The authors in [34] considered ontology mapping as a multilabel classification where each concept represents a classification class for multi-lingual TL. Authors in [35] proposed OntoLDA, an ontology-based methodology for both TM and TL. In [36], the authors proposed the KB-LDA, an ontology-based knowledge-based topic model that labels topics based on concept semantics. Authors in [37] proposed another Onto_TML strategy to label the topic by considering the ancestors of the top topic words. Authors in [38] proposed graph-based analysis inspired by social media network graphs over LDA topics to get the most influential topics and labeling using ontology.

D. NEURAL NETWORK-BASED APPROACHES

It is trivial to mention the dominance of neural network-based architectures in various applications. For HTL, authors in [39] used hLDA [40], [41] for generating HTM and neural embedding [16] for the labeling. Authors in [42] used basic encoder-decoder architecture with GRU-attention for the labeling. Authors in [43] proposed TL with a paired-attention-based deep neural network. Bi-directional Encoder Representation of Transformers (BERT) has been widely utilised as a base framework for various applications. Authors in [44] proposed a BERT-based TM strategy, where they implemented k-means on the sentence BERT embedding matrix to generate topic modeling. The centroid of each cluster has been considered the topic label. Authors in [45] proposed a BERT-based BART-tl architecture to generate topic labels.

[1] https://www.dbpedia.org/

III. PRELIMINARIES

The HTL is closely related to TM, TL, and HTM but differentiates at certain levels. The discussion on a granular level would give a detailed insight into the problem statement. Since the problem statement deals with a diverse set of features, we have considered the following to maintain uniformity or standard:

- *Term* as individual tokens in the Document (D).
- *Document* as set of sentences and equations.
- *Corpus* as set of documents, i.e., $\mathrm{C} = \{D_1, D_2, D_3, \ldots, D_M\}$ assuming there are total M documents (D_i).

A. TOPIC MODELING

The fundamentals of TM have been introduced as the indexing of documents using latent semantic analysis [1]. It inspired to consider document-term association using the respective semantic matrix. Following this work, the author in [46] proposed Probabilistic Latent Semantic Indexing (PLSI). The author replaced singular value decomposition with a generative model in PLSI to get the joint probabilities of words with documents. It gave insight into a strong document-word relationship with an inspiring probabilistic methodology. Such probabilistic strategy inspired authors, in [47], to classify the unlabeled documents using Dirichlet distribution as the Dirichlet Multinomial Mixture (DMM). The authors did not formally mention it as a topic model. Taking into account the strengths of both PLSI [46] and DMM [47], the authors of [48] proposed latent Dirichlet allocation (LDA) with a formal introduction of the ''topic model.'' The performance strength of LDA lies in the strategy where, rather than distributing one topic per document as in PLSI, it distributes collections of topics to each document. The goal of LDA is to maximise the likelihood of documents over k topics and the likelihood of words for respective topics.

In general, TM is the phenomenon of clustering documents based on their similarity to a ''topic'' with a collection of words that are most likely to belong to the ''topic.''

1) TOPIC TERMS

Topic terms are the tokens in the documents that belong to the respective topic and have the highest probability of belonging to that topic. The number of topic terms is variable and depends on the technique of TM. A score is also attributed to each topic term for being in the respective topic. The score is usually the probability that is considered as the likelihood of the term belonging to the respective topic. The topic term is the layer that clearly distinguishes the TM from document clustering and other similar domains. It is also one of the initial steps that help the users to get a brief insight into the documents belonging to the respective topic.

B. TOPIC LABELING

The phenomenon of assigning an individual word that deduces about the documents under the respective topic is known as topic labeling. The word could be a single-unit or multi-unit term. As per the definition proposed by authors in [7] for topic labeling, the label should be semantically meaningful and must cover the latent meaning of the topic.

1) CANDIDATE TERMS

The terms that have the potential for being the label of a topic are considered candidate terms. The candidate terms are dependent upon the methodology used for labeling the topic. Multiple strategies have been used to extract the closest possible term, ''candidate.'' Authors in [44] created clusters and assumed the centroid of clusters as closest to all the entities in the cluster, so they considered the centroid of the proposed cluster as the candidate. The authors in [26] used an information extraction method based on BM25 and TF-IDF [27] to extract the candidate terms. Authors in [25] rely on graph-based methods for ranking the candidate terms. Authors in [29] consider the categories of extracted Wikipedia documents as candidates. Authors in [7] extracted candidate terms using n-gram testing and chunking parsing. Similarly, authors in [22] use a chunk parser for candidate generation.

C. HIERARCHICAL TOPIC MODELING

Hierarchical Topic Modeling is a process of arranging documents of a corpus to a hierarchical structure in which topics are assigned to the documents and arranged so that the topic information gets more generalized as it traverses to the higher topic levels. $HTM = (T, D)$, where collection of documents D belonging to topic T, such that $T \in (T_t, TT_i)$ where TT_i is topic-terms belonging to the respective topic T_t. Each topic $T_t \in (T_{l-1}, T_l, T_{l+1})$ such that T_t incorporates precised features of parent topics in T_{l-1}, generalized features of child topics in T_{l+1}, and distinctive features from sibling topic $T_l^{\cdot}$. HTM also generally refers to the distribution of documents concerning hierarchically related topics. The topics are a probability distribution of words with the same intent. The topic relationship is an essential feature of HTM.

Authors in [41] consider nested tree structure-based probabilistic distributions for document-topic relations for HTM. Instead of a tree structure, the authors in [51] proposed a directed acyclic graph for HTM. Authors in [50] discuss the strengths and limitations of supervised and unsupervised HTM and propose a semi-supervised strategy over a base tree. The graph-based methods are computationally intensive. Considering the computational challenges, the authors in [52] replaced the probability distribution with hierarchical matrix factorisation. Authors in [49] regard the structural relationship of the documents as a hierarchy, i.e., the relation between the words, sentences, and documents. Authors in [53] consider word pairs in the hierarchy using the knowledge-based mining strategy. In our work, we have considered nested tree-based hierarchical topic tree generation.

D. HIERARCHICAL TOPIC LABELING

Hierarchical Topic Labeling is a process of labelling the topics arranged in a hierarchical manner, where the information gets precise with traversal from higher-level topics to lower levels. $HTL = (T, L)$ where topics T are arranged in a hierarchical manner, and L is the respective label of the topic at level l such that label L_l represents specialized information than L_{l-1} and generalized information than L_{l+1}. Since each document belongs to only one topic at the leaf level, it is similar to topic labeling. However, HTL must deal with extra features, including hierarchical relationships among documents and topics. HTL must possess the following properties:

- Relevancy to all the documents belonging to the leaf topic i.
- Relevancy to all the documents belonging to the child topics j.
- Maintain the discreteness among the topic i and j at same level k.
- At the topmost level, must consider the abstract label covering all the topics.

IV. METHODOLOGY

This section discusses the proposed methodology, the algorithm's inspiration, and the trade-off factors between strengths and limitations. Our methodology has been divided into four sequential components:

- Hierarchical Topic Tree Generation
- Candidate Generation
- Candidate Ranking
- Hierarchical Topic Labeling

A. HIERARCHICAL TOPIC TREE GENERATION

In Section III, we have discussed the similarities and differences between the TM and HTM. There have been limited contributions to HTM. The TM techniques dominantly consider the likelihood of topics in the document. However, HTM has the additional requirement of relationships among the topics. In our work, the hierarchy is maintained with the consideration of a widespread phenomenon, the Chinese Restaurant Process (CRP) [40]. This phenomenon is inspired by an assumption made by authors [54], [55] about Chinese Restaurants. The authors assume table distribution in the restaurant such that any time the M customers acquire N tables, however, when a new customer arrives, at least one unoccupied table is available for him. The probability distribution for a new customer, whether to join any occupied

table or any unoccupied table, is drawn as follows:

$$P(\text{occupied-table } i \mid \text{pre-arrived customers}) = \frac{m_i}{\gamma + m - 1}$$
$$P(\text{next occupied-table} \mid \text{pre-arrived customers}) = \frac{m_{i+1}}{\gamma + m - 1} \quad (1)$$

where m_i is the number of customers sitting in table i, and γ is the real valued parameter that consider the probability of occupying an unoccupied table.

A hierarchy can be considered as a nested sequence of partitions. As CRP is used to estimate the number of possible components considering the partition of integers over a single parameter distribution [41]. Similarly, nested Chinese restaurant process (nCRP) is used to estimate hierarchical components. The nCRP, is an unsupervised stochastic process utilizes to cluster the documents based on the topics at multiple levels of abstraction by assigning a probability distributions to an infinite-node trees. For nCRP, each customer has the option to choose a table from an infinite number of restaurants, with an infinite number of tables in each restaurant. The likelihood of a customer choosing the table in a restaurant is estimated using Equation 1. The first opted restaurant is considered as a node. Now for the relationship among the node (edge in the lower-level hierarchy), it is assumed that each table has a card that refers to another restaurant. To keep the structure intact, each restaurant must be referred to only once. The N restaurant traversal pattern of the customer starting with the root node creates a path to the $N^{th}-$level tree. Similarly, the collection of traversal patterns by M number of customers can be a subtree of an infinite tree.

With respect to the hierarchical topic tree generation, the topics are assumed to be essential mixture components. The document-specific mixture distribution has been estimated using the words in the document as follows:

$$P(word \mid \theta) = \Sigma_i^N \theta_i P(word \mid z = i, \beta_i) \quad (2)$$

where, N represents possible number of topics, z represents multinomial variable and β represents parameter for word distribution. The distribution will be random as θ is random in Equation 2. The topics are nodes in the tree (like restaurants), and the documents choose the path from the root to the leaf (like customers). The words are chosen from the documents associated with the path of the topic. The hierarchical topic tree generative process broadly follows the following steps:

1) For each table in the infinite tree,
   a) Draw topic (β_i) as probability distribution across words.
2) For each document in the corpus,
   a) Path for the doc using nCRP
   b) Draw distribution over each level
   c) For each word,
      i) Adopt level
      ii) Adopt relevant word parameterized using Equation 2.

B. CANDIDATE GENERATION

As mentioned in Section III, candidates are probable topic labels. The easiest brute-force method would be to consider all the terms in the corpus as candidates and rank them to get the label. This method might lead to the desired results, but it will consume a lot of computation and resources. The generated candidate should be closer to the requirements of the topic label.

We considered extracting the most important keywords related to the topic as candidates in our work. There have been multiple methods to extract the keywords. In this work, we have patronised unsupervised methods as a fundamental requirement. So, we followed an unsupervised method to extract keywords for candidate generation. As a fundamental step, we collected the documents belonging to the respective topics and created disjoint corpora at each level. It led to completely disjoint corpora at the leaf level. Above the leaf level, the corpus includes documents belonging to each sub-topic. Such collection maintained the hierarchical relationships among the corpora. For each corpus, we tokenise the terms, remove stopwords, and generate *Ngrams*. *Ngrams* as a lengthier slice of a sentence with a higher potential to infuse contextual information [56]. However, we have limited our work to the single unit labels, so we kept only unigrams. We removed stopwords, as we assumed stopwords are not suitable as topic labels. It also reduces the number of tokens, which saves computations. We also removed tokens that appear in fewer than ten documents in the corpus, as such tokens do not have the potential to be the label of a topic. We need semantic relations between tokens and the corpus. The word vector should represent the feature information of the entity. The pre-trained embedding facilitates providing richer entity information in the form of a vector. Pre-trained embeddings have been found to be useful in various applications [57]. We used pre-trained all-MiniLM-L6-v2 BERT embedding[2] for the documents in the corpus. The selected embedding has been trained on more than one billion training pairs. The diversity in the training pairs ensures the embedding quality.

To gather the relationship between the corpus and n-gram vectors, we calculate the semantic distance among the vectors using cosine similarity. To ensure the diversity in the candidate terms, we also re-ranked the n-grams using Maximal Marginal Relevance (MMR) [58]. From the re-ranked tokens of MMR, we extracted the top 100 tokens. These 100 tokens have been the most important keywords of the corpus of the respective topic. However, utilising the probabilistic topic-term output, we filtered the tokens that also belong to the topic-term set of the respective topic. It helps to filter the most relevant terms with a high potential to be the label of the topic, i.e. candidates. The candidate generation process is demonstrated in Algorithm 1.

[2]https://huggingface.co/sentence−transformers/all/−MiniLM−L6−v2

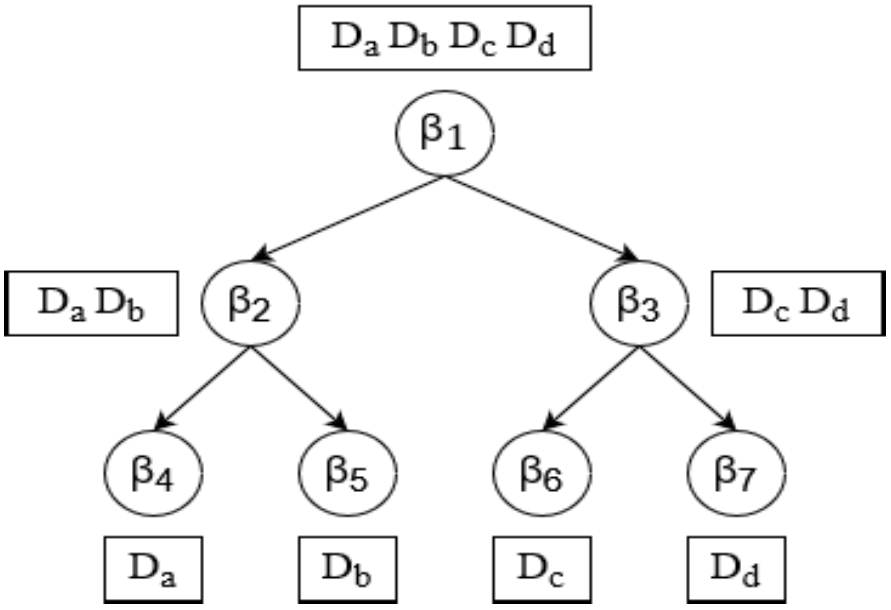

FIGURE 1. The paths of four documents through the infinite tree (as per the assumption in CRP, with three levels). The solid lines connect each topic to the topics referred (as tables referred to restaurants). The collected paths of the four documents describe a particular subtree of the underlying infinite tree.

We implemented the Algorithm 1 with the help of an open source tool.[3]

C. CANDIDATE RANKING

Each candidate term has the potential to be the label for the topic. However, we need an individual term as a label rather than a set of terms. To get the most relevant term as a label, we need to rank the candidate terms based on their potential. Authors in [43] used Kullback-Leibler Divergence [59] to rank the candidates. Authors in [26] used *C-Value* method inspired from [28] to rank the domain-specific candidate terms. Authors in [39] and [16] utilise neural embeddings for candidate ranking. Authors in [29] utilise truth discovery-based algorithms to rank the candidate terms. Authors in [22] sort the candidates using RACO (Related Article Conceptual Overlap) [60] measure over the mapping of word vectors and letter trigram vectors to the respective topic.

In our work, we need a label that possesses both latent and semantic properties of the documents belonging to the respective topic. We used 200-dimensional pre-trained Glove embeddings [61] to vectorise the topic-corpus and candidate label. The 200-dimensional pre-trained Glove used in our work had been trained on the 2B tweets, 27B tokens and have 1.2M vocabulary. The training on the short text suits for our work. We preferred 100-D over 200-D to incorporate larger contextual information. We rank the labels based on their cosine similarity score. To get the topic-corpus vector, we incorporated granular-level information using the following equations:

$$CorpusGloveVec = \frac{\Sigma_i^n DocGloveVec[i]}{n} \tag{3}$$

where, considering a topic carries n documents.

$$DocGloveVec = \frac{\Sigma_j^m SentGloveVec[j]}{m} \tag{4}$$

where, considering each document has m sentences.

$$SentGloveVec = \frac{\Sigma_k^o TermGloveVec[k]}{o} \tag{5}$$

where, considering each sentence carries o number of terms in it.

$$TermGloveVec = \frac{\Sigma_l^p WordGloveVec[lemmatize(l)]}{p} \tag{6}$$

for multi-unit term, we averaged out the *TermGloveVec* for p number of units in each term.

D. HIERARCHICAL TOPIC LABELING

The *candidate ranking* method gives us a set of terms sorted by their potential to be the label of the respective topic. Since, at each level of the hierarchy, we have maintained the discrete distribution of documents concerning the topic to which they belong. The leaf node topics get the labels generated with the discrete corpus of the respective topic. The corpora get merged as per the topic modeling of the documents. So, as per the requirement, at the leaf node, the labels are generated.

Such distribution meets the fundamental requirement of hierarchy. We opted for the term that possesses maximum

[3]https://maartengr.github.io/KeyBERT/api/keybert.html

Algorithm 1 Candidate Generation

Require: a Topic-Document set TD = $\{(T_i, D_i) \mid D_i$ is set of documents for each Topic $T_i\}$ and a Topic-Topic Terms set TT $=\{(T_i, TTrm_i) \mid TTrm_i$ is set of topic terms for each Topic $T_i\}$

Ensure: a Topic-Candidate set TC = $\{(T_i, TC_i) \mid TC_i$ is set of candidate terms for each Topic $T_i\}$

TC = ∅
for each $(T_i, D_i) \in$ TD **do** in parallel
  $corpus_i = \emptyset$
  for each $d_j \in D_i$ **do**
    $corpus_i.append(d_j)$
  end for
  TK_i = Tokenize($corpus_i$)
  $CleanTK_i$ = RemoveStopWords(TK_i) ▷ English language stopwords
  $ThresholdTK_i$ = RemoveMinFreqWords($CleanTK_i$)
  $Ngrams_i$ = NgramVectorizer($ThresholdTK_i$)
  $EmbedVectors_i$ = PreTrainedBERTembedding($corpus_i$)
  $CScore_i = \emptyset$
  for each $Vector_l \in Ngrams_i$ **do**
    $CScore_i(Vector_l, score_l)$ = SemanticDistance($Vector_l$, $EmbedVectors_i$)
  end for
  $RankedTC_i = \emptyset$
  for each $Vector_m \in CScore_i$ **do**
    $RankedTC_i$ = MaximumMarginalRelevance($Vector_m$, $corpus_i$)
  end for
  TC_i = TopK($RankedTC_i$)
  TC = TC $\cup\{(T_i, TC_i) \cap TT_i\}$
end for
return TC

similarity with the topic-corpus at the respective level using Equation 7:

$$TopicLabel_i^k = argmax(cossim(CorpusGloveVec_i^k, TermGloveVec_i^k)) \quad (7)$$

for i^{th} topic at k^{th} level.

E. EVALUATION METRICS

There have been discussions over proper evaluation metrics at each granular level of the problem statement. Relevant evaluation metrics are an essential factor in comparing the performance of the target problem. Some authors have also proposed supervised methodologies to label the topics [34], [42]. The supervised methodology's performance can be evaluated using the F1 score. However, the F1 score metric is not suitable for the HTL. The F1 score requires information for both positive and negative classes to calculate false-positive and false-negative labels. Such information is usually absent from the dataset. The evaluation metric must also ensure the hierarchical relationship. Authors in [9] have discussed various limitations in the popular evaluation metrics of TM. In our work, we have used the following evaluation metrics:

- **Exact match:** A label L is an exact match of the correct label S with parent label P, if either L or a synonym SL of L such that SL or L is exact same string as S, SP or PS [14]. It is calculated as ratio of exact labels counts to the total number of labels as:
$$EM = \frac{ExactMatchCounts * 100}{TotalLabels} \quad (8)$$
- **Partial match:** A label L is an partial match of the correct label S with parent label P, if either L or a synonym SL of L such that SL or L is shares a unit of string with S, SP or PS [14]. It is calculated as ratio of partial match counts to the number of labels as:
$$PM = \frac{PartialMatchCounts * 100}{TotalLabels} \quad (9)$$
- **Cosine similarity:** Use word vectors to convert the topic label as text vector, documents in the topic as average word vector. Consider a topic $T_i = \{d_1, d_2, d_3, \ldots, d_n\}$ carries n documents. Cosine similarity between Topic Vector (TV) and word vector (W) can be define as:
$$cossim(TV, W) = \frac{TV.W}{||TV||||W||} \quad (10)$$
where, Topic Vector (TV) corresponding to the n documents belonging in the respective topic as:
$$TV = \frac{\sum_{i=1}^{n} DV(D_i)}{n} \quad (11)$$
and Document Vector (DV) with m as:
$$DV = \frac{\sum_{j=1}^{m} DV(w_j)}{m} \quad (12)$$

The word vectors could be derive with the various strategies like skip-gram [69], word2vec [68] or pre-trained word vectors like GloVe [61], fastText [62], [63]. The cosine similarity ranges from 1 to −1. The negative score represents the disimilarity among the vectors. −1 represents complete opposite and 1 represents complete similarity among the vectors.

- **Human Evaluation:** Due to standard data unavilability and multi-dimensional features of TL, authors also used human evaluation [42], [45]. The primary motive of TL is to make the topics most relevant for the human reader. However, human evaluation has various limitations like limited scalibility, cost and lack of definite decisions.

Among these evaluation strategies, exact match and partial match matrices consider the hierarchical relationship among the topic labels. Authors in [70] introduced exact match and partial match matrices for labeling hierarchical document clusters. The cosine similarity score ranges from −1 to 1, where −1 represents complete dissimilarity, and 1 represents exact similarity. The matching score ranges from 0 to 100, where 0 represents no match. We have used the same evaluation metrics in our work.

V. EXPERIMENTS

In this section, we have discussed the datasets used. The dataset plays an essential role in the evaluation of the task. We have discussed the strengths, limitations, and steps to minimise the limitations of the datasets. We have incorporated some pre-processing steps based on our data exploration to minimise the limitations. The pre-processing steps are maintained with the standard set of steps that mandatorily avoid any bias in the performance through hyper-processing. We have also discussed the hyperparameters used in the experiments and their contributions to the performance of the methodology.

A. CORPORA

The dataset contributes a vital role to the strategy. We used various keywords like 'topic modeling dataset', 'hierarchical topic modeling dataset', 'topic labeling dataset', 'hierarchical topic labeling dataset', 'text clustering', etc., to search for the appropriate dataset. We manually scrutinize each dataset. The hierarchical relationship is an essential requirement in the dataset to analyze the methodologies effectively. The majority of the datasets do not possess relevant hierarchical information. We explored around 1,892 datasets carrying textual features. Among those datasets, we manually filter the datasets most appropriate for our work. The hierarchical relationship is a must-have requirement for the dataset. The datasets with hard classification are not best suitable for our problem statement. The hard classification led to the multi-label text classification [65]. Multi-label text classification is a stand-alone problem without consideration of any hierarchical relationship. Considering the requirements for our work, we opted for the following datasets:

1) The **Abstract**[4] dataset carries 14004 entities belonging to 29 classes. The classes are so related that the dataset avoids hard classification, i.e., an entity may belong to one or more than one class. Such distribution of classes also helps in considering the hierarchy. For example, among 5884 entities in class *Computer Science* 1376 entities also belong to class *Artificial Intelligence*. This supports problem statement requirement assuming *Computer Science* at level l, then *Artificial Intelligence* at level $l+1$. Similarly, 29 classes broadly concatenate to 4 classes, i.e. *Computer Science, Mathematics, Physics and Statistics*. The dataset carries 5884, 2831, 3856, and 3794 number of instances belonging to *Computer Science, Mathematics, Physics and Statistics* respectively. Such instance distribution avoids biassing in the dataset. It contains 42840 number of unique tokens with 19614 average number of tokens per category. The article abstracts have been popularly used in the text labeling domain [38], [42].
2) The **App summary**[5] dataset carries total 292372 entities and 73011 number of unique tokens. These entities have been distributed into 48 classes. However, the number of entities in each class ranges from 22,978 to 199. To maintain the efficient balance of instances in each class of the dataset, we considered only the top five *genre*, i.e., *Entertainment, Tools, Music & Audio, Education, Books & Reference*. The respective selection led to a number of instances in each class ranging from 22869 to 13170. The filtered dataset contains 84309 instances, 38166 unique tokens, and 12768 average tokens in each class. The *genre* are further merged with *genres*, which gives the essence of hierarchy in the dataset.
3) The **App description**[6] dataset carries total 293392 entities and 879623 number of unique tokens. These entities have been distributed into 61 classes. However, the number of entities in each class ranges from 30716 to 4. We considered the entities belonging to different classes in the previous two datasets. The input dataset has some sense of hierarchy. However, we intend to test the proposed methodology even if the dataset has a minimal hierarchical sense. Hence, in the *App description* dataset, we considered entities that only belong to the *Games* class. The *Games* class carries the highest number of instances, i.e., 30716 instances and 63069 unique tokens, which makes it sufficient for the training. Similar to the *App Summary* dataset, the *'genres'* carries more detailed information about the

[4]https://www.kaggle.com/datasets/abisheksudarshan/topic-modeling-for-research-articles?select=Train.csv

[5]https://www.kaggle.com/datasets/sagol79/stemmed-description-tokens-and-application-genres?resource=download&select=bundles_desc_tokens.csv

[6]https://www.kaggle.com/datasets/sagol79/stemmed-description-tokens-and-application-genres?resource=download&select=bundles_desc_tokens.csv

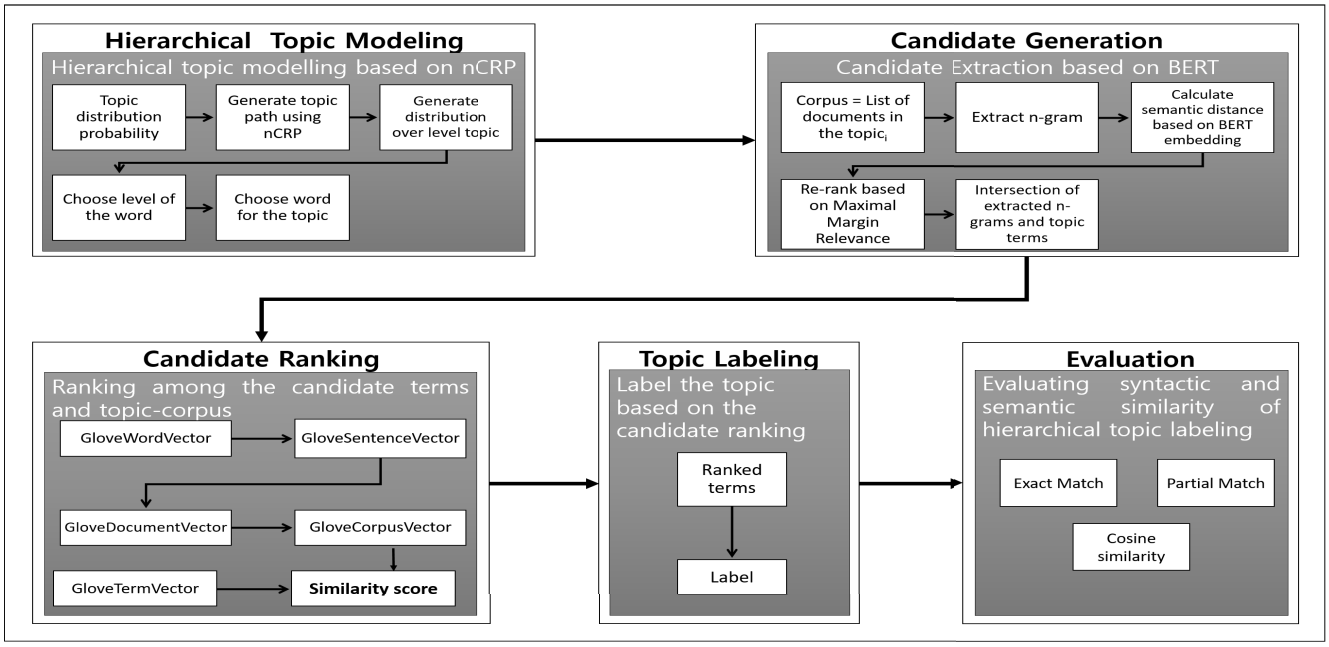

FIGURE 2. **Proposed methodology components.**

TABLE 1. **Dataset labels.**

Dataset	Dataset Labels	Instances
Abstract	Mathematics	2831
	Physics	3856
	Statistics	3794
	Computer Science	5884
App Description	Games	30716
App Summary	Entertainment	22869
	Tools	17555
	Music & Audio	17428
	Education	13287
	Books & Reference	13170

TABLE 2. **Description of dataset used in experiments.**

Dataset	Features	Instances
Abstract	Number of instances	14004
	Total number of unique tokens	42840
	Avg number of tokens per class	19614
App Description	Number of instances	30716
	Total number of unique tokens	63069
	Avg number of tokens per class	63069
App Summary	Number of intances	84309
	Total number of unique tokens	38166
	Avg number of tokens per class	12768

description of the app that helps expert annotators for labeling with the hierarchical relationship.

The relevant information about the datasets for experiments being represented in the Table 1 and Table 2.

B. PRE-PROCESSING

The abstract of an article is part of a formal document bound to follow standard grammar. However, the description and summary of apps are informal documents. The app developers use multiple creative ways to write the description and summary that can impress the users. The absence led to various noises in the dataset. The pre-processing of text plays an essential role in the performance of architecture [66], [67]. We followed the following pre-processing steps:

- Removed all the Non-English alphabets.
- Replaced any emoji with the respective text using an open source tool.[7]
- Replaced symbol based emojis with the respective text for example, *<3* with *heart*, *:-(* with *sadface*, etc.
- Since our methodology have not considered the hyperlink relations of the document. We have replaced any web url with *url*.
- Generally host developer use *@username* to acknowledge other developers or collaborators in the description or summary of the app. Since these named-entities have minimal eminance to the topic of the document, we have removed such entities in the pre-processing.
- Since unsupervised HTM methodology depends upon the terms available in the corpus, the methodology

[7]https://pypi.org/project/emoji/

TABLE 3. Hyperparameter values used in the experiments.

Hyperparameter	Value
alpha	[3, 4, 5, 6, 7]
gamma	0.5
eta	0.9
num_levels	3
n_samples	20

may get biassed with the availability of stopwords.[8] Considering this assumption, we removed the stopwords using a popular open-source *nltk*[9] library from the corpus before modeling.

C. HYPERPARAMETERS

The performance of our proposed methodology indirectly depends on the quality of HTM. The hyperparameters significantly influence the number of topics, topic-word distribution, document-topic coherence, and hierarchical relations. To create the hierarchical tree for HTM, we implemented an nCRP-based strategy using an open-source library.[10] To get optimum output, we performed experiments with the following hyperparameters:

- **alpha:** It manages smoothing over level distributions.
- **gamma:** In nCRP, table-customer distribution, it considers the number of imaginary customers at next, as yet unused table.
- **eta:** This is an important hyperparameter that considers the smoothing over topic-word distributions.
- **num_levels:** It led to decide the number of levels in the hierarchical topic tree.
- **n_samples:** number of iterations for the sampler.

Table 3 mentions exact values used in the experiments.

VI. RESULTS AND DISCUSSION

Authors in [36] and [35] preferred semantic relevancy, easy understanding and high coverage as essential features of gold standard labeling. After intense scrutiny, we included datasets that have the potential to be represented in hierarchical relationships. However, hierarchical labeling is absent for all the instances. Authors in [35] mention human understanding as a fundamental requirement of the TL. Hence, we compared our proposed methodology using the gold standard labels generated by human annotators. Considering the challenges in human annotations and dataset strength, we have limited our experiments to a three-level hierarchy. The evaluation against human-generated labels overcomes the limitations of the unsupervised method. This strategy also validates performance concerning the core requirement of labeling. To maintain the gold standard for annotation, we chose a group of human annotators who were experts in diverse domains. Along with domain diversity, we maintained experience diversity in the group. The annotators have experience ranging from 1 year to 15 years in research principles.

In this section, we have compared our results with the Dataset Labels and, TWL [14] as benchmark. We have discussed stat based proposed TWL method in detail in Section II. TWL considers text-based information in its strategy. Another related work contribution, [39] infused a broad range of features that also require hyperlink availability in the dataset and other related information. Every dataset is not enriched with such information. In this work, we aim to propose a methodology that supports a broad range of applications. Hence, we exclude any such dependencies from our work. In App summary and App description dataset we considered *genre* and, *classes* of Abstract dataset as *Dataset Labels* for comparison. The *genres* and *other classes* information in the datasets was useful to human annotators to maintain the hierarchical labeling. However, due to multiple missing information in *genres* and *other classes* they did not qualify as Dataset Labels.

Since proposed methodology and TWL are unsupervised and graph based, they seems to be time consuming. Candidate generation and candidate ranking are two vital steps in both the methodologies. For calculating the time complexity, we have assumed DN as the number of documents, V as the vocabulary size of the respective topic-corpus, TN as the number of topics, TL number of child topics, TS as the number of sibling topics, C as the number of candidate terms and NG as the number of ngrams. The TWL have used the Ngram Testing strategy to generate candidates. The Ngram testing strategy follows three sequential steps, i.e., phrase extraction, document reweighting, and phrase diversification. The time complexity of phrase extraction, document re-weighting, and phrase diversification is $\mathcal{O}(NG+NG\log NG)$, $\mathcal{O}(DN \times NG)$, $\mathcal{O}(NG^2)$, respectively. The value of DN is trivially smaller than NG. Hence, the candidate generation time complexity for all topics in the TWL is $\mathcal{O}(TN \times NG^2)$. The candidate generation process in our methodology depends upon six sequential steps, i.e., topic-corpus creation, processing steps, ngram semantic distance calculation, maximal margin relevance for each ngram, ranking ngrams and extracting top K ngrams. Assuming the calculation time of distance and relevance constant, the time complexity of our proposed methodology is $\mathcal{O}(TN(DN + NG + NG + NG\log NG + K))$. Since the value of K must be less than equal to NG hence, the net candidate generation time complexity is $\mathcal{O}(TN \times NG\log NG)$. The candidate ranking step has the highest impact on the time complexity of the TWL strategy. The candidate ranking time complexity of TWL is $\mathcal{O}(TN^2 \times C^2 \times TS \times TL)$ whereas, the candidate ranking time complexity of proposed methodology is $\mathcal{C}\log\mathcal{C}$. The time complexity of our proposed methodology is much lower than the benchmark methodology.

The previous works for HTL have used exact match and partial match for evaluating the performance of proposed methodologies [14]. However, various other contributions

[8] 'stopwords' are a list of words that have a high frequency in the corpus, but do not contribute any convincing information regarding the domain of the corpus.

[9] https://www.nltk.org/

[10] https://pypi.org/project/hlda/

in the TL preferred cosine similarity. The cosine similarity is the preferred evaluation matrix as it minimises the need for human evaluation. We used Equation 10 discussed in Section IV-E for calculating the cosine similarity. For each topic, we have a single unit TWL label and a proposed methodology label, but multiple Dataset labels. In unsupervised hierarchical topic tree generation, the topic may include entities belonging to different dataset labels. To address this issue, we have calculated the average of Dataset Labels vectors using Equation 6. Figure 3 and Figure 5 represent topic-wise cosine similarity among TWL, proposed methodology, and Dataset Labels. Considering the space limitation in the article, rather than presenting the similarity score of each topic, normalised topic buckets have been presented. The size of a bucket and the number of buckets depend on the number of total topics in the respective hierarchical tree. We have maintained the bucket size to keep it closest to the even number of topic distributions and elegant representation. Figure 4 and Figure 6 represent level-wise cosine similarity among TWL, proposed methodology. In the level-wise presentation, we excluded level 0, i.e., root node label similarity with the corpus. As we assume, the root is the most abstract level in the hierarchy that does not require automation.

Considering the diversity of the datasets, we have discussed the results with respect to each dataset in the following subsections:

A. ABSTRACT DATASET

The size of the Abstract dataset is about one-tenth of the other datasets. The abstract corpus carries 42198 tokens, with a maximum of 607 tokens in a single entity. The average number of tokens in each entity is 150. We chose this dataset to test the methodology against a well-structured short text dataset.

Table 4 shows the performance of the proposed labeling methodology with respect to the class relevancy information in all the topics. Since the vectors have been converted into Glove vectors using Equation 3, the similarity considers multi-dimensional information. Since, for different *alpha* values, we got a different number of topics. We consider the normalised similarity score with the number of topics in the respective *alpha* values. Our proposed hierarchical topic labels outperformed benchmark and Dataset Labels respective to the cosine similarity for different parameters. It represents the relevancy of labels to the corpus. Our proposed methodology performed best for *alpha* 4. The Dataset Labels score maximum similarity score for *alpha* 7. The Dataset Labels performed close to the proposed methodology for a few *alpha* values; however, the proposed methodology significantly outperformed TWL. Our proposed methodology improved the similarity score comparing the dataset labels with 580.65% and TWL with 114.81%. The average cosine similarity difference between the best and least performing *alpha* is 0.0951.

TABLE 4. Average cosine similarity of hierarchical topic label over Abstract dataset for multiple alpha values.

alpha	Methodology	Dataset Labels	TWL
3	**0.2671**	0.2549	0.0400
4	**0.3622**	0.2581	0.0664
5	**0.3464**	0.2797	0.0758
6	**0.2720**	0.2691	0.0121
7	**0.3102**	0.2973	0.0740

Table 4 represents the abstract level performance analysis of hierarchical topic labels. Figure 3 presents a more detailed topic-wise analysis of each label. Considering the diversity in the frequency of topics for different parameters, we presented the performance using buckets of topics for each parameter. The number of buckets and size of each bucket depend upon the frequency of the topics. As presented in Figure 3, the proposed methodology outperformed the majority of buckets. However, the proposed methodology struggles in a few topics to outperform the Dataset Labels. The TWL has also outperformed the proposed methodology and Dataset Labels for a few topics. However, TWL has also scored negatively for a few topics. It represents the dissimilarity of labels generated through TWL with the documents in the respective topics.

The similarity of a three-topic bucket scored zero for *alpha* 5 and 7, as presented in Figure 3. It presents no similarity between the three topics and the proposed label. In other words, it represents the disjointed behaviour of topic terms and keywords extracted. The topic terms are part of the HTM. Such cases represent the cascading performance impact of HTM over HTL. HTM and the performance evaluation of HTM is a stand-alone task [9] and is out of the scope of this paper.

Level-wise discussion is essential for HTL. Table 5 presents the level-wise[11] similarity score of labels. We have ignored the Dataset Labels in the level-wise similarity discussion due to various limitations in the Dataset Labels discussed above. The benchmark outperformed the proposed methodology for level 1 with *alpha* 3. The proposed methodology outperformed the benchmark at each level for every diverse parameter. The proposed methodology scored best at *alpha* 7 and at *alpha* 4 for level 1 and level 2, respectively. The performance differences at levels 1 and 2 for each *alpha* are as expected. Level 1's label covers a broad range of diversified domain documents compared to level 2.

Table 6 and Table 7 represent the comparative performance of the proposed methodology for hierarchical label matching. Hierarchical labeling evaluation is preferred over the exact match and partial match. Section IV discussed the relevancy of these metrics for HTL. The proposed methodology outperformed others for most of the *alpha* values. TWL outperformed the proposed methodology for *alpha* values 4 and 7, however, with a thin margin. The proposed methodology scored highest for *alpha* 4. The performance difference is also highest for *alpha* 4. The proposed

[11] In our work, we have assumed root at Level 0.

TABLE 5. Hierarchical levelwise cosine similarity in the Abstract dataset.

alpha	Levels	Proposed	TWL
3	Level 1	0.0036	**0.0383**
	Level 2	**0.2854**	0.0326
4	Level 1	**0.1381**	0.1271
	Level 2	**0.3921**	0.0525
5	Level 1	**0.2820**	-0.0287
	Level 2	**0.3649**	0.0836
6	Level 1	**0.2757**	0.1664
	Level 2	**0.3282**	0.0399
7	Level 1	**0.3094**	0.0769
	Level 2	**0.2782**	-0.0105

TABLE 6. Exact Match labeling performance over Abstract dataset.

alpha	Methodology	Dataset Labels	TWL
3	**35.45**	11.29	27.41
4	**53.19**	2.12	21.27
5	33.34	3.92	**35.29**
6	**40.00**	1.05	26.67
7	33.34	7.50	**37.03**

TABLE 7. Partial Match labeling performance over Abstract dataset.

alpha	Methodology	Dataset Labels	TWL
3	**38.70**	12.90	32.25
4	**53.19**	4.25	27.65
5	**39.21**	5.88	35.29
6	**46.67**	1.02	26.67
7	**37.03**	7.40	**37.03**

methodology significantly outperformed the Dataset Labels and benchmark with 33.88 and 9.53, respectively, for the average exact match score. The performance limitation of Dataset Labels is due to various challenges in the dataset.

The partial match score has slightly improved performance over the exact match. The impact of a partial match is higher for multi-token labels. We restricted ourselves to unigrams in our work, but we did not apply such constraints to human annotators. Our proposed methodology outperformed the Dataset Labels and TWL with a difference of 36.664 and 11.176, respectively. The proposed methodology scored highest for *alpha* 4. Interestingly, the proposed methodology and TWL scored equally for *alpha* 7, up to two decimal places.

B. APP SUMMARY DATASET

The summary of apps is a robust dataset to present the relevancy of HTL for a large number of applications. The proposed methodology in the App summary dataset deals with various challenges like noises, high diversity, and alphanumeric tokens. These challenges are more prevalent for supervised algorithms. In our work, we have used the preprocessing steps discussed in Section V to reduce the impact of noises.

Table 8 presents the average topic similarity of proposed labels to the respective documents. The proposed methodology outperformed the available dataset labels and TWL

TABLE 8. Average cosine similarity of hierarchical topic label over App Summary dataset for multiple alpha values.

alpha	Methodology	Dataset Labels	TWL
3	**0.5781**	0.5370	0.0405
4	**0.5675**	0.5350	0.0625
5	**0.5564**	0.5353	0.0735
6	**0.5593**	0.5365	0.0608
7	**0.5606**	0.5386	0.0430

for each *alpha* value. However, the proposed methodology outperformed Dataset Labels only by a thin margin. In each experiment, even the least performance score of the proposed methodology surpasses the highest-performing Dataset Labels score. The proposed methodology achieves an average similarity score of 0.56438. The average similarity score of the proposed methodology is higher than the Dataset Labels and benchmark TWL by 0.0279 and 0.50832, respectively. The proposed methodology has significantly outperformed the benchmark.

Similar to Figure 3, Figure 5 presents the topic-wise similarity in buckets. Since the dataset size of the App summary is comparatively larger than the Abstract dataset, the number of topics also behaves the same. We have followed the same assumptions for bucket frequency and size as in the Abstract dataset. The majority of topic-buckets have higher similarity for the proposed methodology. The performance of Dataset Labels does not have sharp fluctuations over multiple parameters. The consistent score of dataset labels presents the uniformity of the Dataset Labels.

As discussed, the significance of level-wise similarity, Table 9 and Figure 5 represent level-wise similarity for each *alpha*. Our proposed methodology outperformed the benchmark for *alpha* 7 at both levels. In other *alpha* parameters, our proposed methodology outperformed at level 2 but struggled at higher levels. The proposed methodology scores an average cosine similarity of 0.1009 for level 1 and 0.5668 at level 2.

Our proposed methodology significantly outperforms the benchmark and Dataset Labels for both exact match and partial match. The proposed methodology scored an average of 63.62 and 64.08 for exact match and partial match, respectively. The score difference between partial match and exact match is significant for the Abstract dataset but small for the App summary dataset.

C. APP DESCRIPTION DATASET

The App description dataset carries only single-category instances. This dataset carries the most noise compared to the other two datasets. The cosine similarity of the Dataset Label would be insignificant for this dataset. Since all the instances belong to only one genre, all the topics will have only one dataset label. In other words, the cosine similarity of the dataset label will represent the cosine similarity of the corpus with a single label divided into parts (topics). Table 12 and Table 13 represent the cosine similarity comparison of the proposed methodology and benchmark. The performance

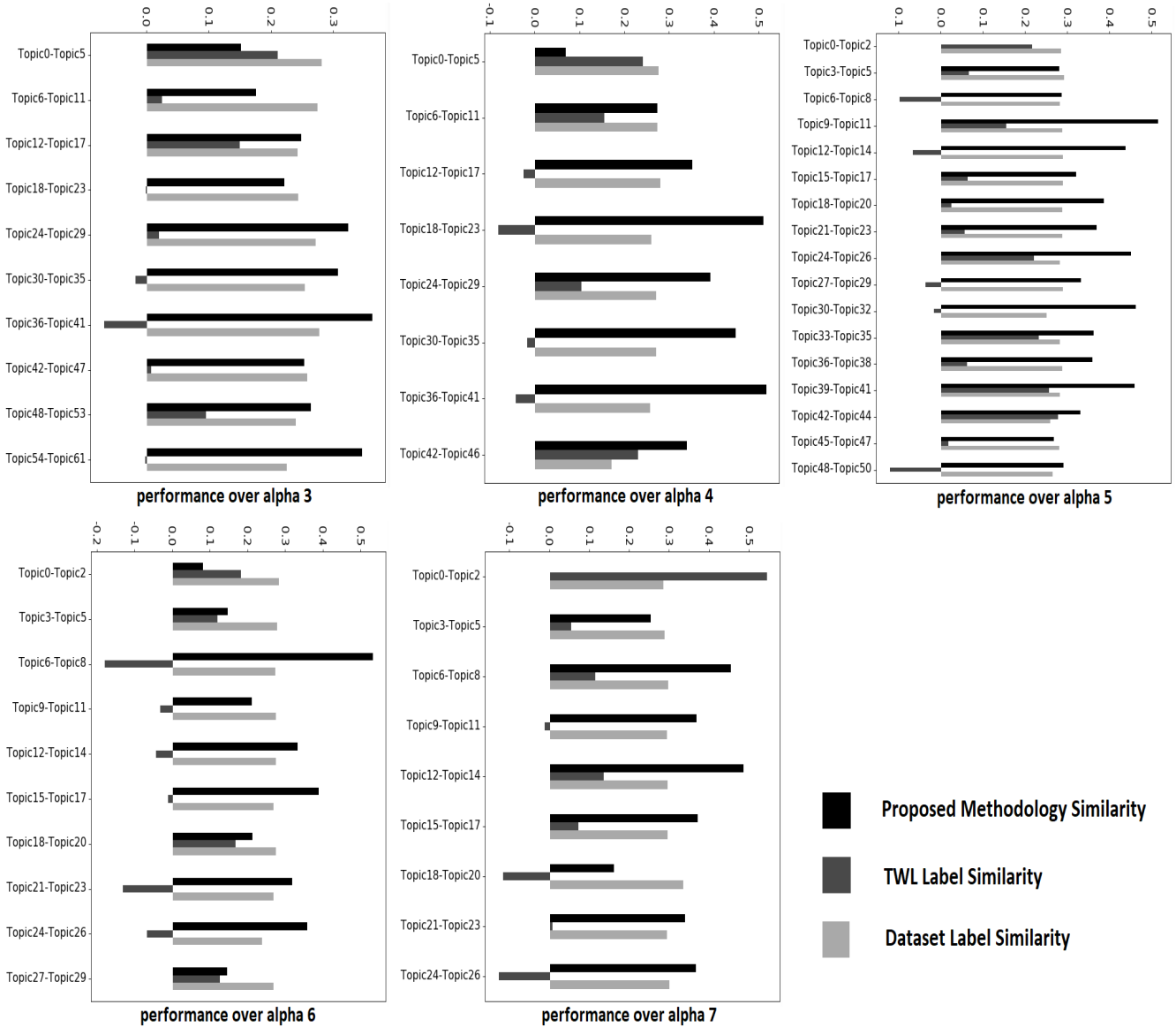

FIGURE 3. Topic-wise cosine similarity performance of proposed methodology, dataset given labels and benchmark over Abstract dataset.

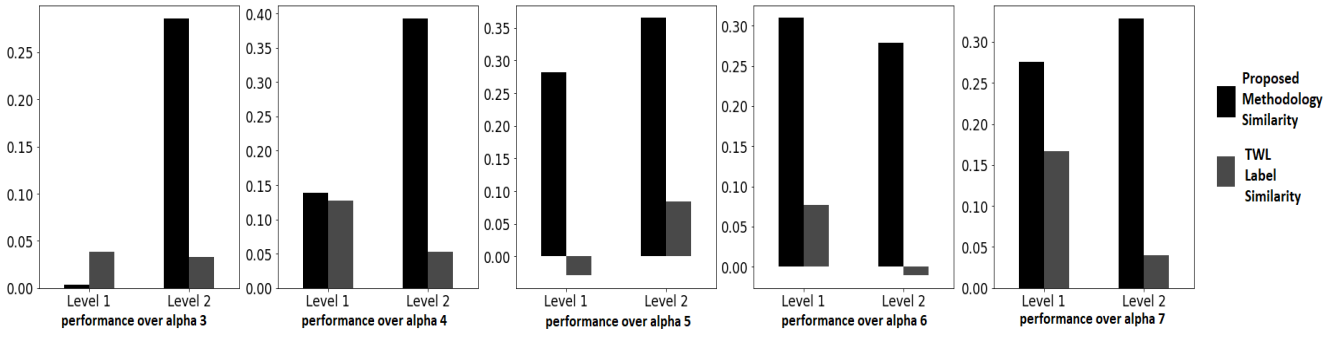

FIGURE 4. Level-wise cosine similarity performance of proposed methodology and benchmark over Abstract dataset.

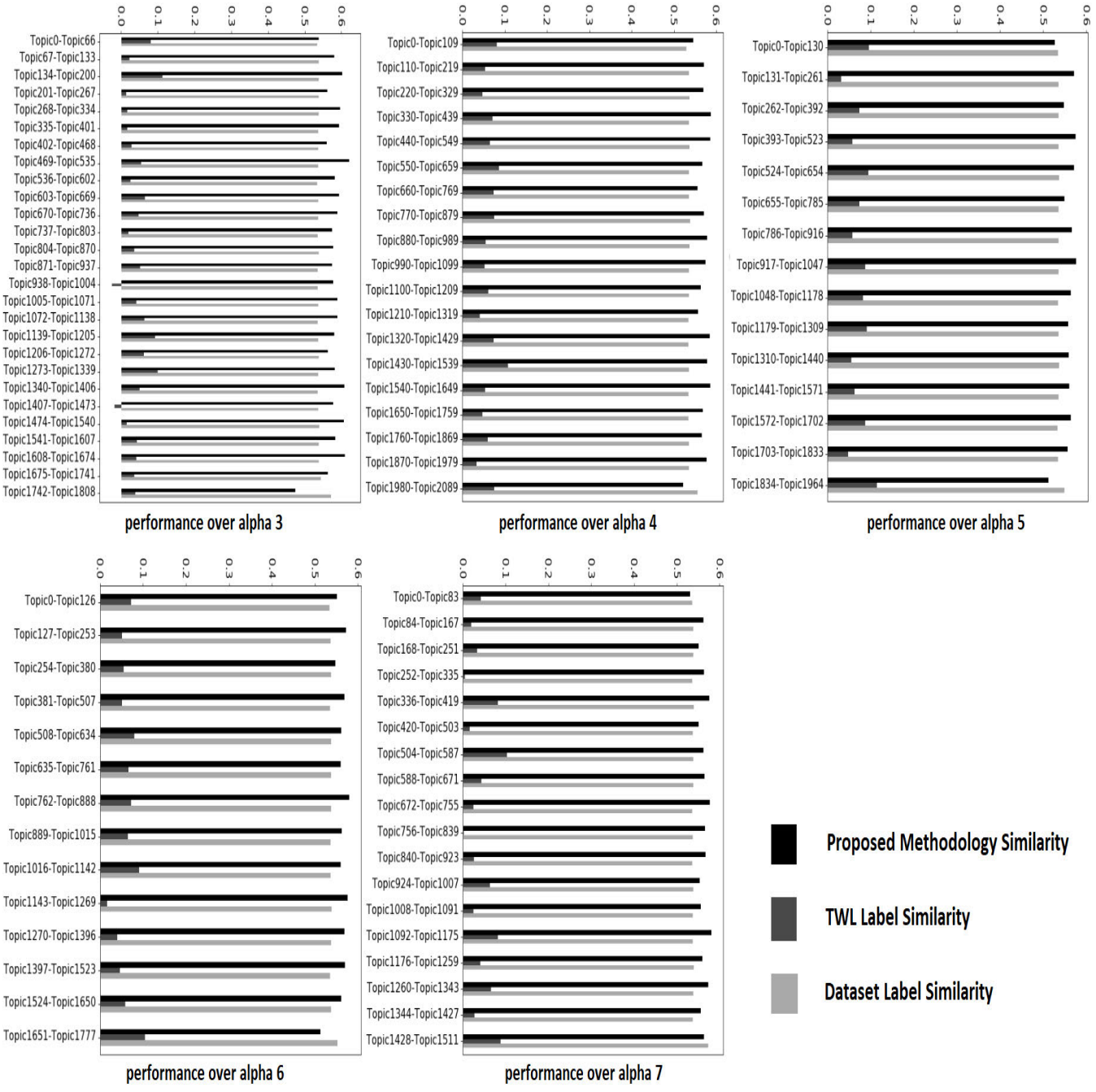

FIGURE 5. Topic-wise cosine similarity performance of proposed methodology, dataset given labels and SOTA over App summary dataset.

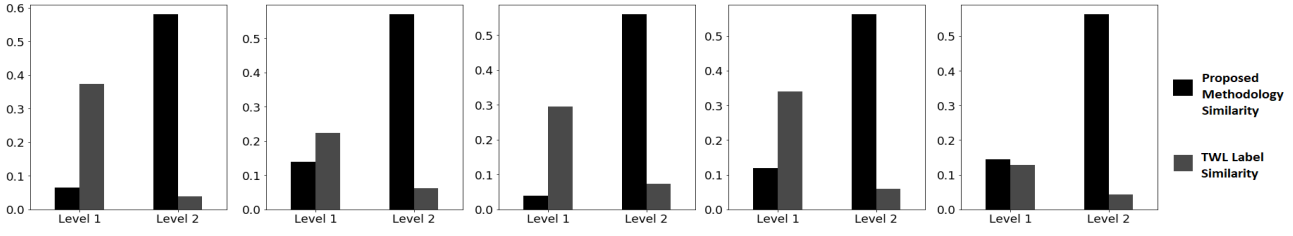

FIGURE 6. Level-wise cosine similarity performance of proposed methodology and benchmark over App summary dataset.

TABLE 9. Hierarchical levelwise cosine similarity in the App summary dataset.

alpha	Levels	Proposed	TWL
3	Level 1	0.0645	**0.3728**
	Level 2	**0.5805**	0.0392
4	Level 1	0.1381	**0.2241**
	Level 2	**0.5698**	0.0618
5	Level 1	0.0390	**0.2956**
	Level 2	**0.5591**	0.0726
6	Level 1	0.1191	**0.3406**
	Level 2	**0.5621**	0.0592
7	Level 1	**0.1438**	0.1290
	Level 2	**0.5629**	0.0427

TABLE 10. Exact Match labeling performance over App Summary dataset.

alpha	Methodology	Dataset Labels	TWL
3	**65.06**	2.54	30.90
4	**62.80**	1.91	33.87
5	**61.11**	1.67	35.55
6	**63.83**	1.96	33.46
7	**65.34**	1.91	31.95

TABLE 11. Partial Match labeling performance over App Summary dataset.

alpha	Methodology	Dataset Labels	TWL
3	**65.78**	2.54	31.17
4	**63.30**	1.91	34.30
5	**61.62**	1.67	35.72
6	**64.11**	1.96	33.46
7	**65.60**	1.91	32.01

TABLE 12. Average cosine similarity of hierarchical topic label over App description dataset for multiple alpha values.

alpha	Methodology	TWL
3	**0.2973**	0.099
4	**0.3801**	0.0604
5	**0.3992**	0.1243
6	**0.3840**	0.0819
7	**0.4385**	0.1186

TABLE 13. Hierarchical levelwise cosine similarity in the App description dataset.

alpha	Levels	Proposed	TWL
3	Level 1	0.1567	**0.2026**
	Level 2	**0.3006**	0.0961
4	Level 1	**0.2039**	-0.0515
	Level 2	**0.3860**	0.0630
5	Level 1	**0.1711**	0.0482
	Level 2	**0.4069**	0.1253
6	Level 1	**0.1470**	0.0672
	Level 2	**0.3908**	0.0808
7	Level 1	**0.3006**	-0.07573
	Level 2	**0.4421**	0.1223

on the App description dataset follows the trend of higher similarity at a lower level for each *alpha*. The proposed methodology achieves the highest similarity for both the levels for *alpha* 7. The proposed methodology significantly outperformed the benchmark with a 0.3798 average cosine similarity.

TABLE 14. Exact Match labeling performance over App description dataset.

alpha	Methodology	Dataset Labels	TWL
3	39.58	8.03	**51.12**
4	**52.23**	6.36	40.76
5	41.60	4.90	**50.90**
6	**48.52**	6.61	44.11
7	**47.19**	6.54	44.85

TABLE 15. Partial Match labeling performance over App description dataset.

alpha	Methodology	Dataset Labels	TWL
3	40.17	8.03	**51.48**
4	**52.86**	6.36	40.76
5	42.37	4.95	**51.93**
6	**48.89**	6.61	44.48
7	**48.13**	6.65	44.85

The performance of the proposed methodology over HTL is represented in Table 14 and Table 15. Our proposed methodology achieved an average score of 45.82 for exact matching and 46.48 for partial matching. Even with a single genre corpus, the proposed methodology efficiently covers hierarchical labels for the topics. The exact and partial match scores are very close in the App description dataset. Interestingly, the benchmark has outperformed the proposed methodology for *alpha* 3 and 5 in matching. It represents the limitation of the proposed methodology for highly noisy datasets. The limited matching score of Dataset Labels is due to the dataset limitations we discussed above.

VII. CONCLUSION AND FUTURE WORK

Since the amount of data available is increasing, labeling each cluster of documents is insufficient. Hence we propose hierarchical topic labeling that facilitates better insight into the documents belonging to diverse topics. It also helps to get the most similar documents belonging to the closest topics. It ensures the relevant information extraction at the desired depth. Our proposed methodology ensures the most suitable topic coherence for the hierarchy labeling. However, our methodology considers the interactions between topic terms and extracted keywords, and it may encounter no common terms. Such cases may limit the performance of the proposed methodology. We also understand the cascading impact of the performance of HTM and other components over HTL. In future work, there is scope for improvement in every component. The small performance differences in cosine similarity in the Abstract dataset for different *alphas* motivate further experimentation in future work to optimise other parameters.

The proposed methodology significantly outperformed the benchmark. However, for either of the datasets, the maximum cosine similarity only reached 0.5781, a partial match to 65.78 and exact match to 65.34. The intent to evidently

demonstrate the scope of improvement at different levels encouraged us to discuss the proposed methodology in the components. Being a graph-based method, the proposed methodology may not be well suited for large-size datasets. For future work, we plan to reduce the dimensions while creating the hierarchical tree. To further improve the performance, we also intend to use hyperparameter optimisation algorithms to tune the essential hyperparameters. In this work, our contribution is limited to uni-grams. However, we understand some labels will be more relevant as multi-gram. We intend to design an architecture suitable for multi-gram hierarchical topic labels.

ACKNOWLEDGMENT

The authors would like to thank the volunteers Surabhi Malav, Vikas Malviya, Kushagra Awasthi, Deepika Shukla, and Sahil Rai for their efforts in appropriate labeling the topics respective the document clusters.

REFERENCES

[1] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, 1990.

[2] Y. Yiran and S. Srivastava, "Aspect-based sentiment analysis on mobile phone reviews with LDA," in *Proc. 4th Int. Conf. Mach. Learn. Technol.*, Jun. 2019, pp. 101–105.

[3] A. Fuad and M. Al-Yahya, "Analysis and classification of mobile apps using topic modeling: A case study on Google play Arabic apps," *Complexity*, vol. 2021, pp. 1–12, Feb. 2021.

[4] E. Meeks and S. B. Weingart, "The digital humanities contribution to topic modeling," *J. Digit. Humanities*, vol. 2, no. 1, pp. 1–6, 2012.

[5] L. Liu, L. Tang, W. Dong, S. Yao, and W. Zhou, "An overview of topic modeling and its current applications in bioinformatics," *SpringerPlus*, vol. 5, no. 1, pp. 1–22, Dec. 2016.

[6] J. W. Mohr and P. Bogdanov, "Introduction—Topic models: What they are and why they matter," *Poetics*, vol. 41, no. 6, pp. 545–569, 2013.

[7] Q. Mei, X. Shen, and C. Zhai, "Automatic labeling of multinomial topic models," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2007, pp. 490–499.

[8] R. Churchill and L. Singh, "The evolution of topic modeling," *ACM Comput. Surv.*, vol. 54, no. 10, pp. 1–35, Jan. 2022.

[9] A. Hoyle, P. Goel, A. Hian-Cheong, D. Peskov, J. Boyd-Graber, and P. Resnik, "Is automated topic model evaluation broken? The incoherence of coherence," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 2018–2033.

[10] A. A. Al-Subaihin, F. Sarro, S. Black, L. Capra, M. Harman, Y. Jia, and Y. Zhang, "Clustering mobile apps based on mined textual features," in *Proc. 10th ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas.*, Sep. 2016, pp. 1–10.

[11] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Exp. Syst. Appl.*, vol. 165, Mar. 2021, Art. no. 113679.

[12] D. Magatti, S. Calegari, D. Ciucci, and F. Stella, "Automatic labeling of topics," in *Proc. 9th Int. Conf. Intell. Syst. Design Appl.*, 2009, pp. 1227–1232.

[13] R. Mizoguchi, "Part 3: Advanced course of ontological engineering," *New Gener. Comput.*, vol. 22, no. 2, pp. 193–220, Jun. 2004.

[14] X.-L. Mao, Z.-Y. Ming, Z.-J. Zha, T.-S. Chua, H. Yan, and X. Li, "Automatic labeling hierarchical topics," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manag.*, Oct. 2012, pp. 2383–2386.

[15] J. H. Lau, D. Newman, S. Karimi, and T. Baldwin, "Best topic word selection for topic labelling," in *Proc. Coling*, 2010, pp. 605–613.

[16] S. Bhatia, J. H. Lau, and T. Baldwin, "Automatic labelling of topics with neural embeddings," in *Proc. 26th Int. Conf. Comput. Linguistics*, 2016, pp. 953–963.

[17] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1188–1196.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 1–9.

[19] T. Joachims, "Training linear SVMs in linear time," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2006, pp. 217–226.

[20] W. Kou, F. Li, and T. Baldwin, "Automatic labelling of topic models using word vectors and letter trigram vectors," in *Proc. 11th Asia Inf. Retr. Societies Conf. (AIRS)*. Cham, Switzerland: Springer, 2015, pp. 253–264.

[21] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Stanford Univ., Stanford, CA, USA, Tech. Rep. SIDL-WP-1999-0120, 1999.

[22] J. H. Lau, K. Grieser, D. Newman, and T. Baldwin, "Automatic labelling of topic models," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Human Lang. Technol.*, 2011, pp. 1536–1545.

[23] I. Hulpus, C. Hayes, M. Karnstedt, and D. Greene, "An eigenvalue-based measure for word-sense disambiguation," in *Proc. 24th Int. FLAIRS Conf.*, 2012, pp. 1–6.

[24] K. T. Shahriar, M. A. Moni, M. M. Hoque, M. N. Islam, and I. H. Sarker, "SATLabel: A framework for sentiment and aspect terms based automatic topic labeling," in *Proc. MIDAS*, 2022, pp. 63–75.

[25] D. He, M. Wang, A. M. Khattak, L. Zhang, and W. Gao, "Automatic labeling of topic models using graph-based ranking," *IEEE Access*, vol. 7, pp. 131593–131608, 2019.

[26] C.-O. Truica and E.-S. Apostol, "TLATR: Automatic topic labeling using automatic (domain-specific) term recognition," *IEEE Access*, vol. 9, pp. 76624–76641, 2021.

[27] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, no. 4, pp. 333–389, 2009.

[28] K. Frantzi, S. Ananiadou, and H. Mima, "Automatic recognition of multi-word terms: The C-value/NC-value method," *Int. J. Digit. Libraries*, vol. 3, no. 2, pp. 115–130, Aug. 2000.

[29] N. A. Sanjaya, M. L. Ba, T. Abdessalem, and S. Bressan, "Harnessing truth discovery algorithms on the topic labelling problem," in *Proc. 20th Int. Conf. Inf. Integr. Web-Based Appl. Services*, Nov. 2018, pp. 8–14.

[30] I. D. Mienye and Y. Sun, "A survey of ensemble learning: Concepts, algorithms, applications, and prospects," *IEEE Access*, vol. 10, pp. 99129–99149, 2022.

[31] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han, "A survey on truth discovery," *ACM SIGKDD Explor. Newslett.*, vol. 17, no. 2, pp. 1–16, Feb. 2016.

[32] I. Hulpus, C. Hayes, M. Karnstedt, and D. Greene, "Unsupervised graph-based topic labelling using DBpedia," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, Feb. 2013, pp. 465–474.

[33] M. McDaniel and V. C. Storey, "Evaluating domain ontologies: Clarification, classification, and challenges," *ACM Comput. Surv.*, vol. 52, no. 4, pp. 1–44, Jul. 2020.

[34] E. Zosa, L. Pivovarova, M. Boggia, and S. Ivanova, "Multilingual topic labelling of news topics using ontological mapping," in *Proc. 44th Eur. Conf. IR Res.* Springer, Cham, 2022, pp. 248–256.

[35] M. Allahyaria, S. Pouriyeha, K. Kochuta, and H. R. Arabniaa, "OntoLDA: An ontology-based topic model for automatic topic labeling," Tech. Rep., 2009.

[36] M. Allahyari, S. Pouriyeh, K. Kochut, and H. Reza, "A knowledge-based topic modeling approach for automatic topic labeling," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 9, p. 335, 2017.

[37] S. A. Kinariwala and S. Deshmukh, "Onto_TML: Auto-labeling of topic models," *J. Integr. Sci. Technol.*, vol. 9, no. 2, pp. 85–91, 2021.

[38] H. H. Kim and H. Y. Rhee, "An ontology-based labeling of influential topics using topic network analysis," *J. Inf. Process. Syst.*, vol. 15, no. 5, pp. 1096–1107, 2019.

[39] R. Kozono and R. Saga, "Automatic labeling for hierarchical topics with NETL," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 3740–3745.

[40] T. Griffiths, M. Jordan, J. Tenenbaum, and D. Blei, "Hierarchical topic models and the nested Chinese restaurant process," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 16, 2003, pp. 1–8.

[41] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies," *J. ACM*, vol. 57, no. 2, pp. 1–30, Jan. 2010.

[42] A. Alokaili, N. Aletras, and M. Stevenson, "Automatic generation of topic labels," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2020, pp. 1965–1968.
[43] D. He, Y. Ren, A. M. Khattak, X. Liu, S. Tao, and W. Gao, "Automatic topic labeling model with paired-attention based on pre-trained deep neural network," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–9.
[44] O. Kozbagarov, R. Mussabayev, and N. Mladenovic, "A new sentence-based interpretative topic modeling and automatic topic labeling," *Symmetry*, vol. 13, no. 5, p. 837, May 2021.
[45] C. Popa and T. Rebedea, "BART-TL: Weakly-supervised topic label generation," in *Proc. 16th Conf. Eur. Chapter Assoc. Comput. Linguistics, Main Volume*, 2021, pp. 1418–1425.
[46] T. Hofmann, "Probabilistic latent semantic indexing," in *Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 1999, pp. 50–57.
[47] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn.*, vol. 39, no. 2, pp. 103–134, 2000.
[48] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Jan. 2003.
[49] Y. Jin, H. Zhao, M. Liu, L. Du, and W. Buntine, "Neural attention-aware hierarchical topic model," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 1042–1052.
[50] X. Mao, Z. Ming, T. Chua, S. Li, H. Yan, and X. Li, "SSHLDA: A semi-supervised hierarchical topic model," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process. Comput. Natural Lang. Learn.*, 2012, pp. 800–809.
[51] W. Li and A. McCallum, "Pachinko allocation: DAG-structured mixture models of topic correlations," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 577–584.
[52] F. Viegas, W. Cunha, C. Gomes, A. Pereira, L. Rocha, and M. Goncalves, "CluHTM—Semantic hierarchical topic modeling based on CluWords," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 8138–8150.
[53] Y. Xu, J. Yin, J. Huang, and Y. Yin, "Hierarchical topic modeling with automatic knowledge mining," *Exp. Syst. Appl.*, vol. 103, pp. 106–117, Aug. 2018.
[54] D. J. Aldous, "Exchangeability and related topics," in *École d'Été de Probabilités de Saint-Flour XIII*. Berlin, Germany: Springer, 1985, pp. 1–198.
[55] J. Pitman, "Exchangeable and partially exchangeable random partitions," *Probab. Theory Rel. Fields*, vol. 102, no. 2, pp. 145–158, Jun. 1995.
[56] W. B. W. B. Cavnar and J. M. Trenkle, "N-gram-based text categorization," in *Proc. 3rd Annu. Symp. Document Anal. Inf. Retr.*, vol. 161175, 1994, pp. 1–14.
[57] Q. Liu, M. J. Kusner, and P. Blunsom, "A survey on contextual embeddings," 2020, *arXiv:2003.07278*.
[58] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proc. 21st Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 1998, pp. 335–336.
[59] D. I. Belov and R. D. Armstrong, "Distributions of the Kullback–Leibler divergence with applications," *Brit. J. Math. Stat. Psychol.*, vol. 64, no. 2, pp. 291–309, May 2011.
[60] K. Grieser, T. Baldwin, F. Bohnert, and L. Sonenberg, "Using ontological and document similarity to estimate museum exhibit relatedness," *J. Comput. Cultural Heritage*, vol. 3, no. 3, pp. 1–20, Mar. 2011.
[61] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[62] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "FastText.Zip: Compressing text classification models," 2016, *arXiv:1612.03651*.
[63] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.
[64] S. Kinariwala and S. Deshmukh, "Short text topic modelling using local and global word-context semantic correlation," *Multimedia Tools Appl.*, pp. 1–23, Feb. 2023.
[65] X. Chen, J. Cheng, J. Liu, W. Xu, S. Hua, Z. Tang, and V. S. Sheng, "Multi-label Arabic text classification based on deep learning," in *Proc. 12th Int. Conf. Inf. Commun. Syst. (ICICS)*. Cham, Switzerland: Springer, May 2021, pp. 443–456.
[66] U. Naseem, I. Razzak, and P. W. Eklund, "A survey of pre-processing techniques to improve short-text quality: A case study on hate speech detection on Twitter," *Multimedia Tools Appl.*, vol. 80, nos. 28–29, pp. 35239–35266, Nov. 2021.
[67] P. Tiwari and S. Rai, "Mind your tweet: Abusive tweet detection," in *Proc. Int. Conf. Speech Comput.* Cham, Switzerland: Springer, 2021, pp. 704–715.
[68] K. W. Church, "Word2Vec," *Natural Lang. Eng.*, vol. 23, no. 1, pp. 155–162, 2017.
[69] D. Guthrie, B. Allison, W. Liu, L. Guthrie, and Y. Wilks, "A closer look at skip-gram modelling," in *Proc. 4th Int. Conf. Lang. Resour. Eval.*, 2006, pp. 1222–1225.
[70] P. Treeratpituk and J. Callan, "Automatically labeling hierarchical clusters," in *Proc. Int. Conf. Digit. Government Res.*, 2006, pp. 167–176.

PARAS TIWARI received the B.Tech. degree in computer science and engineering from the Institute of Engineering and Technology, Ayodhya, India, in 2016, and the M.Tech. degree in computer science engineering from the PDPM Indian Institute of Information Technology Design and Manufacturing, Jabalpur, India, in 2019. He is currently a Research Scholar with the Indian Institute of Technology (BHU), Varanasi, India. His major research interests include text classification, sentiment analysis, information extraction, and recommendation systems.

ASHUTOSH TRIPATHI received the Bachelor of Engineering degree from the Oriental Institute of Science and Technology, Bhopal, and the M.Tech. degree from the Indian Institute of Information Technology, Allahabad. He is currently a Ph.D. Scholar with PDPM, IIITDM Jabalpur. His research interests include online social network security, natural language processing, cyber security, nature-inspired algorithms, machine learning, and deep learning.

AVANEESH SINGH received the B.Tech. degree from the Computer Science and Engineering Discipline, Goel Institute of Technology & Management, Lucknow, the M.Tech. degree from the Computer Science and Engineering Discipline, Kamla Nehru Institute of Technology, Sultanpur, Uttar Pradesh, and the Ph.D. degree from the Indian Institute of Information Technology, Design, and Manufacturing, Jabalpur, India. He is currently a Postdoctoral Researcher with the Indian Institute of Technology Kanpur. His current research interests include epidemic forecasting, computational modeling, machine intelligence, deep learning, and big data.

SAWAN RAI received the B.E. degree in computer science and engineering from the Global Engineering College, Jabalpur, India, and the M.Tech. and Ph.D. degrees in computer science and engineering from the PDPM Indian Institute of Information Technology Design and Manufacturing, Jabalpur, in 2017 and 2022, respectively.

His research interests include software engineering, machine learning, and data analytics.