TOP2LABEL: EXPLAINABLE ZERO SHOT TOPIC LABELLING USING KNOWLEDGE GRAPHS

by

Akhil Chaudhary

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
December 2022

I dedicate my dissertation work to my parents, Anita and Rajendra Chaudhary. None of this would have been possible without your sacrifices and dedication to providing me with a quality education. And to my sister Aditi and my better half Anu, who has never left my side and are very special. I also dedicate this dissertation to my many friends who have supported me. I will always appreciate all who have helped me achieve my goals.

Table of Contents

List of Tables

List of Figures

Abstract

Automatic topic labelling aims to generate sound, interpretable, and meaningful topic labels used to interpret topics. A topic is usually represented by a list of terms and documents, ranked by their probability, and we are using Top2Vec for our topic modelling. Automatic Topic labelling intends to reduce the effort to interpret while investigating the topics. In this study, we introduce a novel three-phase zero-shot topic labelling framework using the ConceptNet knowledge graph (a freely-available semantic network of words and phrases) and language models as external sources of information. The first phase uses the knowledge graph by extending the top n words (based on semantic similarity) neighbourhood and filling missing connections and information gaps by querying ConceptNet and generating a candidate sub-graph to generate candidate labels. In the second phase, it develops a neighbourhood graph for each candidate label, scores each node based on its semantic similarity with the topic and retains the best sub-graph based on semantic similarity. In the third phase, we utilize the language model to determine the labels using the final graph as input. We use a knowledge graph and language model to extend the knowledge beyond topic documents to optimize discovered topics with better representative terms while retaining the topic information. The proposed framework decreases the computation burden by utilizing a zero-shot approach and reduces the cognitive and interpretation load of the end-user by creating three types of labels for each topic, i.e., a one-word label, sentence label and summary label. The experimental results showed that our model significantly outperforms the unsupervised baselines and classic topic labelling models and is comparable to supervised baselines topic labelling models.

List of Abbreviations

Graph2Corpus	- Graph Nodes to Text Corpus Generation
KG	- Knowledge Graph
CKG	- ConceptNet Knowledge Graph
ML	- Machine Learning
KLD	- Kullback–Leibler divergence
NumberBatch Embedding	- ConceptNet Emmbeddings

Acknowledgements

First, I thank Dr. Evangelos E. Milios and Dr. Enayat Rajabi for supervising my master's research. Being your student offered me exceptional opportunities to learn and grow. Your support and friendly presence, especially during the difficulties of a pandemic, made this journey much easier.

I thank Baqir for your sincere friendship and support in this journey.

Finally, I thank Dr. Sageev Oore and Dr. Qiang Ye. Your courses and mentorship laid the foundations that enabled me to work on this thesis.

I would like to thank my committee members, who were generous with their expertise and precious time. A special thanks to Dr. Vlado Keselj and Dr. Hassan Sajjad.

Thank you, committee members, for agreeing to serve on my committee. I want to acknowledge and thank my school division for allowing me to conduct my research and providing any assistance requested.

Finally, thanks to the beginning teachers, mentor teachers and administrators in our school division who assisted me with this project. Their excitement and willingness to provide feedback made completing this research an enjoyable experience.

Chapter 1

Introduction

Topic modelling [11, 5, 24, 19, 24] is a significant technique in natural language processing (NLP) tasks, such as information retrieval and text mining [6]. The topics discovered are usually represented by a list of terms with a probability score. If the users want to understand the meaning of a topic discovered, they must carefully check the topic term list ranked in descending order of probability [22].

This topic form is often sufficient when the output of the topic model is used as input to another task, such as query expansion. But it may not be when the model output is presented to a user, such as within an exploratory search system. Even if word distribution makes intuitive sense, it is still challenging to understand what a topic means and the difference between topics. Therefore, the cognitive overhead in interpreting the topic-discovered terms could be high, especially when users need more background knowledge in the field of topics.

An easy solution would be manually labelling the topics; manually labelled topics are more interpretive and understandable. But it has several issues, such as manually labelling topics requiring considerable human efforts to review a large amount of text. There tends to be subconsciously influenced by the subjective options of the person doing the labelling [25].

For example, the topic with top terms such as “pain,“ “disorder,“ “symptom,“ “depression,“ “anxiety,“ “patient,“ “chronic,“ “depressive,“ “study,“ “psychiatric.“ There are two ways to understand this topic: we carefully read all the top words and try to guess the overall coverage of the topic. Or we can read a few documents to get an idea. Both tasks can be time-consuming and difficult. So if we read the topic word or the document, it can take time to get a clear picture, depending on the content we want to read. And if the topic contains information about something we don’t understand or have a background, it becomes even more difficult. Instead, if we can automate this task and generate a label, we can save both the time and hassle

of interpreting the label. For this topic, all the topic words can be easily represented under the label "Mental Disorder," which signifies what the topic covers.

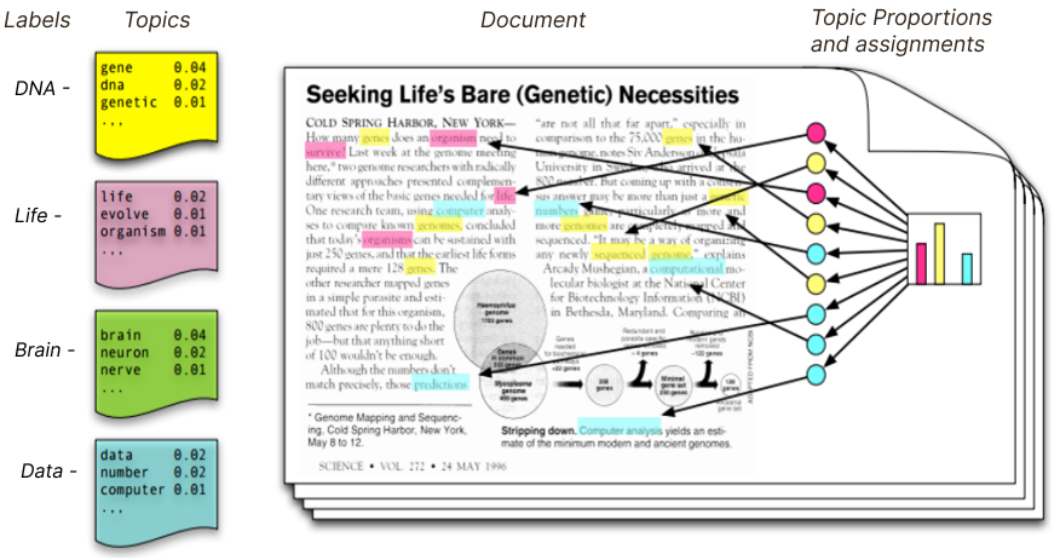

Figure 1.1: Topic Labelling gives meaningful names to the topics discovered in a corpus of documents. In this Figure, we see an example. Here we use the Top2Vec library to model topics, and our proposed algorithm labels them. Each topic contains the top n terms (based on cosine similarity scores) and document sub-collection. Labels are the assigned titles to each topic based on its content (documents and top n terms).

Consequently, researchers have explored and developed a range of techniques. Previous work on topic labelling has mainly focused on generating a word or keyword label, sentence or headline label, and summary labels for the topics.

Several past approaches, such as sequence-to-sequence neural networks, neural embeddings and re-ranking methods, were used to generate a word or keyword labels [4, 10, 3].

Researchers developed several approaches such as pre-trained neural embeddings, sentence embeddings, and centroid and large language models were used to generate sentence labels [17, 23, 36, 11].

Researchers have generated summary labels and developed several past approaches such as supervised deep learning models, language models such as BERT, connectivity matrix-based summary generation, and text-to-summary language models to generate a summary as a label [9, 16, 14, 38].

However, all the prevailing methods rely only on handcrafted features. There are two common pitfalls in the topic labelling task. First, exact word matching suggests that different words must have different meanings, which may damage the accuracy of the topic modelling. Secondly, capturing the real meanings of words or sentences with contextual information is more manageable. Third, they cannot fill the information gaps in data such as social media.

To address this problem, researchers have explored approaches that utilize external knowledge to improve the generalization of labels, capture the word's real meaning, and fill the information gaps [3, 13, 15, 1, 40, 27].

Several approaches were exploited, such as using ontology concepts instead of words alone to generate topic labels [40], and using graph centrality measure to extract the concepts that best represent the topics from the knowledge graph [20].

Despite their success, all the above approaches have two major drawbacks. One, they only generate labels of one specific kind, such as word, sentence, or summary labels, which can be useful for one dataset but may not be helpful for others. Two approaches that utilize external knowledge for better generalization and to fill information gaps are limited to word labels and can't generate sentence or summary labels as needed.

Why do we need all three types of labels? To explain the need for word labels, suppose we have data consisting of social media comments such as Facebook posts and WhatsApp messages, and we want to label the topic. For this example, we have a topic with comments such as participating in polls, voting for republicans, etc. Then, if we try to create a sentence or summary label, it will not be possible because there is insufficient information. But this can easily fit under a word label and give us a theme of the topic. Such as the Election in this example. Compared to the gibberish in the summary and sentence labels, this is very helpful in explaining the topic or at least giving an idea about the theme of the case.

To explain the need for a sentence label, suppose we have data that consists of news articles, and we modelled a topic with news article data; here, it says, "A 16-year-old student at school killed one teacher and so on". In this example, a word label will not be helpful because it gives the only theme of the article and it has limited use. But there is enough information to generate a sentence or a summary. If we

see the sentence label gives an idea of the article, like a headline. And users can go through the summary if they need more information.

Finally, to explain the need for a summary label, we have a topic with a news article with two different pieces of information. The first part of this article talks about ISRO launching an explorer to mars. And second, ISRO built the cheapest explorer in the least amount of time. The complete information can't be conveyed with word or sentence labels. But a summary label can easily make this enormous article digestible and maintain all the information from both parts of the paper. As we can see, the generated summary contains both launch and builds information from the article.

Together, these three labels create a very informative labelling system for the user, which can handle different data and allow the user to analyze the information as per their need. If they want to know the theme, then they can only look at the word label, but if they're going to get a clear picture, then they can choose to read the sentence label, and if they want to get even more details then they have a summary label available.

We introduce a novel three-phase zero-shot topic labelling framework that combines ConceptNet [33] (a freely-available semantic network of words and phrases) and deep language models to generate appropriate semantically correct labels. Our approach can generate all three labels for most datasets while ensuring the label is semantically correct and sound.

In the first phase, we apply a candidate label creation algorithm. Gathering the appropriate candidate labels is the most crucial task in any topic labelling approach. To do this, we take the top n words (based on cosine similarity scores) outputted from the topic modelling algorithm and then enrich the list by generating a neighbourhood for each word in the list by querying ConceptNet. We then prune it by retaining the most significant (based on defined selection criteria) fully connected graph by combining the neighbourhood of all words, pruning them by removing all unnecessary nodes based on the pruning criteria, and keeping a final fixed set.

We calculate the quality and assess each candidate label in the second phase. We exploited ConceptNet [32] and neighbourhood generation method [15]. We created a method that generates a neighbourhood graph for each candidate label by querying

ConceptNet again and calculating the document overlap score for each node in the graph. To make this method more stable, we added our matrices, i.e., three graph centrality, semantic similarity, and scoring approaches to select the most semantically meaningful nodes in the graph and defined a scoring scheme.

In the third phase of this approach, we utilize the language model to select the most appropriate label from the final graph created in the second phase. Here, we are making three labels, i.e., word, sentence, and summary.

Generated labels are per the topic's domain and ensure labels are semantically correct, explainable, and meaningful.

With every word being a node in the knowledge graph, it is straightforward to justify the similarity between words in the topic and their assigned label using cosine similarity. Explaining the similarity is almost impossible for other distributional word embeddings as they are built on the statistical aggregations of large volumes of textual data.

Extensive experiments have been conducted on datasets topics_bhatia, and AP News [4, 17, 18]. The results show that ZeTL can generate all three labels (One Word, Sentence, and Summary) semantically correctly and efficiently. Besides, it significantly outperforms unsupervised and classic labelling techniques [31, 14] and performs on par with state-of-the-art models [4, 18, 16].

The summary of our contribution follows:

1. We propose a novel three-phase zero-shot topic labelling framework that uses external knowledge (ConceptNet) and a language model along with the original underlying text assigned to the topic and provides the means to exploit its potential fully.

2. To the best of our knowledge, we are the first to calculate all three types of labels using our approach, i.e., Single Worded/Two Worded label, Short Sentence Label, and Short Summary Label for the modelled topics.

3. We are the first to utilize the blend of knowledge graph and language model for topic labelling tasks.

4. We did a quantitative results comparison and analysis of the capabilities of proposed topic labelling on multiple test datasets.

The remainder of this thesis is structured as follows: we present related work for topic labelling, emphasizing methods for all three labelling approaches, i.e., one-word, sentence, and summary labels that use external semantic knowledge and transformer models (Related Work). In the Methodology, we present our proposed method, Top2Label (Explainable topic Labelling using a knowledge graph). We describe our approach to topic labelling. We have defined different baselines with multiple benchmark data sets. Furthermore, we conclude.

Chapter 2

Related Work

Topic modelling [11, 5, 24, 19, 24] is a popular approach to finding hot spots and tracking the event development trends in a corpus [17]. The topics discovered are usually represented by a list of terms with a marginal probability. Several works [36, 18, 10] in the past have been done to help users understand discovered topics clearly and consider topic labelling as a solution. The topics represented by a set of words must be labelled to reduce the cognitive overhead from users to understand the underlying meaning of topics discovered.

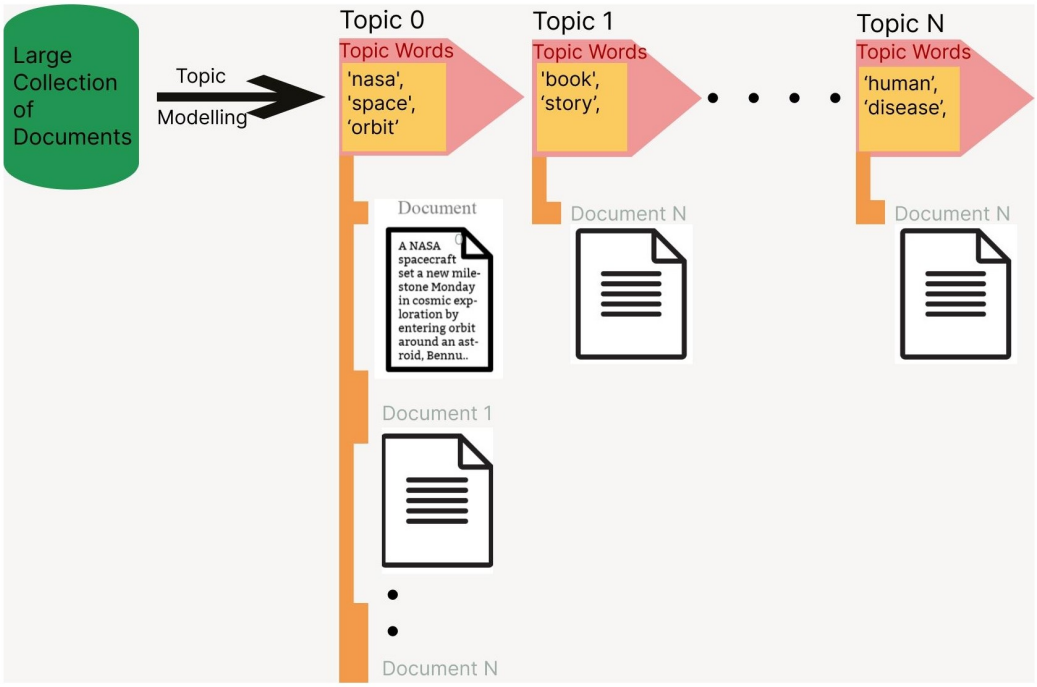

Figure 2.1: The Top2Vec topic modelling algorithm takes a collection of the document as input and returns n topics. Each Topic contains top n words (based on cosine similarity scores) and topic documents (most semantically similar documents with cosine similarity scores) assigned to it.

A significant part of the approaches [4, 35, 2, 17, 16] extract the most likely

label from the text. Researchers studied the problem of labelling sentiment-bearing topics [9] and introduced a method to label them with descriptive sentences; the method outperformed others and promoted the explanation and interpretation of the discovered topics.

To label discovered topics with higher Relevance, Coverage, and Discrimination, a novel two-stage textual summary framework [35] was created (candidate sentence selection and summary generation). The framework was based on submodular optimization to generate a fixed-length summary involving the top-score sentences for each discovered topic.

A graph-based automatic summarisation method (GRAPHSUM) was introduced [8], which represents the node combinations of two or more terms. The algorithm learned and exploited a specific strategy and used a PageRank algorithm to choose significant sentences for topic labelling tasks.

Researchers proposed the first seq2seq topic labelling model to produce appropriate textual labels for discovered topics and used BERT Score to measure the generated topic labels [11, 4]. Although the experimental results were efficient and concise, they might still be insufficient to express rich topics.

Another technique that researchers exploited was automatic topic labelling with an attention-based model paired with a pre-trained deep neural network [17, 16].

Researchers exploited several approaches to generate a word or keyword labels.

Researchers exploited a sequence-to-sequence neural network to generate labels [4]. The model is trained over a new large synthetic dataset created using distant supervision.

An approach is a neural embedding approach that uses Wikipedia document titles as label candidates and computes neural embeddings for documents and words to select the most relevant labels for topics [10].

Another researcher extracted candidate labels from a large pool (e.g., Wikipedia article titles) and then re-ranked based on their semantic similarity to the topic terms [3].

Researchers exploited several approaches to generate sentence labels.

Researchers proposed an approach that extracts candidate sentences from the topic using pre-trained neural embeddings and ranks them based on similarity [17].

Another approach exploited is relevant sentences are selected based on sentence embeddings and centroid measure [23].

Researchers exploited several approaches to generate summary labels.

Using deep learning models to generate a summary by training them on specialized datasets [9].

Extracting candidate sentences and using language models such as BERT to generate summaries [16].

A connectivity matrix based on intra-sentence cosine similarity as the adjacency matrix of the graph representation of sentences to generate a summary [14].

Extracting summaries based on rank fusion [21].

Another popular approach is utilizing language models designed to generate text summaries [38].

A significant drawback in all these approaches is that they rely on the assumptions that (1) The correct label can be found in the documents and (2) the corpus is rich enough to identify a label with confidence. However, this is only sometimes the case. For example, a cluster of documents might be about artificial intelligence without mentioning the phrase. On the other hand, it might contain many more specialized phrases that cannot be related just based on the text (e.g., probabilistic reasoning and first-order logic). External data sources can help overcome this problem. Besides the work at hand, this idea motivates a wide range of recent research [17, 16, 20, 40, 3, 13].

Several techniques have been implemented to resolve this issue. The topic labelling method selects the most relevant labels (semantically correct phrases) for discovered topics by computing neural embedding of documents and words. They trained a Doc2vec model on the English Wikipedia corpus to generate sentence and word embedding (Word2vec generated the latter during the internal training process). Finally, compared with the competitor methods, their model achieved the best results across many domains [10].

Another automatic topic labelling approach is to exploit structured data such as DBpedia2 [37]. Given a topic, they find the terms with the highest marginal probabilities and then determine a set of DBpedia [7] concepts. Each concept represents the identified sense of one of the top terms of the topic. After that, they create a graph out of the concepts and use graph centrality algorithms to identify the most

representative concepts for the topic [20].

In more recent work, a topic model that integrates knowledge with data-driven topics within a single general framework [3]. Prior works primarily emphasize the topics discovered from the LDA topic model, whereas, in this model, they introduce another random variable, namely the concept between topics and words. In this case, each document comprises topics where each topic is defined as a probability distribution over concepts, and each concept has a multinomial distribution over vocabulary [3].

Another topic-labelling approach using an external data source [15]. In this approach, the researchers [15] leveraged ConceptNet to produce a list of candidate words related to the labels of our interest. Then generate a "topic neighbourhood" for each label containing all the semantically related concepts/nodes, and compute a score for each label based on the document content. Moreover, select labels from candidates based on the highest cosine similarity; one significant difference is that this approach is for document classification and not for topic labelling, as they pre-required to choose from a set of labels.

Our approach differs from all the above works from four perspectives:

1. First, All the above works focus either on single or double-worded labels [4, 17, 16, 20, 18], a short sentence [10, 9] or a summary [35, 12, 16] of topics as labels, but sometimes it is not possible to get the exact meaning out from just one of these. For example, a summary would not be helpful if a document has concise positive or negative reviews. Moreover, similarly, if we are modelling articles about different topics such as space but with various internal topics, one is talking about satellite launch and maintenance. Furthermore, the other talking about building spaceships, then one or two worded labels will not be able to give the full context of the underlying topic and requires a summary for clarity to address this situation. As a resolution for this, we are generating all three together, i.e., One Word, Sentence and Summary Labels.

2. Second, the above approaches either use pretraining to train a network or heavily depend just on external knowledge. At the same time, our solution differs as we do not train models specific to one data type or rely only on external knowledge.

We generate a concise knowledge graph from the modelled data to consider labels within the original document and its neighbourhood in the knowledge graph.

3. We propose a novel three-phase zero-shot topic labelling framework which uses external knowledge (ConceptNet) and language model with the original underlying text assigned to the topic and provides the means to exploit its potential fully.

4. Quantitative results comparison and qualitative analysis of the capabilities of proposed topic labelling approach with the human labelled dataset.

Chapter 3

Methodology

This section presents the proposed work and formally defines the problem in the focus of this work. An overview is shown in Fig. 3.1.

We also introduce the primary notation and terms used throughout the paper.

We present a novel three-phase zero-shot topic labelling framework using ConceptNet [33]. The approach requires the output of the Top2Vec [5] topic modelling algorithm as input to generate all the labels.

In the first phase, we select the candidate labels by querying ConceptNet, generating a neighbourhood graph for all the top n terms (based on cosine similarity scores), and extending it to get candidate labels.

In the second phase, we query ConceptNet and generate the neighbourhood graph for each label and calculate document overlap scores for each node in the graph and retain the final graph based on the defined pruning and validation scheme.

In the third phase, we use our novel selection algorithm to select the most appropriate one-word and use a language model along with the final graph for sentence and summary labels for each topic.

The topic extraction applies the Top2Vec topic model [5] to extract topics from a corpus of documents. Each topic gives a set of top n words (based on cosine similarity scores) with a semantic similarity score of the word with a sub-collection of documents.

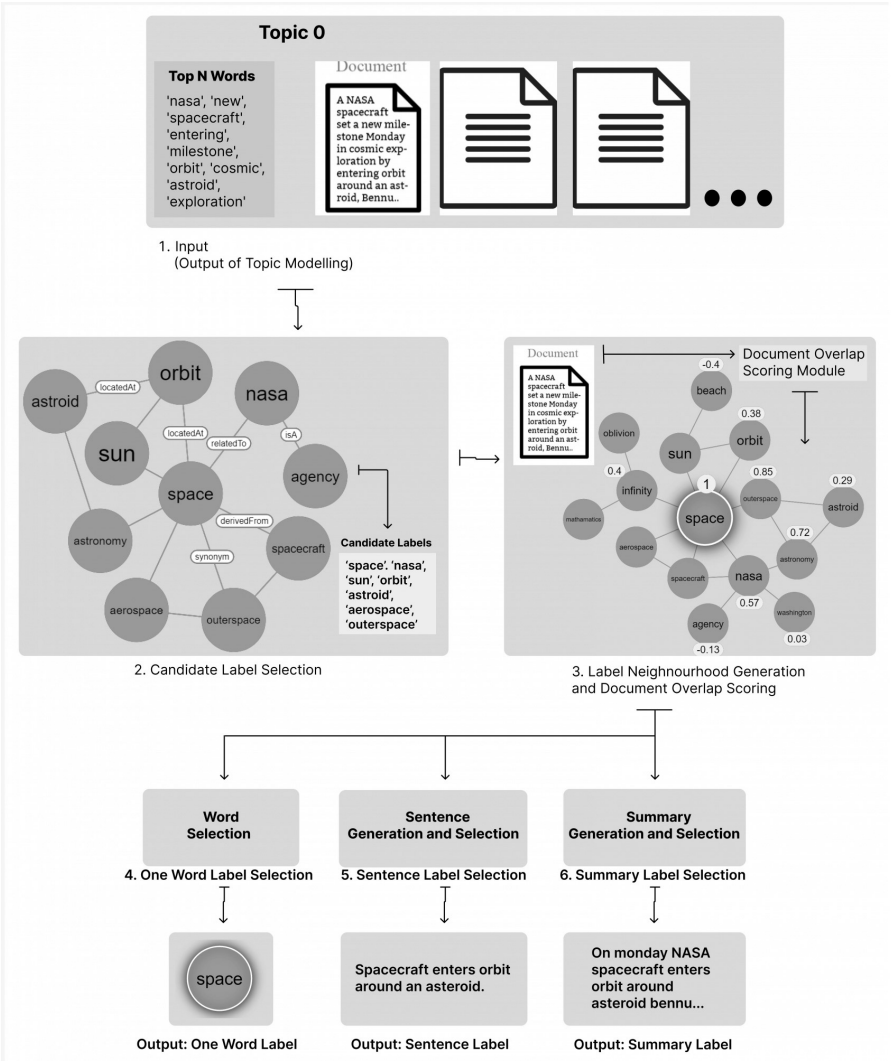

Figure 3.1: The proposed approach for creating labels: **1. Input:** Topics where each topic contains top words and the documents (based on cosine similarity scores). **2. Candidate Label Selection** Generate a list of candidate labels by creating a sub-knowledge graph with top n words by querying ConceptNet [33], details in section 3.3. **3. Neighbourhood generation and Scoring:** Create another sub-knowledge graph for each candidate label and calculate the overlap score for each node with the topic. **4. One Word Label Selection:** Outputs the one-word label based on the similarity score, details in section 3.3.1. **5. Sentence Label Selection:** Outputs the sentence label using sentence corpus generated using the proposed Graph2Corpus algorithm, passing it to the language model [38], details in section 3.3.2. **6. Summary Label Selection:** Outputs the summary label by utilizing the generated corpus and topic documents and passing them to abstract summary language model [38] to create summaries details in section 3.3.3.

Algorithm 1: Label Creation

Input: $Topics \geq 0$ $ConeptNetLookup()$ $tripletToText()$ $summarizer()$ n #Threshold to select triplets

Output: Summary Label, Sentence Label, Word Label

```
for Topic in Topics do
    # Select candidate label CandLabels = CandLabelSelection(Topic);
        # Neighbourhood graph generation and calculation of document
    overlap scores. NGraphWithDocScore = NGraphOverlapScore(Topic,
    CandLabel);
        # Word label selection WordLabel =
    WordLabelSelection(NGraphWithDocScore);
        # Sentence label generation and selection SentenceLabel =
    SentenceLabelSelection(NGraphWithDocScore);
        # Summary label generation and selection SummaryLabel =
    SummaryLabelSelection(NGraphWithDocScore);
end

```

In this approach, we will generate three labels, 1. One Worded Label, 2. Sentence Label, 3. Summary Label.

3.1 ConceptNet

To overcome information gaps and better generalize on topics, we use ConceptNet [32] knowledge graph as a source of external knowledge.

ConceptNet is a freely-available semantic network designed to help computers understand the meanings of words that people use.

The famous source of external knowledge is DBpedia [7], but the main difference between DBpedia and ConceptNet is that DBpedia is an encyclopedia with all the information it could have and ConceptNet is a knowledge base.

ConceptNet originated from the crowdsourcing project Open Mind Common Sense, launched in 1999 at the MIT Media Lab. It has since grown to include knowledge from other crowdsourced resources, expert-created resources, and games with a purpose.

ConceptNet also provides word embeddings called Numberbatch embeddings [34]. ConceptNet is used to create word embeddings – representations of word meanings as vectors, similar to word2vec, GloVe, or fastText, but better.

These word embeddings are free, multilingual, aligned across languages, and designed to avoid representing harmful stereotypes. Their performance at word similarity, within and across languages, was shown to be state-of-the-art at SemEval 2017 [34].

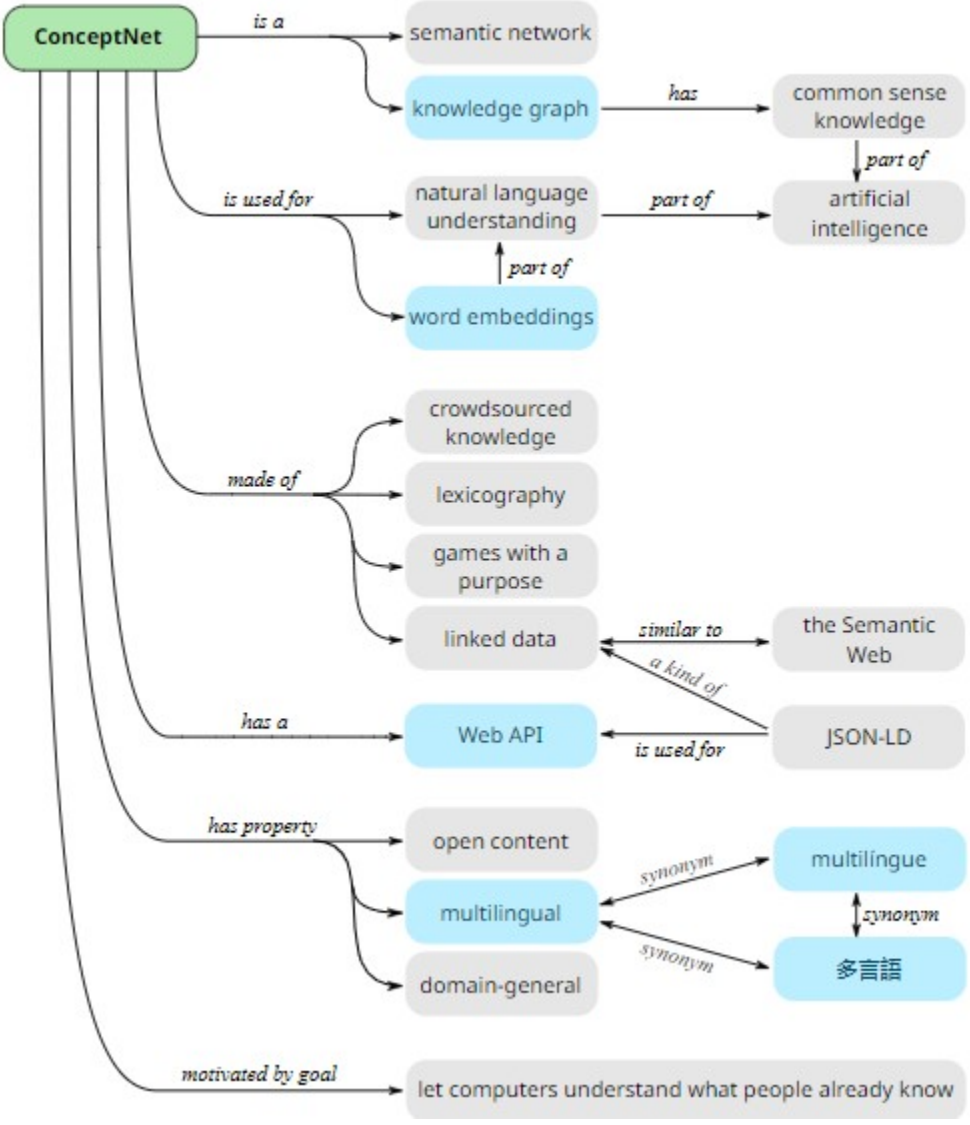

Figure 3.2: The ConceptNet [32] Knowledge Graph

3.2 Graph2Corpus

We propose Graph2Corpus (Text Generation), a novel algorithm to generate text corpus from knowledge graph triplets (edges) to introduce external knowledge into the topic data for language model inputs.

We start by searching all the nodes (Triplets) of the graph in the KELM Corpus [1]. KELM Copus is a dataset created from Wikidata Knowledge Graph by creating a sentence for each DBpedia triplet. If we find any sentence in the KELM corpus, we check their relevance score with the topic document based on the threshold and select them. We add the graph nodes directly as keywords if no sentence is determined. And with this, we also choose all the sentences from the topic document with graph nodes and create a combined sentence corpus.

KELM Corpus is a synthetic corpus that consists of the entire Wikidata Knowledge Graph as natural text sentences. It has 15M sentences synthetically generated using a T5 model fine-tuned on the data from Part 1 with some additional components. It can be used as additional data in language model pre-training to integrate KGs with natural text [1].

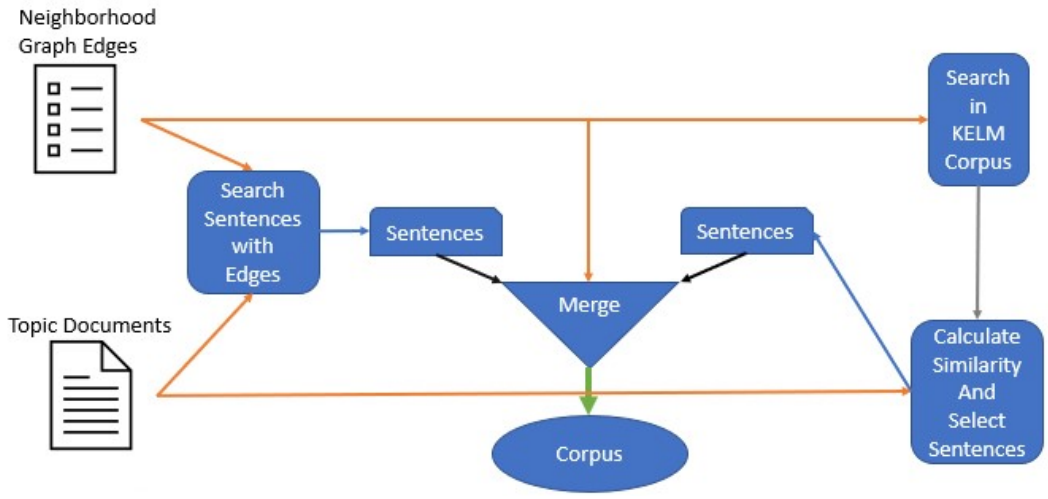

Figure 3.3: TEXTGEN: Novel Text Generation algorithm.
This algorithm generates text corpus from knowledge graph triplets (edges) and makes the inclusion of external knowledge in topic labelling simple and fast.

3.3 Candidate Label

Overall the labelling task is divided into three phases, and the first phase is where we generate all the potential labels called candidate labels. So that in phases two and three, we can select and validate appropriate labels accordingly.

In this first phase, we will generate candidate labels 3.4 then we will utilize them to develop the final label by scoring and selecting the best among them.

The intuition behind our approach is that the concepts of a topic are related. They should lie close in the ConceptNet graph [33], and by expanding from each such concept for a few hops, all the topic concepts related to that term will ideally form one connected graph. The graph extraction phase uses this intuition to address the problem of finding label candidates.

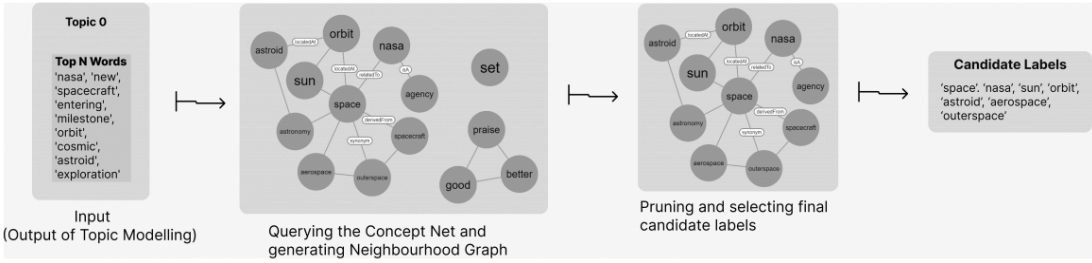

Figure 3.4: The zoomed-in version of part two, Candidate Label Selection, from Fig. 3.1 Approach Summary. We use the output of Top2Vec [5] by feeding topics as input, then querying the ConceptNet [33] for each top n word (based on cosine similarity) to generate a knowledge graph and extend it to n neighbour edges for each current edge. Then pruning is done by removing all nodes not part of the biggest fully connected sub-graph and word nodes with a cosine score less than average. Then finally most significant fully connected graph (based on cosine similarity and the biggest fully connected sub-graph) is retained as candidate labels and converted into lists for further processing.

To generate the candidate labels for each topic, we start by querying the ConceptNet [33] for each top n words (based on cosine similarity scores) of the topic; for each node, we select the directly connected word. The word neighbourhood graph is created by querying every node n hops away from the word node. Then we retain the most significant connected graph, assuming that all the topic concepts will ideally form one connected graph. Again, we perform second pruning by removing all the nodes in the top n words (based on cosine similarity scores) list and scoring a

Algorithm 2: Candidate Label Selection

Input: $Topics \geq 0$ $ConeptNetLookup()$ n # Number of hopes to extend neighbourhood

Output: List of Candidate Labels

```
while Topics − − ≥ 0 do
    TopNWords = Topic.TopNWords;
        for word in TopNWords do
        # Exreacting neighbourhood from ConceptNet for each word. # And
          extending the neighbourhood for n hopes from word
          Triplets = ConeptNetLookup(word, n);
            if Triplet ≠ null then
            # Store all the triplets add Triplet to ListOfTriplets ;
        end
    end
    # Selecting the biggest fully connected sub-graph based on cosine
      similarity after pruning. #And converting it to a list of candidate labels
      candidateList = graphPruning(ListOfTriplets) ;
    return candidateList
end
```

similarity of less than the average. We convert this neighbourhood graph to a list of terms and consider them candidate labels to generate the candidate set.

3.3.1 One Word Label

The one-word label is the first label we are generating out of the three types of labels we will be generating in this work. To generate a one-word label, we start by generating a label neighbourhood 3.5 for each candidate label from the candidate labels, and we query ConceptNet [33] for nodes directly connected to the label node. Although the assertions contain a finer granularity when referring to concepts, we only consider the root word for each concept to build the neighbourhood.

For example, the word “match” has multiple meanings. It connects to multiple

definition edges inside the ConceptNet graph: the tool to light a fire /en /match connects to /artifact, the event where two contenders meet to play /en /match connects to /event, and the concept of several things fitting together /en /match connects to /cognition. We map all these nodes (and others, such as the verb form) to the term "match." We then add (inverse) relations from the object to the subject for each triplet to ensure that every term in the graph has a neighbourhood. The label neighbourhood is created by querying every node that is n hops away from the label node. Given a score to every node based on the cosine similarity between the label and the node computed using ConceptNet Numberbatch [33] (ConceptNet's graph embeddings). This score represents the relevance of any term in the neighbourhood to the main label and would also allow us to refine the neighbourhood and produce a score. In the case of a label with multiple tokens (e.g., the topic "Arts, Culture, and Entertainment"), we take the union of all word components' neighbourhoods, weighted by the maximum similarity score if the same concept appears in the vicinity of multiple label components. The higher n is, the bigger the generated neighbourhoods become.

We thus propose multiple methods to vary the size of the neighbourhood:

1. Coverage: we vary the number of hops N.

2. Relation masking: we consider subsets of all possible relations between words from the ConceptNet knowledge graph. More precisely, we consider three cases:

   - The sole relation RelatedTo is the most frequent one in the graph.
   - The ten semantic and lexical similarity relations only, i.e., 'DefinedAs,' 'DerivedFrom,' 'HasA,' 'InstanceOf,' 'IsA,' 'PartOf,' 'RelatedTo,' 'SimilarTo,' 'Synonym,' 'Antonym';
   - The whole set of 47 relations is defined in ConceptNet [33].

3. Filtering: we filter out some nodes based on their similarity score:

   - Threshold (Thresh T): we only keep nodes in the neighbourhood if their similarity score to the label node is greater than a given threshold T.
   - Hard Cut (Top N): we only keep the top n (based on cosine similarity scores) nodes in the neighbourhood ranked by their similarity score.

- Soft Cut (Top P %): we only keep the top P% nodes in the neighbourhood, ranked on their similarity score.

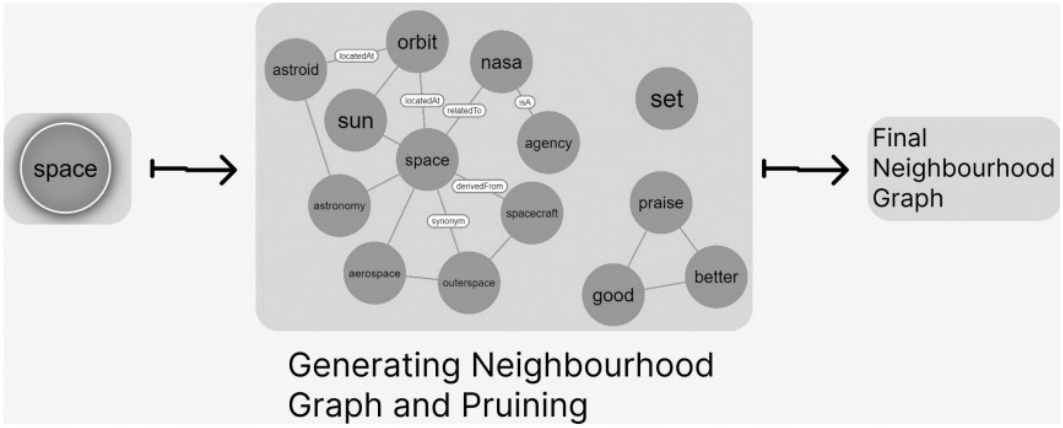

Figure 3.5: Zoomed in version of 3. Label Neighbourhood Generation from Fig. 3.1 Approach Summary. This module generates the neighbourhood graph for each candidate label we pass to it. We begin generating a neighbourhood graph by querying ConceptNet [33] and extend the current graph to n directly connected neighbour edges in series. We retain the biggest fully connected graph and give scores using cosine similarity between the label and all nodes. Then this graph is pruned by selecting the top n nodes (based on cosine similarity scores) with a score of more than a threshold.

Algorithm 3: Neighbourhood Generation

Input: $candidateLabelList \geq 0$ $ConeptNetLookup()$ n #number of edges to extend the graph

Output: Neighbourhood Graph

```
1  while candidateLabelList do
2  |  Triplet = ConeptNetLookup(candidateLabel);
3  |      if Triplet ≠ null then
4  |  |  add Triplet to ListOfTriplets ;
5  |  |
6  |  end
7  end
8  #Extend the neighbourhood by n edges
   ListOfTriplets = extendNeighbourhood(ListOfTriplets, n);
9      # Extracting the biggest fully connected graph after pruning
   NeighbourhoodGraph = graphPruning(ListOfTriplets) ;
10     # Calculating cosine similarity score between each node with
   candidate label NeighbourhoodGraph =
   cosineSimilarity(NeighbourhoodGraph, candidateLabel) ;
11
```

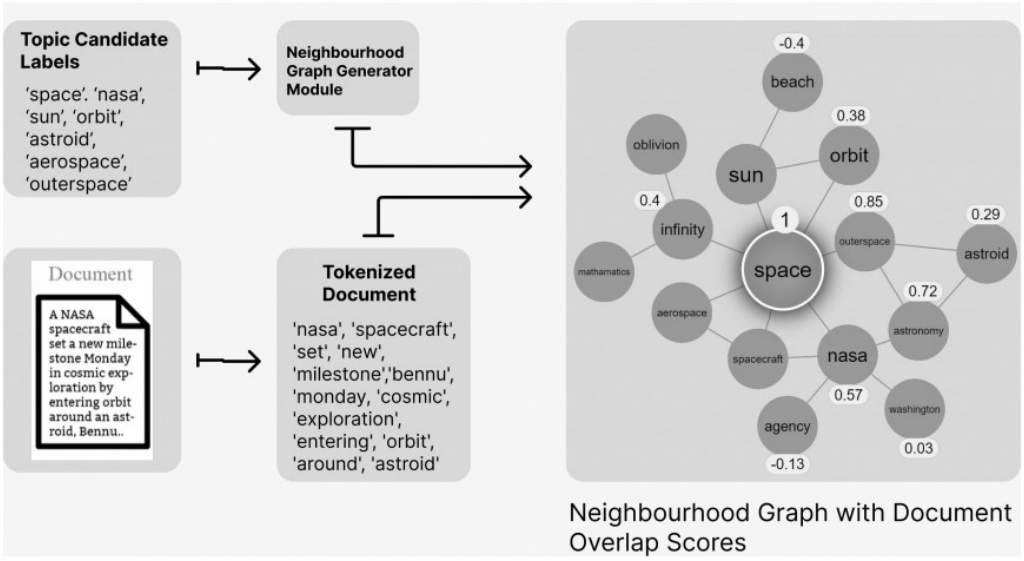

Figure 3.6: Zoomed in version of 3. Document Overlap Scoring from Fig. 3.1 Approach Summary.This module calculates the overlap scores for graph nodes. Once Neighbourhood Generator Module Fig. 3.5 decides on the neighbourhood graph. Then we use the topic documents by tokenizing them and use Numberbatch Embeddings (ConceptNet Embedding) [33] to calculate the overlap score between each node and document. Then outputs the final knowledge graph with the overlap scores.

Algorithm 4: Document Overlap Scoring

Input: $NeighbourhoodGraph \geq 0 \qquad Topic$

Output: Neighbourhood Graph with Document Overlap Score

#Generating embeddings of original doc from topic using Numberbatch Embedding (ConceptNet Embedding)

$docEmmbeding = ConceptNetEmbedding(Topic.Documents)$;

while $NeighbourhoodGraph$ **do**

  for $edge$ *in* $NeighbourhoodGraph.Edges$ **do**

    if $edge \neq null$ **then**

      $NeighbourhoodGraph[edge].OverlapScore = calculateOverlap(docEmmbeding, edge)$;

    end

  end

end

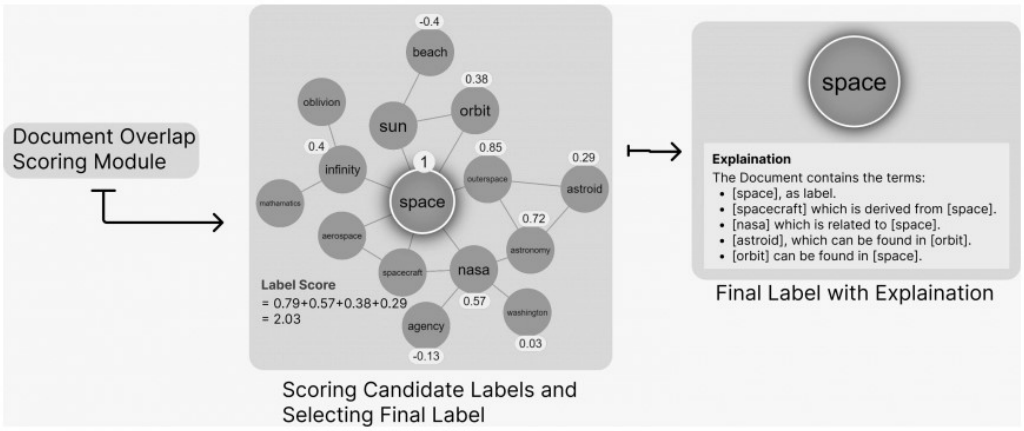

Figure 3.7: Zoomed in version of 4. One Word Label Selection from Fig. 3.1 Approach Summary. Generates one worded label, we use the final graph with overlap scores Fig. 3.6 for each candidate label. For each node in the graph, we calculate the score based on our proposed scoring scheme and select the label with the highest score as the final one-word label.

Algorithm 5: One Word Label Selection

Input: $NeighbourhoodGraphWithDocOverlapScore \geq 0$

Output: One Word Label

```
1  if NeighbourhoodGraphWithDocOverlapScore ≠ null then
2      LabelWithMaxScore = "";
3      CurrentScore = 0;
4          for edge in NeighbourhoodGraphWithDocOverlapScore.Edges
       do
5          if wdge.Score ≥ CurrentScore then
6              CurrentScore = Edge.Score;
7              LabelWithMaxScore = Edge.Value;
8          end
9      end
10 end
```

Scoring a Label

Once the neighbourhood is generated, we can predict the document label by quantifying the overlap between the document content (as broken down to a list of tokens)

shown in Fig. 3.5. The label neighbourhood nodes, denoted in the following equations as $doc \cap LN(label)$ as defined by Harrando et al. [15]. We consider the following scoring schemes:

1. Counting: assigning the document with the highest overlap count between its terms and the topic neighbourhood.

$$count\_score(doc, label) = |doc \cap LN(label)|$$

2. Distance: In the graph, the distance between the term in the document and the label (number of nodes or path length between the token node and the label): the further a term is from the label vicinity, the lower its contribution to the score.

$$dis\_Score(doc, label) = \sum_{token \in doc \cap LN(label)} 1/min\_path\_length(token, label) + 1$$

3. Degree: each node's score is computed using the number of incoming edges to it, reflecting its importance in the topic graph (we use $f(n) = log(1 + nedges)$ to amortize nodes with a very high degree). $degree\_score(doc, label) = f(node\_degree(token))token \in doc \cap LN(label)$

4. Number batch similarity: for each term in the document included in the label neighbourhood, we increase the score by its similarity to the label embedding (we denote the NumberBatch concept embedding for word w by nbw).

   $numberbatch\_score(doc, label) = sim(nbtoken, nblabel)token \in doc \cap LN(label)$

5. Word Embedding similarity: like the Numberbatch similarity, but we use pre-trained 300-dimensional GloVe [16] word embedding instead to measure the word similarity (we denote the GloVe word embedding for word w by glove).

   $glove\_score(doc, label) = sim(glovetoken, glovelabel)token \epsilon doc \cap LN(label)$

6. Centrality measures [30] is used to identify nodes (or actors) that are most important (and thus, central) for the network, an objective in line with our requirements. Different criteria for importance, suitable for other purposes and scenarios, led to a range of centrality measures proposed in the literature [30, 28]. Two of the most popular ones are:

(a) Closeness centrality: a node is essential if it lies close to all the other nodes in the network. In the context of topics, nodes with high closeness centrality indicate concepts closely related to all other concepts of the topic graph [28].

(b) Between-ness centrality: a node is crucial if it facilitates the flow of information between other nodes in the graph. In a semantic network, nodes with high between-ness centrality are the nodes that establish fast connections between the different nodes in the graph [28].

The model is thus the neighbourhood set for each candidate label coupled with a scoring scheme.

Explainability

Given the label neighbourhood, we can explain why a topic has been given a specific label. This explanation can be generated in natural language. It can be seen in the sub-graph of ConceptNet that connects the label node with every word in the document that appears within its neighbourhood and counts towards its score. Although the "RelatedTo" edge does not offer much explanation beyond semantic relatedness, its explicit presence in ConceptNet confirms this relatedness beyond any non-explicit measure (e.g., word embedding similarity). Since this graph is usually quite big, we can generate a more manageable summary by picking up the closest n terms to the label in the graph embedding space. They constitute the nodes contributing most to the document's score. We can show one path (for instance, the shortest) between the top term nodes and the label node. The paths can then be verbalized in natural language.

For example, we select the final label as Space. We can explain based on the topic document containing the words spacecraft, NASA, astroid, and orbit, this can be further explained using the final graph path, and we can build relation spacecraft derived from space, NASA, which is related to space, astroid which can be found in orbit, and orbit can be found in space (RelatedTo and IsA are two relations from ConceptNet) as shown in Fig. 3.7.

3.3.2 Sentence Label

The sentence label is the second label we are generating from the three types of labels we will generate in this work. To generate the sentence label, we start by taking all candidate labels for each topic, we begin by querying the ConceptNet [33] for each label, and we generate a label neighbourhood 3.5; we query ConceptNet [33] for nodes that are directly connected to the label node. Although the assertions contain a finer granularity when referring to concepts, we only consider the root word for each concept to build the neighbourhood. The label neighbourhood is created by querying every node that is n hops away from the label node. Every node is then given a score based on the cosine similarity between the label and the node computed using the ConceptNet Number batch (ConceptNet's graph embedding). This score represents the relevance of any term in the neighbourhood to the main label and would also allow us to refine the neighbourhood and produce a score. In the case of a label with multiple tokens (e.g., the topic "Arts, Culture, and Entertainment"), we take the union of all word components' neighbourhoods, weighted by the maximum similarity score if the same concept appears in the vicinity of multiple label components. The higher n is, the bigger the generated neighbourhoods become.

We thus propose multiple methods to vary the size of the neighbourhood:

1. Coverage: we vary the number of hops N.
2. Relation masking: we consider subsets of all possible relations between words from the ConceptNet knowledge graph. More precisely, we consider three cases:
    - The sole relation RelatedTo is the most frequent one in the graph.
    - The ten semantic and lexical similarity relations only, i.e., 'DefinedAs,' 'DerivedFrom,' 'HasA,' 'InstanceOf,' 'IsA,' 'PartOf,' 'RelatedTo,' 'SimilarTo,' 'Synonym,' 'Antonym';
    - The whole set of 47 relations is defined in ConceptNet.
3. Filtering: we filter out some nodes based on their similarity score:
    - Threshold (Thresh T): we only keep nodes in the neighbourhood if their similarity score to the label node is more significant than a given threshold T.

- Hard Cut (Top N): we only keep the top n (based on cosine similarity scores) nodes in the neighbourhood ranked by their similarity score.
- Soft Cut (Top P %): we only keep the top P% nodes in the neighbourhood, ranked on their similarity score.

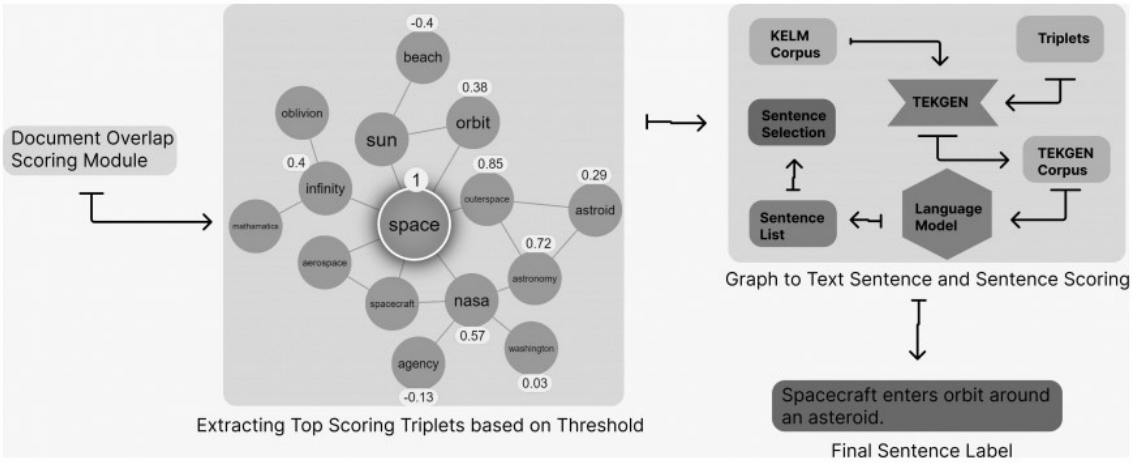

Figure 3.8: Zoomed in version of 5. Sentence Label Selection from Fig. 3.1 Approach Summary. This module generates the sentence label by utilizing the final graph from Fig. 3.6 and passing it to Graph2Corpus [1] (Triplet to Text Generation) algorithm to generate text corpus. Then we use this corpus and pass it to language model [38] to generate a sentence. Finally, we use our proposed scoring scheme to score the sentence and select the sentence with the highest score as the final sentence label.

Algorithm 6: Sentence Generation

Input: $NeighbourhoodGraphWithDocOverlapScore \geq 0$ n # Threshold to select triplets

Output: Sentence

```
if NeighbourhoodGraphWithDocOverlapScore ≠ null then
    for edge in NeighbourhoodGraphWithDocOverlapScore.Edges do
        if edge.score ≥ n then
            # Searching KELM Corpus for sentences which contains the Edge (Graph Node) and selecting the most relevant using ConceptNet Emmbedings sentence = searchEdgeInKELMCorpus(Edge) if sentence ≠ null then
                SentenceList.add(sentence)
            end
            if sentence == null then
                sentence = searchSentenceWithEdgeInTopic(Edge) if sentence ≠ null then
                    SentenceList.add(sentence)
                end
                if sentence == null then
                    SentenceList.add(edge)
                end
            end
        end
    end
end
# Generating the sentence using the list of sentences created using the Abstract text summary model, which generates headlines. # Thus, only creating a sentence sentence = generateSentence(SentenceList)
```

Algorithm 7: Sentence Label Selection

Input: $CandidateLabelsList \geq 0$

Output: Sentence Label

```
if CandidateLabelsList ≠ null then
    for candidate in CandidateLabelsList do
        # Generating sentence from each candidate label using
        Sentence Generation Algorithm
        sentenceList = SentenceGeneration(candidate)
    end
end
# Selecting Sentence using the Scoring scheme
SentenceLabel = sentenceWithMaxScore(sentenceList)
```

After finalizing the graph, we pass these graph nodes to Graph2Corpus 3.8, a novel algorithm which creates a sentence corpus. Then we pass this corpus to our language model, the abstract headline summary creation language model [38], which creates a list of headlines, and then finally, we score all the sentences (headlines) using the scoring scheme defined below to select the final sentence label.

Scoring a Label:

Once the neighbourhood is generated, we can predict the document label by quantifying the overlap between the document content (broken down into a list of tokens) and the sentence label. We consider the following scoring schemes:

1. Overlap: assigning the score to the label with the highest overlap between the label and the documents using ConceptNet Embedding.

2. Cosine Similarity: Calculating cosine similarity score between documents and the generated label.

3. Sentence Similarity Score: Sentence Similarity is the task of determining how similar two texts are. Sentence similarity models convert input texts into vectors (embedding) that capture semantic information and calculate how close (similar) they are between them. This task is beneficial for information retrieval and clustering/grouping.

3.3.3 Summary Label

The summary label is the third and final label we are generating from the three types of labels we will generate in this work. To generate a summary label, we start by taking candidate labels 3.4, and for each candidate label, Fig. 3.4, we will create a label neighbourhood; we query ConceptNet [33] for nodes directly connected to the label node. Although the assertions contain a finer granularity when referring to concepts, we only consider the root word for each concept to build the neighbourhood. The label neighbourhood is created by querying every node that is n hops away from the label node. Every node is then given a score based on the cosine similarity between the label and the node computed using ConceptNet Number-batch (ConceptNet's graph embedding). This score represents the relevance of any term in the neighbourhood to the main label and would also allow us to refine the neighbourhood and produce a score. In the case of a label with multiple tokens (e.g., the topic "Arts, Culture, and Entertainment"), we take the union of all word components' neighbourhoods, weighted by the maximum similarity score if the same concept appears in the vicinity of multiple label components. The higher n is, the bigger the generated neighbourhoods become.

We thus propose multiple methods to vary the size of the neighbourhood:

1. Coverage: we vary the number of hops N.
2. Relation masking: we consider subsets of all possible relations between words from the ConceptNet knowledge graph. More precisely, we consider three cases:
    - The sole relation RelatedTo is the most frequent one in the graph.
    - The ten semantic and lexical similarity relations only, i.e., 'DefinedAs,' 'DerivedFrom,' 'HasA,' 'InstanceOf,' 'IsA,' 'PartOf,' 'RelatedTo,' 'SimilarTo,' 'Synonym,' 'Antonym';
    - The whole set of 47 relations is defined in ConceptNet.
3. Filtering: we filter out some nodes based on their similarity score:
    - Threshold (Thresh T): we only keep nodes in the neighbourhood if their similarity score to the label node is more significant than a given threshold T.

- Hard Cut (Top N): we only keep the top n nodes in the neighbourhood ranked by their similarity score.
- Soft Cut (Top P %): we only keep the top P% nodes in the neighbourhood, ranked on their similarity score.

After finalizing the graph, we pass these graph triplets to Graph2Corpus 3.8, a novel algorithm that creates a corpus.

After getting the corpus, we combine it with the original documents of the topic and pass it to our language model, which is an abstract summarization model such as Pegasus [38] to generate summaries for each candidate label neighbourhood and select the most relevant summary label based on scoring below.

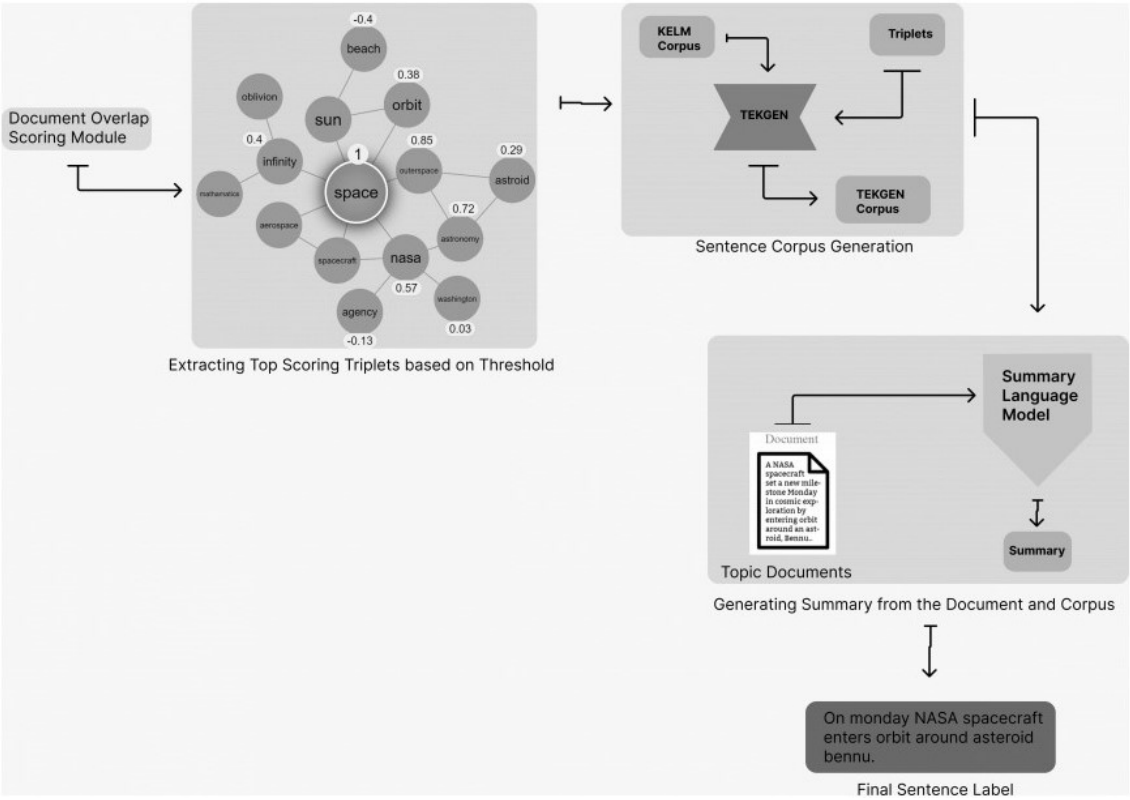

Figure 3.9: Zoomed in version of 6. Summary Label Selection from Fig. 3.1 Approach Summary. This module generates the summary label by utilizing final graph from Fig. 3.6. We pass the graph to Graph2Corpus (Text Generation) algorithm to generate the text corpus. The generated text corpus combined with original documents of the topic is passed through language model [38] (abstract summarize model) to generate the summaries. Finally, we use our proposed scoring scheme to score the summary and select the summary with the highest score as the final summary label.

Algorithm 8: Summary Generation

Input: $NeighbourhoodGraphWithDocOverlapScore \geq 0$
$summarizer()$ # Generate abstract summary $Topic$ n # Threshold for selection triplets

Output: Summary

```
1  if NeighbourhoodGraphWithDocOverlapScore ≠ null then
2      for edge in NeighbourhoodGraphWithDocOverlapScore.Edges do
3              if Edge.Score ≥ N then
4          # Searching KELM Corpus for sentences which contains the Edge
             (Graph Node) # And selecting most relevant using ConceptNet
             Emmbedings sentence = searchEdgeInKELMCorpus(Edge) if
             sentence ≠ null then
5              SentenceList.add(sentence)
6          end
7          if sentence == null then
8              sentence = searchSentenceWithEdgeInTopicDoc(Edge) if
                sentence ≠ null then
9                  SentenceList.add(sentence)
10             end
11             if sentence == null then
12                 SentenceList.add(Edge)
13             end
14         end
15     end
16 end
17 end
18 # Generating abstract summary
   summary = summarizer(Topic.Document, sentenceList) ;
```

Algorithm 9: Summary Label Selection

Input: $CandidateLabelsList$

Output: Summary Label

```
if CandidateLabelsList ≠ null then
    for label in CandidateLabelsList do
        summaryList.add(SummaryGeneration(label))
    end
end
# Selecting the summary with the highest score
SummaryLabel = summaryWithMaxScore(summaryList) ;
```

Scoring a Label:

Once the summary labels are generated, we can predict the document label by quantifying the overlap between the document content (broken down into a list of tokens) and the summary label. We consider the following scoring schemes:

1. Overlap: assigning the score to the label with the highest overlap between the label and the documents using ConceptNet Embedding.

2. Cosine Similarity: Calculating cosine similarity score between documents and the generated label.

3. Sentence Similarity Score: Sentence Similarity is the task of determining how similar two texts are. Sentence similarity models convert input texts into vectors (embedding) that capture semantic information and calculate how close (similar) they are between them. This task is handy for information retrieval and clustering/grouping.

Chapter 4

Experiments

This chapter presents experiments to select the best model (chapter 4.1). Next, we describe the datasets used to evaluate our approach (chapter 4.2.1, 4.3.1, 4.4.1) for all three types of labels (One Word, Sentence and Summary labels, respectively). We then detail the baselines we compare to our approach (chapter 4.2.1, 4.3.2, 4.4.2) before discussing our results (chapter 5).

4.1 Model Selection

This chapter evaluates some options regarding neighbourhood filtering and document scoring mentioned in Chapter 3. We use the BBC News dataset as a testbed for selecting and optimizing model parameters to choose the best-performing models. We report the results on the other datasets using the best parameters found at this stage. We first evaluate the different choices made to generate the candidate label neighbourhood, as discussed in Chapter 3 and reported in Table 4.1.

Finally, we evaluate the choices made to generate the neighbourhood of each candidate label, as discussed in Chapter 3 and reported in Table 4.3.

We observed that the most consistent way of improving the results is to use more prominent neighbourhoods, as 3-hop neighbourhoods systematically outperform the one and 2-hops ones. Our experiments show that going beyond $n = 3$ comes at the cost of increasing the computation time (mainly the computation of cosine similarity between the label and related nodes) while offering only marginal improvement. The filtering method also impacts the performance but only sometimes (especially for $n = 2, 3$). Finally, using all the relations yields better results than using only a subset, which is enough to justify the speed trade-off. It is also worth noting that using only the "r/RelatedTo" relation yields overall good results, highlighting that "common-sense word relatedness," as expressed in ConceptNet, is a strong signal for topic labelling. For the scoring scheme, we evaluate the various methods mentioned

		Filtering Method			
Relations	Depth	Keep All	Top50%	Top20%	Thresh
All	$n = 1$	150	75	30	25
	$n = 2$	250	125	50	35
	$n = 3$	370	185	74	50
Similarity	$n = 1$	50	25	10	12
	$n = 2$	75	35	14	17
	$n = 3$	120	60	24	30
One	$n = 1$	10	5	2	5
	$n = 2$	21	10	3	8
	$n = 3$	37	18	7	12

Table 4.1: We are optimizing the best parameters for the candidate label selection. To select the best parameters by comparing the different neighbourhood selection and filtering configurations on the BBC News dataset (performance expressed in average no. of candidate labels) for candidate label selection. We later used these while testing on test datasets. We defined the type of relations to consider out of "All," "Similar," and "One." Then we defined the depth of the neighbourhood by selecting n (1, 2, or 3). Finally, we experimented with the number of nodes, such as Keep All the nodes, only the top 50%, 20%, or a fixed threshold.

in Chapter 3. The results are reported in Table 3.

Using the ConceptNet Numberbatch embeddings gives the best results as they can condense the count, distance, degree of the nodes, and linguistic similarity concerning the label into a measure of similarity in the embedding space. Accounting for term frequency (counting a word twice in the scoring if it appears twice in the document) in all scoring schemes did not translate to an improvement in the results. Accounting for n-grams improves the results slightly. However, require the availability of a corpus to mine such n-grams. Therefore, for the rest of our experiments, we do not account for n-grams but instead keep the following configuration: ("All relations,"$n = 3$, "Top20K", "Numberbatch scoring"). We use ConceptNet v5.7 and Numberbatch embeddings v19.08.

Count	Distance	Degree	NumBatch	NumBatch Center	GloVe
81.8	78.4	77.4	88.4	89.4	82.4

Table 4.2: We evaluated the different scoring (label selection) schemes. Utilizing the BBC News dataset (performance expressed in average Accuracy) to perform experiments. We are selecting the best parameters for selecting the most appropriate label for the topic and its documents.

		Filtering Method			
Relations	Depth	Keep All	Top50%	Top20%	Thresh
All	$n = 1$	78.4	77.4	78.4	77.4
	$n = 2$	88.2	88.8	82.0	83.9
	$n = 3$	89.6	89.6	84	85.6
Similarity	$n = 1$	60.8	57.5	60.8	60.8
	$n = 2$	81.3	80.9	81.2	80
	$n = 3$	82.9	81.9	83.4	81.9
One	$n = 1$	55.4	54.4	55.4	55.4
	$n = 2$	69.0	65.8	64.8	66.2
	$n = 3$	81.0	81.3	83.5	81.3

Table 4.3: We optimized the neighbourhood selection algorithm to select the best parameters.
By comparing the different neighbourhood selection and filtering configurations on the BBC News dataset (performance expressed in Accuracy) for candidate label selection. We defined the type of relations to consider out of "All," "Similar," and "One." Then we defined the depth of the neighbourhood by selecting n (1, 2, or 3). Finally, we experimented with the number of nodes, such as Keep All the nodes, only the top 50%, 20%, or a fixed threshold.

4.2 One Word Label

4.2.1 Experiment Setup and Dataset

Based on previous research [4], we used Topic_Bhatia [10] public dataset. A data set contains 228 topics with 19 labels for each topic from four different domains (blogs, books, news, and PubMed). Human ratings for those candidate labels were collected by formulating a crowd-sourcing task on Amazon Mechanical Turk (MTurk). Annotators (i.e., crowd-workers) gave ratings for the labels between 0 and 3, where 3 is the highest rating. Only labels that received a high average rating (of 2 or above) were used for the data set, resulting in 219 topics and 1156 pairs (instead of 4332, i.e., 228 topics $\times$ 19 labels). Unfortunately, the topic-document distributions are not available for topics_bhatia. While the lack of information about the topic-document distributions is far from ideal, we chose to use topics_bhatia since it provides ratings for labels, which are expensive to obtain. Several pretreatments were applied, including removing extra metadata and a few unnecessary stop words. In this study, all methods were implemented using Python. We used Top2Vec [5] model implementation to discover topics. The parameters were either directly borrowed from prior studies

or empirically set, for example, to discover the topics. Samples from the dataset are shown in Table 4.4.

topics_bhatia Dataset	
Topic Terms/Article	Label/Title
vote house election poll bill republican party voter candidate senate	election
health patient medical doctor hospital disease cancer care drug study	hospital
food eat cook chicken recipe cup cheese add taste tomato	cooking
god church jesus christian faith lord christ catholic give prayer	catholicism

Table 4.4: Displaying a few samples of topics and their corresponding labels from the dataset topics_bhatia of different categories.

Baselines

The labels generated by our models were compared with three baselines:

1. **the top two terms**: In terms of the highest marginal probabilities for a topic (Top-2 label).

2. **the top three terms**: In terms of the highest marginal probabilities for a topic (Top-3 label).

3. **The Alokaili seq2seq method**: [4] proposed a supervised seq2seq automatic topic labelling method.

Label Evaluation

BERTScore [39] was used to evaluate the quality of the generated labels.[1] BERTScore is a measure that computes the similarity between predictions and references using contextual embeddings that have shown to have a high correlation with human judgments. Since BERTScore does not rely on exact matches between predicted and gold-standard labels, it can identify appropriate label words that do not appear in

[1]Results were generated using the reference implementation: `https://github.com/Tiiiger/bert_score`

the gold labels. Pairwise BERTScores between the topic's generated label l and gold labels

$$(gold\_l_1, ..., gold\_l_n)$$

is computed as follows:

$$score\_topic_t = \max_{[1,...,n]} BERTScore(l_t, gold\_l_t i)$$

The model's overall score is the mean score for overall topics:

$$score\_model = \frac{1}{T} \sum_{t=1}^{T} score\_topic_t$$

4.3 Sentence Label

4.3.1 Experiment setup and Dataset

Based on previous research [18, 1], we used APNews[2] public document collection. Several pretreatments were applied, including removing punctuations and a few unnecessary stop words and filtering out elements (other than nouns, verbs, adjectives, adverbs, and pronouns). Consequently, we achieved 2246 documents, 40,766 sentences, and 26,777 vocabularies in APNews. In this study, all methods were implemented using Python. We used Top2Vec [5] model implementation to discover topics. All the parameters are borrowed from prior studies or empirically set.

4.3.2 Baselines

The labels generated by our models were compared with three baselines:

1. LexRank: The LexRank[3] [14] creates a graph based on the candidate sentences and votes equally to related vertices.

2. TextRank: TextRank[4] [26] is regarded as an improved version of LexRank. For LexRank and TextRank, if each article is viewed as a single document, the topic labelling task can be transformed into a single document summary task.

[2]A total of 2246 APNews articles downloaded from GitHub, `https://github.com/Blei-Lab/lda-c/blob/master/example/ap.tgz`.

[3]LexRank method implemented by lexrank (Python tools package) `https://github.com/crabcamp/lexrank`.

[4]TextRank method implemented by summa-textrank (Python tools package) `https://github.com/summanlp/textrank`.

3. TLRE: Based on surface features computing, He et al. [18] proposed a method for automatic topic labelling using graph-based ranking. Based on neural embedding, TLRE [17] is an improved method that significantly improves.

4.3.3 Label Evaluation

In this section, inspired by the BERTScore method, we use a new automatic metric proposed by [17] that highly correlates with human judgments and does not rely on perfect matching.

1. Relevance: BERTScore[39] was used to evaluate the quality of the generated labels. [5] BERTScore is a measure that computes the similarity between predictions and references using contextual embeddings that have shown to have a high correlation with human judgments proven by [39]. Since BERTScore does not rely on exact matches between predicted and gold-standard labels, it can identify appropriate label words that do not appear in the gold labels. Pairwise BERTScores between the topic's generated label l and the topic documents

$$(doc\_l_1, ..., doc\_l_n)$$

is computed as follows:

$$score\_topic_t = \max_{[1,...,n]} BERTScore(l_t, doc\_l_t i)$$

The model's overall score is the mean score for overall topics:

$$score\_model = \frac{1}{T} \sum_{t=1}^{T} score\_topic_t$$

The F1 score of BERTScore is used as the Relevance between the original document and generated topic labels. The documents with a probability score of at least fifty percent are selected from the topic for relevance calculation. This threshold is selected after the evaluation. Suppose a document with a probability score of less than fifty percent is assigned to a topic. In that case, it belongs to some other topic in the majority based on the probability. Another metric

[5] Results were generated using the reference implementation: `https://github.com/Tiiiger/bert_score`

we used is selecting documents with a probability score above the mean of all the documents.

2. Coverage: According to the study by researchers [17], Coverage is defined as the ratio of words that appeared in the top 20 topic terms, and the equation is as follows. It is important to note that $UNI(Si)$ returns a set of unique sentence characters Si.

$$Coverage = ((\sum_{i=1}^{k}(\sum_{w \epsilon UNI(S_i)} [w \epsilon Top20(T_i)]/|Top20(T_i)|))/K)$$

The top 20 words were selected as opposed to the remaining since they are more meaningful and representative than the others. Higher Coverage often indicates more thorough coverage of the sentence label's top topic phrases. We average each number for coverage.

4.4 Summary Label

4.4.1 Experiment setup and Dataset

Based on previous research [18, 1], we used APNews[6] public document collection. Several pretreatments were applied, including removing punctuations and a few unnecessary stop words and filtering out elements (other than nouns, verbs, adjectives, adverbs, and pronouns). Consequently, we achieved 2246 documents, 40,766 sentences, and 26,777 vocabularies in APNews. In this study, all methods were implemented using Python. We used Top2Vec [5] model implementation to discover topics. All the parameters are borrowed from prior studies or empirically set. Unless stated otherwise, all summaries are generated using the Pegasus [38] summarizing model.

4.4.2 Baselines

The labels generated by our models were compared with three baselines:

1. LexRank: The LexRank[7] [14] creates a graph based on the candidate sentences and votes equally to related vertices.

[6]A total of 2246 APNews articles downloaded from GitHub, `https://github.com/Blei-Lab/lda-c/blob/master/example/ap.tgz`.

[7]LexRank method implemented by lexrank (Python tools package) `https://github.com/crabcamp/lexrank`.

2. TextRank: TextRank[8] [26] is regarded as an improved version of LexRank. For LexRank and TextRank, if each article is viewed as a single document, the topic labelling task can be transformed into a single document summary task.

3. TLRANK: Based on surface features computing, He et al. [18] proposed a method for automatic topic labelling using graph-based ranking. Based on neural embedding.

4. TLPA: Based on paired-attention TLPA [16] proposed a method for automatic topic labelling using paired-attention-based pre-trained deep neural networks.

4.4.3 Label Evaluation

1. Relevance: The summary subject labels are more effective when they are more relevant and less redundant. KLD combines relevance and redundancy effectively, according to the study [29], making it an ideal tool for measuring how well the produced summary text performs these two functions. We first calculated the KLD value between each topic label created and the matching topic, then averaged all the KLD values.

2. Coverage: According to the study by researchers [17], Coverage is defined as the ratio of words that appeared in the top 20 topic terms, and the equation is as follows. It is important to note that $UNI(Si)$ returns a set of unique sentence characters Si.

$$Coverage = ((\sum_{i=1}^{k}(\sum_{w \epsilon UNI(S_i)} [w \epsilon Top20(T_i)]/|Top20(T_i)|))/K)$$

   Because the top 20 terms are more relevant and indicative than the others, we chose them above the top 500. Higher Coverage often means that the summary label has more top-topic phrases covered and is more thorough. All of the Coverage figures are averaged.

[8] TextRank method implemented by summa-textrank (Python tools package) `https://github.com/summanlp/textrank`.

Chapter 5

Results and Discussion

5.1 One Word Label

First, let us discuss how one-word label generation is performed. Our labelling model is based on the Zero-Shot approach while utilizing the ConceptNet [33] knowledge graph to leverage the external knowledge. For the evaluation, the model generated labels for the Bhatia et al. [10] dataset topics by passing the topic's top 10 terms (topic_bhatia) as top n words (based on cosine similarity scores) as well as the topic document. Results are shown in Table 5.1.

All model variations produce significantly higher scores than baselines of selecting the top two or three terms with the highest marginal probability as labels.

The highest scores are obtained when the labelling model is set to the best performance and allowed relations are set to "all" (**topics_bhatia_all_relations**). Having all the relations allowed extending the neighbourhood of the candidate label to the max while considering all the relations inside ConceptNet and selecting max neighbours, which improves the results because it adds more related keywords to the candidate list. Then we used numberbatch ConceptNet embeddings to see the relevance of the keyword with the topic and select the most appropriate keywords, thus improving the results. Sample labels generated using our model are shown in Table 5.1. The topics and golden labels are taken from **topics_bhatia**.

To further test the significance of each component in our proposed algorithm. We have two other variations of our proposed algorithm.

First, we switched off the algorithm's candidate label selection component (Top2 Label _without_candidate). After this, the algorithm will not generate additional candidate labels other than top n words (based on cosine similarity scores) and now only consider top n words (based on cosine similarity scores) as candidate labels. Switching off this component reduces the number of candidate labels we have to evaluate further in our validation and selection step. After this, we can see a significant

performance reduction compared to the complete algorithm while considering all the relations (Top2Label_all_relations). It performed even less than when we considered only related relations (Top2Label_related_relations). This performance reduction is seen because we now have very few candidate labels, only top n words (based on cosine similarity scores). Among all the candidate labels, only a few were significant enough to be considered suitable. Which reduces the document overlap score and thus reduces the performance in cases when we could not find the top three predictions with a good enough overlap score, thus reducing the performance Table 5.1.

Second, we switched off the document overlap evaluation scheme (Top2Label _without _eval_scheme) and used GloVe embedding as the metric to calculate the overlap score between the candidate labels and documents. Switching off the evaluation scheme significantly impacted the algorithm's performance, and now we can only achieve an f1 score of 0.865; results can be seen in Table 5.1. This reduction is seen because the neighbourhood we are generating for each label depends on this document overlap evaluation scheme. Without it, it reduces the quality of the neighbourhood we are developing for each label. Thus, it reduces the number of good-quality candidate nodes with good document overlap scores for final label selection and reduces performance.

When we compare the results of our algorithm Top2Label during the ablation study when different components of our algorithm are switched off, such as Top2Label _without _candidate and Top2Label_without_eval_scheme. Furthermore, compared to the entire algorithm, Top2Label_all_relations. It can be seen that there is a significant reduction in the overall performance of the algorithm and how significant these components are for the proposed algorithm Top2Label to perform at its peak, and can be seen in Table 5.1.

It can also be seen that the labels generated by the model are within the correct domain and similar to the gold labels. In some cases where the model did not generate an entirely new label, it picked one or two words from the topic terms. For example, the topic vmware, server, virtual, oracle, update, virtualization, application, infrastructure, management, microsoft was labelled with vmware by the model when it was set for best speed where allowed relations are set to "related."

In terms of inference time, our model performs best by computing labels almost

Results			
	BERTScore		
Baselines	Precision	Recall	F1
Top-2 label	0.902	0.912	0.902
Top-3 label	0.870	0.903	0.882
Alokaili topics_bhatia	0.919	0.926	0.919
Alokaili topics_bhatia_tfidf	0.930	0.933	0.929
Propsed Method	Precision	Recall	F1
Top2Label_all_relations	**0.935**	**0.937**	**0.936**
Top2Label_related_relations	0.91	0.915	0.912
Top2Label_without_candidate	0.903	0.907	0.905
Top2Label_without_eval_scheme	0.863	0.867	0.865

Table 5.1: Displaying the outcome of testing on topics_bhatia dataset. The scores shown are produced using the BERTScore scoring scheme defined in section 4.2.1 between predicted and actual labels. Each predicted label is compared to a set of original labels to measure appropriateness. Here we are comparing four baselines to our model with four variations of the proposed algorithm. We can see in the bold the best results.

half the time. When we made the inference on the labelling model and set to best speed and allowed relations are set to "related" (**topics_bhatia_related_relation**), which consider only a few relations (i.e., 'defined as,' 'DerivedFrom,' 'HasA,' 'InstanceOf,' 'ISA,' 'PartOf,' 'RelatedTo,' 'SimilarTo,' 'Synonym,' 'Antonym'). While comparing it to the baseline on a system with eight GB ram, a dual-core processor, and no graphics card, results can be seen in Fig. B.1.

We conducted an error analysis to examine cases where the model produced suboptimal labels. For example, the topic mr, Mrs, young, lady, look, friend, tell, mother, miss, father was labelled with the may be due to the topic needing to be more coherent and with a prominent theme.

For different domain topics artery, vascular, coronary, stent, vein, vessel, carotid, aortic, aneurysm, arterial was labelled by vascular, disease, which is handled perfectly even though the topic is from a different domain among the topics in the dataset, which shows that our zero-shot approach can easily handle any data and performs well on it.

Model Tested On **topics_bhatia**		
	bhatia_all_rl	**bhatia_related_rl**
Topic 1	vmware server virtual oracle update virtualization application infrastructure management microsoft	
Gold labels (Top 5)	cloud computing, vmware, web application, virtualization, operating system	
Top2Label	vmware, virtual, server	vmware, virtualization
Topic 2	obama mccain campaign john barack president senator candidate convention clinton	
Gold labels (Top 5)	conservative democrat,democratic party presidential nominee,bill clinton	
Top2Label	obama, campaign, presidential	obama, presidential

Table 5.2: Displaying samples of labels predicted by variations of the proposed models tested on topics_bhatia datasets with their original gold labels.

5.2 Sentence Label

The model generated labels for the APNews dataset topics by passing the topic's top n words (based on cosine similarity scores) and the topic document. For each topic, first, we calculate and generate the candidate labels. We select the top candidate labels to generate or extract the sentences, then we validate and select the most suitable candidate sentence as the label. Results are shown in Table 5.4.

Our model produces significantly higher scores than most baselines by creating a sentence label with higher relevance and coverage scores. The highest scores are obtained when the labelling model was set to the best performance and allowed relations from the ConceptNet knowledge graph are set to "all" (**AllowedRelations="All"**).

Extending the neighbourhood of top n words (based on cosine similarity scores) to generate the maximum number of candidate labels while using all the relations and selecting max neighbours improves the results. Our model's sample labels are shown in Table 5.3. We can also see that the labels generated by the model are within the correct domain and similar to the original article.

In some cases where the model did not generate an entirely new label, it picked parts of the original text. Although the approach is zero-shot, it has performed very well compared to all the baselines and is near to supervised methods.

We propose a novel zero-shot framework which utilizes external knowledge from a knowledge graph and language model to label the topics—our approach in contrast with TLRE. The evaluation results of the automatic topic labelling method (TLRE) under Relevance and Coverage can be seen in Table 5.4. As depicted in the results, our process performed better in coverage due to the processing of external knowledge with the help of a language model. At the same time, we create the neighbourhood graph in our approach and have more top keywords included in the label. However, this seemed to be inapplicable to Relevance. To further illustrate the Coverage details of the topic label, we gathered specific correlated data in APNews, seen in Fig. As we consider more top n terms (based on cosine similarity scores) of the topic in the neighbourhood graph, the coverage tends to increase, but this is not the case with Relevance. This coverage increase might be because if we use more top n terms and during ConceptNet querying, we will select more terms related to these words, which tends to increase the coverage. However, this inclusion changes the sentences and increases the sample space of sentences which affects the selection of a better sentence due to a change in the emphasis of input to sentence generation.

It is known that the top terms can represent the topic in high fidelity. We demonstrate that the top terms of a topic discovered in the case of APNews are significant. However, reading only these topic terms still needs to be clarified, as in such a case, we cannot understand the exact content of the topic, only the most probable. Fortunately, by reading the sentence label of the topic, we can understand the meaning of the topic. Therefore, generated topic label examples, generated by Baseline and Zero-Shot using KG, are shown below. It should be noted that after preprocessing the corpus, all generated topic labels and the candidate sentences do not contain any stop words. Therefore, for smooth and coherent reading, we used the original sentences containing the stop words to form the example of sentence topic labels.

Sentence topic label example 1 (Zero-Shot using KG): Police in Virginia say a **teacher** was **killed** and another was **wounded** in a **shooting** at a **school** in Portsmouth.

Sentence topic label example 2 (baseline): A boy killed a **teacher** at the **school** in Portsmouth.

After observing the label (generated by our model - Zero-Shot using KG and

baseline), we found that higher probability top topic terms are preferable (bold terms in sample). The sentence generation model prefers top terms with higher probability and their neighbours from the graph. Compared with the baseline, our method has better coverage and relevance and had more top20 topic terms (3 vs. 5), more top1-10 topic terms (2 vs. 3), and a higher sum of marginal probabilities of words belonging to top20 topic terms (4.71524 vs. 3.97636).

It is evident that by reading the sentence text from Table 5.4 and Table 5.3 of the topic, our approach performs better in terms of coverage. Sentences generated by the proposed framework not only contain a higher total number of top topic terms but also has more broad coverage of categories of different top topic terms. Therefore, the content of label-Zero-Shot using KG is more comprehensive and rich and has better readability and diversity.

Model Tested On **APNews**	
Article 1	A 16-year-old student at a private Baptist school who allegedly killed one teacher and wounded another before firing into a filled classroom apparently "just snapped," the school's pastor said. "I don't know how it could ...
Predicted Sentence Label	Police in Virginia say a teacher was killed and another was wounded in a shooting at a school in Portsmouth.
Relevance Score:	0.871
Article 2	The Bechtel Group Inc. offered in 1985 to sell oil to Israel at a discount of at least $650 million for 10 years if it promised not to bomb a proposed Iraqi pipeline, a Foreign Ministry official said Wednesday. But then-Prime Minister Shimon Peres said the offer from Bruce Rappaport, a partner in the San Francisco-based told The Associated ...
Predicted Sentence Label	A US company has denied making an offer to Israel 30 years ago to sell oil at a discount if not to bomb proposed pipeline.
Relevance Score:	0.891

Table 5.3: We display sample labels the proposed model predicted while testing on the APNews dataset. We can see the original article from the dataset, the predicted sentence label and the relevance score of each label compared to the article.

We further tested our approach by doing an ablation study and strategically switching off a few components of the proposed algorithm to see the effects of each module on the results. We created two other variations of our proposed model. First, we switched off the algorithm's candidate label selection component (Top2Label

_without_candidate). After this, there will be no additional candidate labels other than top n words (based on cosine similarity scores). Now, we only consider top n words (based on cosine similarity scores) as candidate labels, significantly reducing the number of candidate labels we have to evaluate further in our validation and selection step. After this, we can see a significant performance reduction compared to the complete algorithm while considering all the relations (Top2Label_all_relations). It performed even less than when we considered only related relations (Top2Label_related _relations). This reduction in performance is seen because now we have very few candidate labels to work with, which reduces the size of the neighbourhood graph significantly, which then reduces the number of sentences generated and selected for final label scoring with good enough keywords, which reduces the coverage and relevance of the sentence compared to document Table 5.4.

Second, we switched off the document overlap evaluation scheme (Top2Label _without_eval_scheme) and used glove embedding as the metric to calculate the overlap score between the candidate labels and documents. Switching off the overlap evaluation scheme significantly impacted the algorithm's performance and can be seen in Table 5.4. This reduction is seen because the neighbourhood we are generating for each label depends on this document overlap evaluation scheme. Without it, it reduces the neighbourhood quality we are developing for each label. Which reduces the quality of generated sentences with good document overlap score for final sentence label selection and leads to reduced performance.

Switching off the language model from the approach was not possible because the language model is a core component of the algorithm. Without the language model, sentence label generation is unlikely; thus, we would not have any labels to compare with for our study.

When we compare the results of our algorithm Top2Label, when different components of our algorithm Top2Label_without_candidate and Top2Label_without_eval _scheme were switched off and compared to Top2Label_all_relations when we run the complete algorithm, we can see in Table 5.4 that there is a significant reduction in the overall quality of the sentences generated by the algorithm and how vital these components are for the proposed algorithm Top2Label.

APNews Dataset Results		
Baselines	Relevance	Coverage
LexRank	0.681	0.196
TextRank	0.403	0.284
TLRE	**0.888**	0.222
Proposed Method	Relevance	Coverage
Top2Label_all_relations	0.826	**0.294**
Top2Label_related_relations	0.766	0.275
Top2Label_without_candidate	0.726	0.213
Top2Label_without_eval_scheme	0.706	0.201

Table 5.4: We are displaying the outcome of testing on the APNews dataset. Coverage and Relevance scores between predicted sentence labels and original text articles can be seen. Relevance is calculated using BERTScore, and Coverage defines the number of original keywords in the prediction. Section 4.3.3 has more details on the evaluation scheme. Each predicted label is compared to the original document to measure appropriateness as described. We are comparing our proposed model with its four variations to 4 baselines as defined in section 4.3.2. The highest scores are highlighted in bold in the table.

5.3 Summary Label

The language model generated the summary labels for the APNews dataset topics using generated sentence corpora and original topic documents. Results are shown in Table 5.6. Our model produces significantly higher scores than most baselines by creating a sentence label with higher relevance and coverage scores. The highest scores are obtained when the labelling model is set to consider all the relations while querying the ConceptNet and allowed relations are set to (**AllowedRelation="All"**).

Extending the neighbourhood of the candidate label to all the relations to generate max candidate labels through neighbourhood improves the results. Sample labels generated using our model are shown in Table 5.5. Labels generated by the proposed framework are within the correct domain and similar to the original article.

In some cases where the model did not generate an entirely new label, it picked parts of the summary from the original text.

Inspired by [15], we introduced a novel candidate label creation approach that utilizes the external knowledge of the knowledge graph(ConceptNet) and improves the coverage of the summaries by feeding more keywords related to the top n words (based on cosine similarity scores) of the topic. To further enhance the model's

performance, we selected the candidate labels graph based on the semantic similarity to the original text in the topic. This graph selection eventually led to a better sentence corpus with better coverage. Then we feed the corpora with the original topic document to the summarization language model, which generates a final summary emphasizing candidate labels and improves overall coverage and relevance matrices. There are multiple tuning available in the proposed approach to tune the results. We have considered all the relations while querying the ConceptNet during candidate label creation for best performance. We can reduce the relations to related, which can lead to an immediate improvement in the speed of summary label creation because it reduces the query size and, eventually, the number of candidate labels to evaluate to generate the final summary. To further understand the improvements provided by our novel approach, we compare it with two unsupervised baselines (LexRank and TextRank) and two supervised approaches (TLRANK and TLPA) shown in Table 5.6.

Our approach has a significantly better score than the unsupervised approach, which can be seen in Table 5.6, due to the fact that unsupervised approaches rely on hand-crafted features.

Supervised techniques still have the advantage regarding topic labelling seen with TLPA and can be seen in results when tested on the APNews dataset in Table 5.6. However, we must consider the significant amount of time and resources required to train.

Sensitivity to the top n terms (based on cosine similarity scores), when we use corpus with sentences with more top n terms of the topic in the neighbourhood graph, the coverage increases, but this is not the case with relevance. This might be the fact that if we use more top n terms (based on cosine similarity scores), and during ConceptNet querying, we will be selecting more terms related to these words which tends to increase the coverage, but this inclusion changes the sentences and increases the sample space of sentences which affects the selection of a better sentence due to changes in the emphasis of input to sentence generation.

It needs to be clearer to read only these topic terms, as in such a case, we cannot understand the exact content of the topic, only the most probable. Fortunately, by reading the summary label of the topic, we can understand the meaning of the topic. Let us compare labelled examples generated by Baseline and Zero-Shot using

KG, shown below. After preprocessing the corpus, all generated topic labels and the candidate sentences do not contain any stop words. Therefore, for smooth and coherent reading, we used the original sentences containing the stop words to form the example of sentence topic labels.

Summary topic label example 1 (Zero-Shot using KG): A **wildfire** burning out of **control** in the **western** US has forced the **evacuation** of hundreds of people from their **homes**. **Fire officials** say the blaze has been sparked by **lightning** and is burning out of control in several **states**, including Arizona, California, Colorado, Idaho, Montana, Nevada, Oregon, and Washington, the **AP reports**.

Summary topic label example 2 (Baselines): A **wildfire** is out of **control** in the **western** US. The fire caused by **lightning**. Affecting in several **states**, including Arizona, California, Colorado, Idaho, Montana, Nevada, Oregon, and Washington, the **AP reports**.

After observing the label (generated by our model - Zero-Shot using KG and baseline), we found that higher probability top topic terms are preferable (bold words in sample). Summary give more preference to top terms with higher probability and their neighbours from the graph. Compared with the baseline, our method has better coverage and relevance and has more top 20 topic terms (6 vs. 10) and more top1-10 topic terms (3 vs. 5).

It was evident that reading the sentence text from Table 5.6 and Table 5.5 of the topic, our approach performs very well and better in terms of coverage. Summary labels generated by the proposed framework not only contain a higher total number of top topic terms but also has more broad coverage of categories of different top topic terms. Therefore, the content of label-Zero-Shot using KG is more comprehensive and rich and has better readability and diversity.

Similar to the previous two labels, we tested components of our approach by doing an ablation study and strategically switching off a few elements of the proposed algorithm to see the significance of each module on the results.

We created three other variations of our proposed model.

First, we switched off the algorithm's candidate label selection component (Top2 Label _without_candidate). After this, there will be no additional candidate labels other than top n words (based on cosine similarity scores). We only consider top n

Model Tested On **APNews**	
Article 1	A gunman took a 74-year-old woman hostage after he was foiled in an attempt to steal $1 million in jewelry belonging to the late Liberace, but police shot and killed the man outside the entertainer's museum. "I just tried to stay ...
Predicted Summary Label	Police have identified the gunman who held a 74-year-old woman hostage at the Liberace Museum in Las Vegas on Sunday as Hugh Perry, 47, and say he was attempting to steal more than $1 million in jewelry belonging to the late ...
Relevance Score:	0.891
Article 2	Today is Saturday, Oct. 29, the 303rd day of 1988. There are 63 days left in the year. A reminder: daylight-saving time ends tomorrow at 2 a.m. local time. Clocks "fall back" one hour. Today's highlight in history: In 1929, "Black Tuesday" descended upon the New York Stock Exchange. Prices collapsed amid panic selling, thousands of ...
Predicted Summary Label	The man accused of shooting and killing his wife's best friend in broad daylight on a New York City street is set to appear in court today. Prosecutors are expected ...
Relevance Score:	0.901

Table 5.5: We display samples of labels produced while testing the proposed approach on the APNews dataset. Here we can see the original topic article for which the label is generated, the predicted summary label, and its relevance score compared to the article.

words as candidate labels, significantly reducing the number of candidates we have to evaluate further in our validation and selection step. After this, we can see a significant performance reduction compared to the complete algorithm while considering all the relations (Top2Label_all_relations). It performed even less than when we considered only related relations (Top2Label_related_relations). This reduction in performance is seen because now we have very few candidate labels to work with, which reduces the size of the neighbourhood graph significantly, which then reduces the number of sentences generated and selected for the corpus to create a summary with good enough keywords, which reduces the coverage and relevance of the summary compared to document Table 5.6.

Second, we switched off the document overlap evaluation scheme (Top2Label_without _eval_scheme) and used glove embedding as the metric to calculate the overlap score between the candidate labels and documents. Switching off the document overlap

evaluation scheme had a significant impact on the performance of the algorithm. This reduction is seen because the neighbourhood we are generating for each label depends on this document overlap evaluation scheme. Without it, it reduces the neighbourhood quality we are developing for each label. This reduces the quality generated sentences with good document overlap score for the final sentence corpus and leads to reduced performance on summarization tasks seen in Table 5.6.

Third, we stopped generating sentence corpus from the neighbourhood graph and created summaries by passing topic documents directly to the language model. This led to the highest reduction in performance, specifically in coverage, because now we are not feeding the most important words (top n Words) from the document to summaries as additional corpus Table 5.6.

When we compare the results of our algorithm Top2Label, when different components of our algorithmTop2Label_without_candidate, Top2Label_without_eval_scheme, and Top2Label_without_sent_corpus were switched off and compared to Top2Label_all _relations when we run the complete algorithm, we can see in Table 5.6 that there is a significant reduction in the overall quality of the summaries generated by the algorithm and how significant these components are for the proposed algorithm Top2Label.

Results		
Baselines	Relevance	Coverage
LexRank	4.080	0.0.15
TextRank	3.927	0.17
TLRANK	**2.862**	0.23
TLPA	**2.630**	0.26
Proposed Method	Relevance	Coverage
Top2Label_all_relations	3.121	**0.31**
Top2Label_related_relations	3.323	0.27
Top2Label_without_candidate	3.421	0.25
Top2Label_without_eval_scheme	3.498	0.24
Top2Label_without_sent_corpus	3.612	0.20

Table 5.6: We are displaying the outcome of testing on the APNews dataset. Coverage and Relevance scores between the predicted summary label and the topic can be seen. Relevance is calculated as the average of KL Divergence between topic label and topic, and Coverage defines the number of original keywords in the prediction. Section 4.4.3 details the evaluation scheme. Here we are comparing four baselines with five variations of the proposed approach. Top2Label_all_relations defines the evaluation on a complete approach. Each predicted label is compared to the original document to measure appropriateness as described in Section 4.4.3. The best scores are highlighted in bold in the table. For Relevance, lower is better, and for Coverage, higher is better.

Chapter 6

Conclusion

We presented the first zero-shot model to generate all three types of textual labels (i.e., 1. One Word Label, 2. Sentence Label, and 3. Summary Label) for automatically generated topics. We have defined our evaluation matrix based on BERTScore, which is used to measure the similarities between the generated label and gold standard labels in the case of One Worded Label and between the original Article and generated label for Sentence and Summary labels. Our zero-shot approach is sound and produces appropriate labels.

We successfully proposed a novel zero-shot topic labelling model based on the ConceptNet knowledge graph combined with a language model to prove the effectiveness of this blend in topic labelling tasks. The model is named Top2Label, which aims to generate a topic label of three types (1. One Word Label, 2. Sentence Label, and 3. Summary Label) for each topic modelled by Top2Vec (or similar such as LDA) [5, 11] to help the user understand it clearly. The experimental evaluation results demonstrate that our approach significantly and consistently outperforms the prevailing unsupervised methods and performs on par with supervised methods.

In future research, we would like to resolve the only limitation we found in our system during experiments that arise when we generate sentence labels. During this process, when we use Graph2Corpus to create the corpus if the Graph2Corpus algorithm does not find any related sentences in the KELM corpus and topic documents are not present or do not contains sentences, then Graph2Corpus passes only graph edges as a corpus to the language model to generate the sentence label which then returns these keywords as it is without any sentence which still gives us a label but not is exact sentence form which we are expecting. To remove this, a language model can be fine-tuned to improve the Graph2Corpus corpus quality by using the final neighbourhood graph as input and generating a corpus. We can fine-tune a language model in ways like that it generates sentences from knowledge graph triplets and

output a sentence of all the edges and relationships of a knowledge graph. This fine-tuning will ensure if Graph2Corpus does not find any sentences, then also language model gives a meaningful sentence as the label.

Bibliography

[1] Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. Knowledge Graph Based Synthetic Corpus Generation for Knowledge-Enhanced Language Model Pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565. Association for Computational Linguistics, 2021-06.

[2] Nikolaos Aletras and Arpit Mittal. Labeling Topics with Images Using a Neural Network. In Joemon M Jose, Claudia Hauff, Ismail Sengor Altıngovde, Dawei Song, Dyaa Albakour, Stuart Watt, and John Tait, editors, *Advances in Information Retrieval*, Lecture Notes in Computer Science, pages 500–505. Springer International Publishing, 2017.

[3] Mehdi Allahyari, Seyedamin Pouriyeh, Krys Kochut, and Hamid Reza Arabnia. A Knowledge-based Topic Modeling Approach for Automatic Topic Labeling. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 8(9), 2017/17/29.

[4] Areej Alokaili, Nikolaos Aletras, and Mark Stevenson. Automatic Generation of Topic Labels. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pages 1965–1968. Association for Computing Machinery, 2020-07-25.

[5] Dimo Angelov. Top2vec: Distributed representations of topics. *arXiv*, 2020.

[6] Claus Boye Asmussen and Charles Møller. Smart literature review: A practical topic modelling approach to exploratory literature review. *Journal of Big Data*, 6(1):93, 2019-10-19.

[7] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, Lecture Notes in Computer Science, pages 722–735. Springer, 2007.

[8] Elena Baralis, Luca Cagliero, Naeem Mahoto, and Alessandro Fiori. GraphSum: Discovering correlations among multiple terms for graph-based summarization. *Information Sciences*, 249:96–109, 2013-11-10.

[9] Mohamad Hardyman Barawi, Chenghua Lin, and Advaith Siddharthan. Automatically Labelling Sentiment-Bearing Topics with Descriptive Sentence Labels. In Flavius Frasincar, Ashwin Ittoo, Le Minh Nguyen, and Elisabeth Métais, editors, *Natural Language Processing and Information Systems*, Lecture Notes in Computer Science, pages 299–312. Springer International Publishing, 2017.

[10] Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. Automatic Labelling of Topics with Neural Embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 953–963. The COLING 2016 Organizing Committee, 2016-12.

[11] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003-03-01.

[12] Amparo Elizabeth Cano Basave, Yulan He, and Ruifeng Xu. Automatic Labelling of Topic Models Learned from Twitter by Summarisation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 618–624. Association for Computational Linguistics, 2014-06.

[13] David Carmel, Haggai Roitman, and Naama Zwerdling. Enhancing cluster labeling using wikipedia. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 139–146. Association for Computing Machinery, 2009-07-19.

[14] G. Erkan and D. R. Radev. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004-12-01.

[15] Ismail Harrando and Raphaël Troncy. Explainable Zero-Shot Topic Extraction Using a Common-Sense Knowledge Graph. In *LDK 2021, 3rd Conference on Language, Data and Knowledge*, 2021-09.

[16] Dongbin He, Yanzhao Ren, Abdul Mateen Khattak, Xinliang Liu, Sha Tao, and Wanlin Gao. Automatic Topic Labeling model with Paired-Attention based on Pre-trained Deep Neural Network. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, 2021-07.

[17] Dongbin He, Yanzhao Ren, Abdul Mateen Khattak, Xinliang Liu, Sha Tao, and Wanlin Gao. Automatic topic labeling using graph-based pre-trained neural embedding. *Neurocomputing*, 463:596–608, 2021-11-06.

[18] Dongbin He, Minjuan Wang, Abdul Mateen Khattak, Li Zhang, and Wanlin Gao. Automatic Labeling of Topic Models Using Graph-Based Ranking. *IEEE Access*, 7:131593–131608, 2019.

[19] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57. Association for Computing Machinery, 1999-08-01.

[20] Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 465–474. Association for Computing Machinery, 2013-02-04.

[21] Akanksha Joshi, Eduardo Fidalgo, Enrique Alegre, and Rocio Alaiz-Rodriguez. RankSum—An unsupervised extractive text summarization based on rank fusion. *Expert Systems with Applications*, 200:116846, 2022-08-15.

[22] Pooja Kherwa and Poonam Bansal. Latent Semantic Analysis: An Approach to Understand Semantic of Text. In *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)*, pages 870–874, 2017-09.

[23] Salima Lamsiyah, Abdelkader El Mahdaouy, Bernard Espinasse, and Saïd El Alaoui Ouatik. An unsupervised method for extractive multi-document summarization based on centroid approach and sentence embeddings. *Expert Systems with Applications*, 167:114152, 2021-04-01.

[24] Wei Li and Andrew McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584. Association for Computing Machinery, 2006-06-25.

[25] Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. Automatic labeling of multinomial topic models. In *Automatic Labeling of Multinomial Topic Models*, KDD '07, page 490–499, New York, NY, USA, 2007. Association for Computing Machinery.

[26] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411. Association for Computational Linguistics, 2004-07.

[27] Nihal V. Nayak and Stephen H. Bach. Zero-Shot Learning with Common Sense Knowledge Graphs.

[28] Mark Newman. *Networks: An Introduction*. Oxford University Press, Inc., 2010.

[29] Maxime Peyrard. A simple theoretical model of importance for summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1059–1073, Florence, Italy, July 2019. Association for Computational Linguistics.

[30] C. R. Sandeep, Asif Salim, R. Sethunadh, and S. Sumitra. An efficient scheme based on graph centrality to select nodes for training for effective learning.

[31] Yoon Seok-Ho, K. I. M. Ji-Su, K. I. M. Sang-Wook, and L. E. E. Choonhwa. TL-Rank: A Blend of Text and Link Information for Measuring Similarity in Scientific Literature Databases. *IEICE Transactions on Information and Systems*, E95.D(10):2556–2559, 2012.

[32] Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: An open multilingual graph of general knowledge. In *AAAI*, pages 4444–4451, 2017.

[33] Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 4444–4451. AAAI Press, 2017-02-04.

[34] Robyn Speer and Joanna Lowry-Duda. ConceptNet at SemEval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 85–89, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[35] Xiaojun Wan and Tianming Wang. Automatic Labeling of Topic Models Using Text Summaries. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2297–2305. Association for Computational Linguistics, 2016-08.

[36] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 697–702, 2007-10.

[37] Bo Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Lihan Chen, and Yanghua Xiao. CN-DBpedia2: An Extraction and Verification Framework for Enriching Chinese Encyclopedia Knowledge Base. *Data Intelligence*, 1(3):271–288, 2019-06-01.

[38] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20, pages 11328–11339. JMLR.org, 2020-07-13.

[39] Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020.

[40] Elaine Zosa, Lidia Pivovarova, Michele Boggia, and Sardana Ivanova. Multilingual Topic Labelling of News Topics Using Ontological Mapping. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*, pages 248–256. Springer-Verlag, 2022-04-10.

Appendix A

Graph2Corpus Corpus Generation Algorithm

We have defined a text generation algorithm (TEXTGEN) to generate a text corpus that takes graph nodes (triplets) from neighbourhood graphs with document overlap scores as input and returns the text corpus as output.

We start by taking the document overlap neighbourhood graph nodes as input. Then, we search the graph nodes in KELM Corpus [1] (KELM Copus is a dataset created from DBPedia by creating a sentence for each DBpedia triplet) to find the sentence containing graph nodes. We compare the relevance of each sentence with the topic using Numberbatch Embeddings (ConceptNet Embeddings [33]). If relevance is above the threshold, we select the sentence and add it to the corpus. Additionally, we extract all the sentences from the document containing the graph node words and add them to the corpus. And finally, add all the graph nodes to the corpus as keywords.

Now we have a knowledge graph (external knowledge) in the form of a corpus.

We further use this corpus for sentence and summary label generation.

Algorithm 10: Graph2Corpus Algorihm

Input: $NeighbourhoodGraphWithDocOverlapScore \geq 0$

n # Threshold to select triplets

```
1  if NeighbourhoodGraphWithDocOverlapScore ≠ null then
2      for edge in NeighbourhoodGraphWithDocOverlapScore.Edges do
3
4          if edge.score ≥ n then
5              # Searching KELM Corpus for sentences which contains the Edge
                (Graph Node) and selecting the most relevant using ConceptNet
                Emmbedings
6              sentence = searchEdgeInKELMCorpus(Edge)
7              if sentence ≠ null then
8                  corpus.add(sentence)
9              end
10             if sentence == null then
11                 sentence = searchSentenceWithEdgeInTopic(Edge)
12                 if sentence ≠ null then
13                     corpus.add(sentence)
14                 end
15                 if sentence == null then
16                     corpus.add(edge)
17                 end
18             end
19         end
20     end
21 end
22
23
```

Output: Corpus

Appendix B

Performance of Top2Label

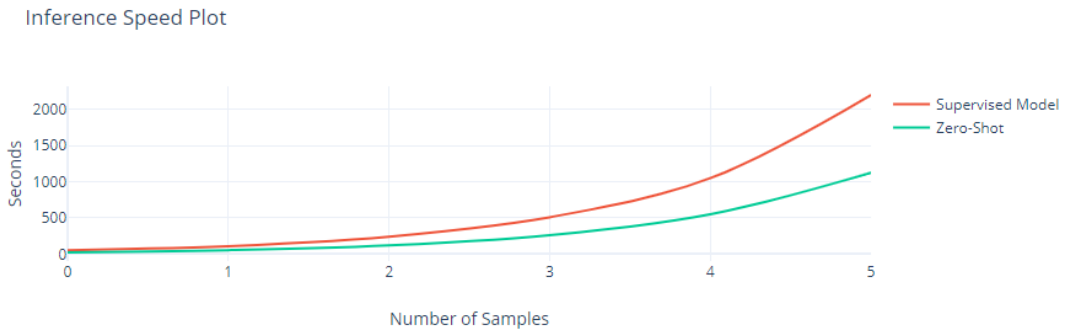

Figure B.1: The above image plot the performance of our Zero-Shot One Word Labelling Approach against the supervised automatic topic labelling approach in the baselines. We can see the performance difference in predicting labels by both approaches. In green is the proposed algorithm, and in red are supervised algorithms.

Appendix C

Change in Coverage and Relevance based on Top Words of Topic

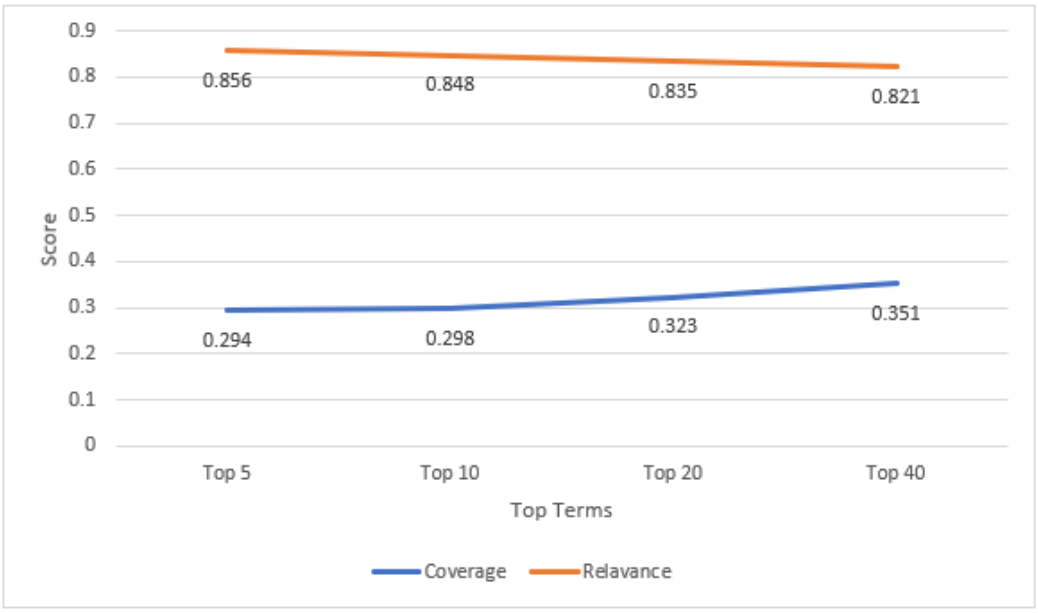

Figure C.1: The above image plot displays how Coverage and Relevance scores changes due to a change in the inclusion of the number of top n terms (based on cosine similarity scores) from the topic and define the relationship between top n terms, Coverage and Relevance evaluation metric.