International Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume-12, Special Issue-1, March-2024
https://doi.org/10.55524/CSISTW.2024.12.1.73
Computing for Sustainable Innovation: Shaping Tomorrow's World (CSISTW-2024) on 16th-17th February, 2024
Organized by B.N. College of Engineering and Technology, Lucknow, India

Text Summarization Using Deep Learning Model

Mamta Prajapati[1], and Vivek Rai[2]

[1] M. Tech Scholar, Department of CSE, B.N. College of Engineering & Technology, Lucknow, Uttar Pradesh, India
[2]Assistant Professor, Department of CSE, B.N. College of Engineering & Technology, Lucknow, Uttar Pradesh, India

Correspondence should be addressed to Mamta Prajapati; prajapatimamta533@gmail.com

ABSTRACT- In our study on text summarization, we have introduced a sophisticated neural network architecture, employing a sequence-to-sequence model with embedding's, Long Short-Term Memory units, and attention mechanisms. The primary goal is to effectively capture intricate patterns and dependencies within sequential data. To enhance versatility, we have implemented a dual-input configuration to handle fixed and variable-length sequences, with embedding layers transforming input sequences into dense vectors, capturing semantic relationships.

Our strategic placement of LSTM units addresses the challenge of handling input sequences of different lengths by incorporating sequential processing capabilities. The concatenation layer integrates both inherent sequential patterns and dynamically attended features. The final time-distributed layer produces sequences with a rich 1989-dimensional output space, reflecting the complexity of information our model aims to capture. Throughout the training process over multiple epochs, we observed decreasing trends in both training and validation losses. Early stopping at epoch 19, triggered by an increase in validation loss after epoch 17, marked the successful conclusion. At this point, we achieved the lowest training loss of 1.7070, showcasing the model's ability to learn underlying patterns. Our proposed model's success in minimizing the disparity between predicted and actual key phrases, coupled with careful consideration of crucial parameters, positions it as a promising approach for key phrase generation.

KEYWORDS- Text Summarization, LSTM, RNN, ANN

I. INTRODUCTION

In the realm of scientific literature, the inclusion of a list of key phrases serves as a crucial attribute, offering a concise representation of a text's contents. Key phrases play a pivotal role in enhancing a paper's visibility and citation count, making their qualitative selection a significant aspect [2, 17]. Within the scientific literature domain, the incorporation of a list of key phrases holds pivotal significance, providing a succinct portrayal of a text's content. Many existing methods for key phrase extraction involve a systematic procedure of culling words from the input text, ranking these candidates, and subsequently selecting the top N key phrases, where N is determined by the model rather than user input. However, conventional methods directly extracting key phrases from the text often fall short, capturing only those phrases explicitly present in the text itself.

In practical terms, key phrases function as abstractive summaries of the text, encompassing not only exact matches but also hypernyms and paraphrased sentences from the source material. This distinction underscores the importance of investigating the applicability of abstractive text summarization methods in generating multiple key phrases as a sequence. In contrast to traditional approaches, those rooted in abstractive text summarization exhibit noteworthy characteristics: (a) the determination of N is a model-driven value, eliminating the need for user input; (b) it takes into account both semantic and syntactic portion of the input text; (c) the proposed key phrases extend beyond words or phrases explicitly found in the input text.

Despite the significance of abstractive text summarization in key phrase extraction, a limited number of research have explored the generation of keywords using this approach [9]. This research gap necessitates a critical exploration of the effectiveness of such models in this domain.

Moreover, an unexplored dimension lies in the comparison of different ordering plan for concatenating target key phrases. The organization of these target phrases can significantly influence the overall efficacy of the key phrase extraction process. Existing literature lacks comprehensive studies systematically comparing and evaluating the impact of various ordering strategies on the generation of key phrases. Such a comparative analysis could provide valuable insights into optimizing the key phrase extraction process.

Remaining of the paper describe as follows: section II involves the related work, section III involves the method and material, section IV contains result and discussion and section V contains conclusion.

II. RELATED WORK

In the exploration of detecting depressive symptoms, numerous studies have engaged in the examination of user interviews and the analysis of social media content. Within the realm of Key phrase Generation (KPG), the process involves mapping a textual sequence to a set of sequences known as key phrases. This task is commonly transformed into a seq2seq (sequence-to-sequence) framework, employing methodologies such as ONE2ONE or ONE2SEQ.

In the ONE2ONE paradigm, the training process involves pairing each key phrase using the provided input text, acting as the target output. During testing, a diverse array of key phrases is chosen from different beams in the generation process. n contrast, during training, ONE2SEQ

employs the roster of key phrases as the target text, while at the testing phase, key phrases are chosen either from the highest-scoring beam or synthesized from the leading k beams.

The primary goal of key phrase extraction is to recognize a set of phrases closely linked to the main topics addressed within a provided document [15]. Numerous studies have been published over the years, presenting a variety of unsupervised and supervised approaches to key phrase extraction. Unsupervised methodologies typically encompass multiple stages, including selecting candidate words or n-grams based on specific characteristics, ranking these candidates, and forming key phrases by choosing the highest-ranked words [25]. Statistical and graph-based methods stand out as the predominant techniques used in unsupervised key phrase extraction. Analytical methods like TFIDF, KPMiner [12], and YAKE! [8] utilize statistical features within the text to identify pivotal words. In contrast, graph-oriented techniques entail establishing a document graph, where candidate phrases serve as nodes interconnected by relationships as edges. Examples of noteworthy graph-based methodologies encompass TextRank [30], TopicRank [6], and PositionRank [13].

Chowdhury et al. [10] demonstrated the competitive efficacy of finely adjusted BART [20] in generating keyphrases, contrasting it with prevailing neural models designed for keyphrase extraction. Their approach involved ranking the keyphrases produced, choosing a predetermined count of keyphrases per input text, and independently computing the F1-score for keyphrases found in the input text, along with the Recall for keyphrases not present in the text.

Meng et al. [25] explored the consequences of various approaches in combining target keyphrases, particularly within the One2Seq framework. Their results highlighted the significance of the organization of merged target expressions, revealing that the best outcomes were attained when organizing target key phrases according to their initial appearances in the input text. In essence, these varied methodologies and models collectively contribute to the evolving landscape of key phrase extraction, highlighting continuous efforts to enhance precision, efficiency, and adaptability in this crucial text summarization [23].

Yuan et al. [25] presented a method relying on ONE2SEQ for producing varied key phrases, providing influence over the quantity of results by managing decoder concealed states. In a different approach, Chen et al. [23] and Chan et al. [11] introduced methodologies for Key phrase Generation based on reinforcement learning (RL). They proposed a multi-task learning framework, simultaneously acquiring knowledge from both extraction and generation based models for key phrases and incorporating information retrieval techniques. However, they did not present separate results for present and absent key phrases, suggesting potential avenues for further investigation in this aspect.

Ye et al. [46] proposed a novel KPG approach grounded in graph neural networks (GNNs). To summarize, a multitude of studies has explored the identification of depressive symptoms through user interviews and the analysis of social media content. In the realm of Keyphrase Generation, the ONE2ONE and ONE2SEQ methodologies play pivotal roles, with CopyRNN serving as a foundational model. Researchers have extended and refined these models, exploring the integration of document structure information and novel methodologies, such as reinforcement learning and graph neural networks, to enhance the generation of keyphrases. The distinction between present and absent keyphrases has become a critical consideration in evaluating the effectiveness of these approaches, prompting ongoing avenues for further exploration and refinement.

KeyGraph [26] emerged as the pioneering model in the realm of graph-based approaches crafted for extracting keyphrases. In this structure, regularly co-occurring expressions were linked to construct a graph, subsequently divided into subgraphs through clustering. The importance of each potential phrase was then evaluated based on statistical details derived from subgraphs. In the publication [27], the author commenced the procedure by calculating edge weights, utilizing semantic relatedness at the phrase level grounded in Wikipedia. Subsequently, they applied the community detection algorithm [28] to recognize dense subgraphs, with key phrases originating from the most essential subgraphs. Likewise, Liu et al. [29] formulated a word graph and grouped words based on semantic distances obtained from word co-occurrence frequency or Wikipedia statistics. Key phrases were then selected from the noun expressions expanded from cluster centers. TextRank, inspired by PageRank (Mihalcea and Tarau [30]; Page et al.,[31]), conducted iterative importance propagation on a co-occurrent word graph. Expanding upon this method, Danesh, Sumner, and Martin [32] adapted TextRank by employing phrases as nodes in the graph. Various characteristics, such as statistical features (phrase frequency and length, word co-occurrence frequency [33], and positional information [34], were investigated to adjust edge weights. The authors of the paper [35] have done the augmentation using the single-document word graph with same type of documents and a citation network, respectively, for incorporating more contexts. In an effort to align keyphrases more closely with document topics, researchers introduced topic information into refined graph-based models. To address computational costs, TPR was extended into Single Topical PageRank [39] and SalienceRank [40], both performing PageRank once for each document. SalienceRank, in particular, could extract not only topic-specific but also corpus-correlated keyphrases. In contrast to LDA-based studies, Bougouin, Boudin, and Daille [41] introduced TopicRank, clustering similar phrases into topics and constructing graph on the basis of topic for PageRank. They then selected the higly representative phrases from every topic as keyphrases. The authors of the paper [42] represented “candidate phrases” and topics in a unified graph, leveraging their mutual reinforcement to enhance candidate ranking.

With the successful integration of PLMs into various text summarization [11]; [20]; [42]; [43], have begun incorporating PLMs into methods for key phrase extraction and generation. PLMs have found application in unsupervised keyphrase extraction [46], keyphrase extraction through sequence labeling and keyphrase generation using sequence-to-sequence (seq2seq) generation [29], [44]; [45]; 46]. However, these studies

currently exist at an applied level, lacking a thorough evaluation of the advantages or limitations of PLMs and the impact of various decision choices when employing PLMs. The absence of well-established results raises several overarching questions in the keyphrase generation research community: When is the optimal time to leverage PLMs? Are we selecting the most appropriate PLMs for the task at hand? Which PLMs should be chosen under constraints related to annotated data or computational resources? While previous research has demonstrated the benefits of PLMs in zero-shot multilingual and low-resource keyphrase generation [47].

III. METHOD AND MATERIALS

A. Dataset

The dataset under consideration comprises reviews of fine foods sourced from Amazon, spanning a period of over 10 years and encompassing approximately 500,000 reviews. To streamline the training time of our model, a judicious decision has been made to extract a representative sample of 100,000 reviews. Prior to delving into the model building phase, it is imperative to undertake fundamental preprocessing steps to ensure the cleanliness and coherence of the text data. Utilizing unprocessed and cluttered text data could potentially lead to adverse outcomes. Thus, the preprocessing tasks involve converting all text to lowercase, removing HTML tags, implementing contraction mapping, eliminating possessive forms ('s), discarding text within parentheses, ridding the text of punctuations and special characters, excluding stop words, and eliminating short words. These measures collectively contribute to refining the dataset and creating a more conducive environment for subsequent model development.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 88421 entries, 0 to 99999
Data columns (total 10 columns):
Id                      88421 non-null int64
ProductId               88421 non-null object
UserId                  88421 non-null object
ProfileName             88421 non-null object
HelpfulnessNumerator    88421 non-null int64
HelpfulnessDenominator  88421 non-null int64
Score                   88421 non-null int64
Time                    88421 non-null int64
Summary                 88421 non-null object
Text                    88421 non-null object
dtypes: int64(5), object(5)
memory usage: 7.4+ MB
```

Figure 1: Dataset Descriptions

Our analysis will focus on examining the length of both reviews and summaries, providing us with a comprehensive understanding of the distribution of text lengths. This exploration is crucial in determining the optimal sequence length, aiding us in establishing a maximum length for our sequences as shown in Figure 2.

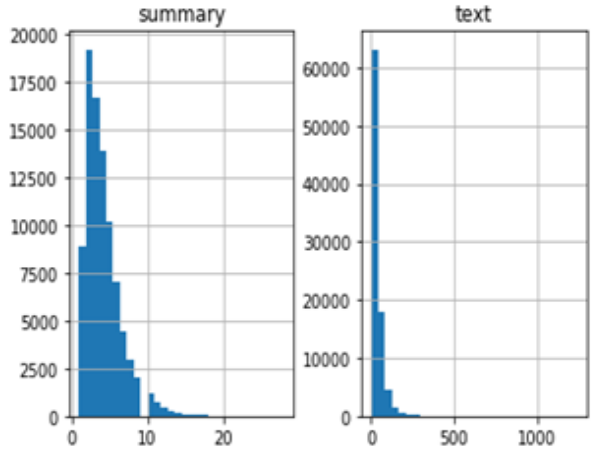

Figure 2: Distribution of Sequences

B. Proposed Model

We will leverage the AutoModel class from the Transformers library, specifically designed for convenient model loading, by using the from_pretrained() method to load the pre-trained DistilBERT model. The availability of a GPU is checked using PyTorch, and the model is configured to run on the GPU if present, else it defaults to CPU.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 30)	0	
embedding (Embedding)	(None, 30, 100)	844000	input_1[0][0]
lstm (LSTM)	[(None, 30, 300), (N	481200	embedding[0][0]
input_2 (InputLayer)	(None, None)	0	
lstm_1 (LSTM)	[(None, 30, 300), (N	721200	lstm[0][0]
embedding_1 (Embedding)	(None, None, 100)	198900	input_2[0][0]
lstm_2 (LSTM)	[(None, 30, 300), (N	721200	lstm_1[0][0]
lstm_3 (LSTM)	[(None, None, 300),	481200	embedding_1[0][0] lstm_2[0][1] lstm_2[0][2]
attention_layer (AttentionLayer	[(None, None, 300),	180300	lstm_2[0][0] lstm_3[0][0]
concat_layer (Concatenate)	(None, None, 600)	0	lstm_3[0][0] attention_layer[0][0]
time_distributed (TimeDistribut	(None, None, 1989)	1195389	concat_layer[0][0]

Total params: 4,823,389
Trainable params: 4,823,389
Non-trainable params: 0

Figure 3: Proposed Model Architecture

We have used neural network architecture represents a sophisticated sequence-to-sequence model designed for a text Summarization, with a particular focus on handling sequences of varying lengths. Comprising multiple layers, the model incorporates embedding's, Long Short-Term Memory (LSTM) units, and attention mechanisms to capture intricate patterns and dependencies within sequential data.

Our proposed model begins with two distinct input layers, namely input_1 and input_2. The former is tailored to handle sequences of fixed length (30), while the latter is designed to manage sequences of variable length. This dual-input configuration suggests the model's versatility in accommodating diverse input scenarios.

Following the input layers, the architecture incorporates two embedding layers, denoted as embedding and embedding_1. These layers play a pivotal role in transforming the input sequences into dense vectors of size 100. The embedding's serving as a means to represent words or tokens in a continuous vector space, capturing semantic relationships and contextual nuances.

The subsequent layers involve the application of LSTM units, known for their ability to capture long-range dependencies in sequential data. The first LSTM layer (lstm) processes the embedded sequences from input_1, generating output sequences of length 30 with 300 dimensions. This layer also produces hidden states and cell states, facilitating memory retention and information flow. The second LSTM layer (lstm_1) builds upon the output of the first LSTM layer, producing output sequences of the same length (30) with 300 dimensions. These recurrent layers contribute to the model's capacity to understand and encode sequential patterns.

Simultaneously, the third LSTM layer (lstm_2) processes the embedded sequences from input_2, generating output sequences of length 30 with 300 dimensions. The subsequent LSTM layer (lstm_3) takes the output of lstm_2 and produces output sequences of variable length with 300 dimensions. This sequential processing is critical for handling input sequences of different lengths, showcasing the model's adaptability.

Introducing an attention mechanism, the attention_layer calculates attention weights between the sequences processed by lstm_2 and lstm_3. Attention mechanisms enhance the model's ability to focus on specific parts of the input sequence, enabling it to weigh the importance of different elements dynamically.

The architecture further incorporates a concatenation layer (concat_layer), which combines the output sequences from lstm_3 with the attention-weighted sequences from the attention layer. This fusion of information ensures that the model integrates both the inherent sequential patterns and the dynamically attended features.

The final layer, time_distributed, applies a dense layer to each time step of the concatenated sequences, resulting in sequences of variable length (None) with 1989 dimensions. The choice of 1989 dimensions implies the model's output space, indicating the richness and complexity of the information it aims to capture and generate.

The model summary provides valuable insights into the overall architecture, detailing the total number of trainable parameters (4,823,389) and non-trainable parameters. The extensive number of parameters underscores the model's capacity to learn intricate relationships and patterns from the input data. As for the loss function, the model employs sparse categorical cross-entropy. This choice is particularly relevant as it dynamically converts the integer sequence to a one-hot vector during training, addressing potential memory constraints. This adaptive mechanism allows the model to effectively optimize its parameters without the need for explicit conversion of target sequences. The concept of early stopping is implemented using the Early Stopping callback in neural network training. The monitor parameter specifies the metric to be monitored, mode defines whether to minimize or maximize the metric, verbose controls the logging, and patience determines the number of epochs with no improvement before stopping. The model trains with a batch size of 128 and validation on a 10% holdout set.

IV. RESULT AND DISCUSSION

We have conducted the training process on our neural network, which involved multiple epochs, with each epoch iterating over a dataset consisting of 41,346 training samples and 4,588 validation samples. Employing the sparse categorical cross-entropy loss function, our aim was to minimize the disparity between predicted and actual key phrases. The training commenced with an initial loss of 2.8152, progressively decreasing in subsequent epochs. To prevent overfitting and enhance computational efficiency, we implemented early stopping, monitoring the validation loss to cease training when no improvement was observed for two consecutive epochs. In this specific instance, the training concluded after 19 epochs due to the early stopping criterion being met. The diminishing trend in both training and validation losses signified the model's learning progress, successfully capturing underlying patterns in the dataset. Crucial model parameters, including batch size, optimizer, and learning rate, played pivotal roles in shaping the training dynamics, ensuring convergence towards a well-performing key phrase generation model. From the Figure 4 , it is evident that the validation loss increased for 2 successive epochs after epoch 17. Consequently, the training process was halted at epoch 19 to prevent overfitting. From the provided data, at the end of epoch 19, the training loss was 1.7070, and the validation loss was 2.0398. This indicates that the training loss achieved its lowest value of 1.7070 during the training process.

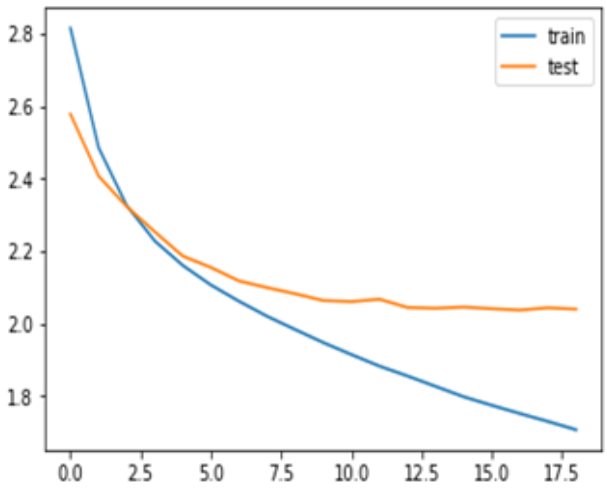

Figure 4: Loss graph Vs Epochs

V. CONCLUSION AND FUTURE WORK

In our study, we have introduced a robust neural network architecture specifically designed for text summarization. Our model utilizes a sophisticated sequence-to-sequence architecture, incorporating multiple layers such as embedding's, LSTM units, and attention mechanisms. The goal is to capture intricate patterns and dependencies within sequential data effectively.

To enhance the versatility of our model, we have implemented a dual-input configuration with distinct input layers for handling fixed and variable-length sequences. The embedding layers play a pivotal role in transforming input sequences into dense vectors, capturing semantic relationships and contextual nuances.

Our strategic placement of LSTM units in the architecture demonstrates their proficiency in capturing long-range dependencies in sequential data. Notably, we have addressed the challenge of handling input sequences of different lengths by incorporating sequential processing capabilities in the LSTM layers.

The introduction of an attention mechanism further improves the model's ability to dynamically focus on specific parts of input sequences. The attention layer calculates weights between sequences, enabling the model to weigh the importance of different elements during processing.

The concatenation layer combines output sequences from LSTM layers with attention-weighted sequences, ensuring the integration of both inherent sequential patterns and dynamically attended features. The final time-distributed layer produces sequences of variable length with a rich 1989-dimensional output space, reflecting the complexity of information our model aims to capture and generate.

In our comprehensive model summary, we highlight insights into the architecture, emphasizing the extensive number of trainable parameters (4,823,389), and the selection of sparse categorical cross-entropy as the loss function. The implementation of early stopping, with a specified monitoring metric, a batch size of 128, and validation on a 10% holdout set, ensures efficient training dynamics and prevents overfitting.

Throughout the training process conducted over multiple epochs, we observed a decreasing trend in both training and validation losses. Early stopping, triggered by an increase in validation loss for two successive epochs after epoch 17, resulted in the conclusion of training at epoch 19. At this point, we achieved the lowest training loss value of 1.7070, highlighting our model's ability to learn and capture underlying patterns in the dataset.

The success of our model in minimizing the disparity between predicted and actual key phrases, coupled with the careful consideration of crucial parameters, positions it as a promising approach for key phrase generation. Looking ahead, we envision future refinements, extensive evaluations, and applications in diverse natural language processing tasks, contributing to the continuous evolution of advanced language models.

CONFLICT OF INTERREST

The authors declare that they have no conflict of interest.

REFERENCES

[1] Z. Alami Merrouni, B. Frikh, and B. Ouhbi, “Automatic keyphrase extraction: A survey and trends,” J. Intell. Inform. Syst. 54, 391–424 (2020). http://dx.doi.org/10.1007/s10844-019-00558-9 2

[2] N. Ale Ebrahim, H. Salehi, M. A. Embi, F. Habibi, H. Gholizadeh, S. M. Motahar, and A. Ordi, “Effective strategies for increasing citation frequency,” Int. Educ. Studies 6 (11), 93–99 (2013). http://dx.doi.org/10.5539/ies.v6n11p93

[3] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, “SemEval 2017 task 10: ScienceIE-extracting keyphrases and relations from scientific publications,” in Proceedings of the 11th International Workshop on Semantic Evaluation SemEval2017 (2017), pp. 546–555. http://dx.doi.org/10.18653/v1/s17-2091

[4] S. Bird, “NLTK: The natural language toolkit,” in Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions (2006), pp. 69–72. http://dx.doi.org/10.3115/1225403.1225421 5.

[5] F. Boudin, “PKE: An open source python-based keyphrase extraction toolkit,” in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations (2016), pp. 69–73

[6] A. Bougouin, F. Boudin, and B. Daille, “TopicRank: Graph-based topic ranking for keyphrase extraction,” in Proceedings of International Joint Conference on Natural Language Processing IJCNLP (2013), pp. 543–551.

[7] I. Cachola, K. Lo, A. Cohan, and D. S. Weld, “TLDR: Extreme summarization of scientific documents,” in Proceedings of the Conference on Findings of the Association for Computational Linguistics: EMNLP 2020 (2020), pp. 4766–4777. http://dx.doi.org/10.18653/v1/2020.findings-emnlp.428

[8] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, “YAKE! Keyword extraction from single documents using multiple local features,” Inform. Sci. 509, 257–289 (2020). http://dx.doi.org/10.1016/j.ins.2019.09.013 E. Cano and O. Bojar, ¸ “Keyphrase generation: A text summarization struggle,” in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2019), Vol. 1, pp. 666–672. http://dx.doi.org/10.18653/v1/N19-1070

[9] M. F. M. Chowdhury, G. Rossiello, M. Glass, N. Mihindukulasooriya, and A. Gliozzo, “Applying a generic sequence-to-sequence model for simple and effective keyphrase generation,” arXiv: 2201.05302 (2022). http://dx.doi.org/10.48550/ARXIV.2201.05302

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (2019), Vol. 1, pp. 4171– 4186. http://dx.doi.org/10.18653/v1/N19-1423
LOBACHEVSKII JOURNAL OF MATHEMATICS Vol. 44 No. 1 2023 APPLYING TRANSFORMER-BASED TEXT 135

[11] S. R. El-Beltagy and A. Rafea, “KP-Miner: A keyphrase extraction system for English and Arabic documents,” Inform. Syst. 34, 132–144 (2009). http://dx.doi.org/10.1016/j.is.2008.05.002 13.

[12] C. Florescu and C. Caragea, “PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (2017), Vol. 1, pp. 1105–1115. http://dx.doi.org/10.18653/v1/P17-1102

[13] M. Grootendorst, “KeyBERT: Minimal keyword extraction with BERT,” Zenodo (2020). http://dx.doi.org/10.5281/zenodo.4461265

[14] K. S. Hasan and V. Ng, “Automatic keyphrase extraction: A survey of the state of the art,” in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (2014), Vol. 1, pp. 1262–1273. http://dx.doi.org/10.3115/v1/P14-1119

[15] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (2003), pp. 216–223. http://dx.doi.org/10.3115/1119355.1119383
[16] D. V. Ilango and D. S. M. Kumar, "Factors for improving the research publications and quality metrics," Int. J. Civil Eng. Technol. 8, 477–496 (2017).
[17] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, "CTRL: A conditional transformer language model for controllable generation," arXiv: 1909.05858 (2019). http://dx.doi.org/10.48550/arxiv.1909.05858
[18] M. Krapivin, A. Autaeu, and M. Marchese, "Large dataset for keyphrases extraction" (2009)
[19] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7871–7880. http://dx.doi.org/10.18653/v1/2020.acl-main.703
[20] Y. Lim, D. Seo, and Y. Jung, "Fine-tuning BERT models for keyphrase extraction in scientific articles," J. Adv. Inform. Technol. Converg. 10 (1), 45–56 (2020). http://dx.doi.org/10.14801/jaitc.2020.10.1.45
[21] C. Y. Lin, "Rouge: A package for automatic evaluation of summaries," in Text Summarization Branches Out (2004), pp. 74–81.
[22] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," arXiv: 1907.11692 (2019). http://dx.doi.org/10.48550/arXiv.1907.11692
[23] O. Medelyan, E. Frank, and I. H. Witten, "Human-competitive tagging using automatic keyphrase extraction," in Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (2009), pp. 1318–1327. http://dx.doi.org/10.3115/1699648.1699678
[24] R. Meng, X. Yuan, T. Wang, P. Brusilovsky, A. Trischler, and D. He, "Does order matter? An empirical study on generating multiple keyphrases as a sequence," arXiv: 1909.03590 (2019). http://dx.doi.org/10.48550/ARXIV.1909.03590
[25] Ohsawa, Y., Benson, N. E., & Yachida, M. (1998, April). KeyGraph: Automatic indexing by co-occurrence graph based on building construction metaphor. In Proceedings IEEE International Forum on Research and Technology Advances in Digital Libraries-ADL'98- (pp. 12-18). IEEE.
[26] Grineva, M., Grinev, M., & Lizorkin, D. (2009, April). Extracting key terms from noisy and multitheme documents. In Proceedings of the 18th international conference on World wide web (pp. 661-670).
[27] Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. Physical review E, 69(2), 026113.
[28] Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009, August). Clustering to find exemplar terms for keyphrase extraction. In Proceedings of the 2009 conference on empirical methods in natural language processing (pp. 257-266).
[29] Mihalcea, R., & Tarau, P. (2004, July). Textrank: Bringing order into text. In Proceedings of the 2004 conference on empirical methods in natural language processing (pp. 404-411).
[30] Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). The pagerank citation ranking: Bring order to the web. Technical report, stanford University.
[31] Danesh, S., Sumner, T., & Martin, J. H. (2015, June). Sgrank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. In Proceedings of the fourth joint conference on lexical and computational semantics (pp. 117-126).
[32] Wan, X., & Xiao, J. (2008, July). Single document keyphrase extraction using neighborhood knowledge. In AAAI (Vol. 8, pp. 855-860).
[33] Florescu, C., & Caragea, C. (2017, July). Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers) (pp. 1105-1115).
[34] Gollapalli, S. D., & Caragea, C. (2014, June). Extracting keyphrases from research papers using citation networks. In Proceedings of the AAAI conference on artificial intelligence (Vol. 28, No. 1).
[35] Wan, X., & Xiao, J. (2008, July). Single document keyphrase extraction using neighborhood knowledge. In AAAI (Vol. 8, pp. 855-860).
[36] Vega-Oliveros, D. A., Gomes, P. S., Milios, E. E., & Berton, L. (2019). A multi-centrality index for graph-based keyword extraction. Information Processing & Management, 56(6), 102063.
[37] Liu, Z., Huang, W., Zheng, Y., & Sun, M. (2010, October). Automatic keyphrase extraction via topic decomposition. In Proceedings of the 2010 conference on empirical methods in natural language processing (pp. 366-376).
[38] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan), 993-1022.
[39] Sterckx, L., Demeester, T., Deleu, J., & Develder, C. (2015, May). Topical word importance for fast keyphrase extraction. In Proceedings of the 24th International Conference on World Wide Web (pp. 121-122).
[40] Teneva, N., & Cheng, W. (2017). Salience rank: Efficient keyphrase extraction with topic modeling.
[41] Bougouin, A., Boudin, F., & Daille, B. (2013, October). Topicrank: Graph-based topic ranking for keyphrase extraction. In International joint conference on natural language processing (IJCNLP) (pp. 543-551).
[42] Boudin, F. (2018). Unsupervised keyphrase extraction with multipartite graphs. arXiv preprint arXiv:1803.08721.
[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-totext transformer. Journal of Machine Learning Research, 21(140):1–67.
[44] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440– 8451, Online. Association for Computational Linguistics.
[45] Md Faisal Mahbub Chowdhury, Gaetano Rossiello, Michael Glass, Nandana Mihindukulasooriya, and Alfio Gliozzo. 2022. Applying a generic sequence-to-sequence model for simple and effective keyphrase generation.
[46] Mayank Kulkarni, Debanjan Mahata, Ravneet Arora, and Rajarshi Bhowmik. 2022. Learning rich representation of keyphrases from text. In Findings of the Association for Computational Linguistics: NAACL 2022, pages 891–906, Seattle, United States. Association for Computational Linguistics.
[47] Yifan Gao, Qingyu Yin, Zheng Li, Rui Meng, Tong Zhao, Bing Yin, Irwin King, and Michael Lyu. 2022. Retrieval-augmented multilingual keyphrase generation with retriever-generator iterative training. In Findings of the Association for Computational Linguistics: NAACL 2022,

pages 1233–1246, Seattle, United States. Association for Computational Linguistics.

[48] Di Wu, Wasi Uddin Ahmad, Sunipa Dev, and Kai-Wei Chang. 2022. Representation learning for resourceconstrained keyphrase generation. In Findings of the Association for Computational Linguistics: EMNLP 2022.

[49] Guangzhen Zhao, Guoshun Yin, Peng Yang, and Yu Yao. 2022. Keyphrase generation via soft and hard semantic corrections. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 7757–7768, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.