Job Descriptions Keyword Extraction using Attention based Deep Learning Models with BERT

Rishit Dagli [1], Ali Mustufa Shaikh [2], Hussain Mahdi [2], and Sameer Nanivadekar [2]

[1]Narayana Junior College
[2]Affiliation not available

October 30, 2023

Abstract

In this paper, we focus on creating a keywords extractor especially for a given job description job-related text corpus for better search engine optimization using attention based deep learning techniques. Millions of jobs are posted but most of them end up not being located due to improper SEO and keyword management. We aim to make this as easy to use as possible and allow us to use this for a large number of job descriptions very easily. We also make use of these algorithms to screen or get insights from large number of resumes, summarize and create keywords for a general piece of text or scientific articles. We also investigate the modeling power of BERT (Bidirectional Encoder Representations from Transformers) for the task of keyword extraction from job descriptions. We further validate our results by providing a fully-functional API and testing out the model with real-time job descriptions.

Job Descriptions Keyword Extraction using Attention based Deep Learning Models with BERT

Dr. Hussain Falih Mahdi
dept. Computer Engineering, Faculty of Engineering
University of Diyala
Baqubah, Iraq
Hussain.mahdi@ieee.org

Rishit Dagli
Student
Narayana Junior College
Mumbai, India
rishit.dagli@gmail.com

Ali Mustufa
Student, dept. Information Technology
A P Shah Institute of Technology
Mumbai, India
iali@ieee.org

Dr. Sameer Nanivadekar
dept. Information Technology
A P Shah Institute of Technology
Thane, India
deanadmin@apsit.edu.in

***Abstract*—In this paper, we focus on creating a keywords extractor especially for a given job description job-related text corpus for better search engine optimization using attention based deep learning techniques. Millions of jobs are posted but most of them end up not being located due to improper SEO and keyword management. We aim to make this as easy to use as possible and allow us to use this for a large number of job descriptions very easily. We also make use of these algorithms to screen or get insights from large number of resumes, summarize and create keywords for a general piece of text or scientific articles. We also investigate the modeling power of BERT (Bidirectional Encoder Representations from Transformers) for the task of keyword extraction from job descriptions. We further validate our results by providing a fully-functional API and testing out the model with real-time job descriptions.**

***Index Terms*—keyword extractor, job descriptions, SEO, attention mechanism, BERT, screening, summarization**

I. Introduction

A valuable concept for searching and categorizing job descriptions is the keyword, a short set of words (one or few) which represent concepts and offer a compact document's content representation. Perfectly, keywords are representing in compressed form the important document content [1].

Extracting keywords from a general text is a difficult activity as we need some idea of how these keywords are related to the paragraph and if they are trending for better Search Engine Optimization (SEO) [2, 3]. We aim to solve this problem using a state of the art algorithm which once trained can be used for multiple purposes such as Resume Screening, Keywords for Job Description, etc. SEO can heavily impact a company's ranking on search engines [4, 5].

We propose a Machine Learning system that can analyze a text corpus from a job description and output some suggested keywords which could be applied specifically using topic modelling techniques [6]. One of the main objectives is to facilitate easy, efficient and meaningful job search. Millions of jobs are posted but are usually not accessible due to improper SEO. It is also practically impossible to manually list down keywords for a given job description when they are in quantity. The proposed system is more flexible and versatile than the traditional manual way of doing so. The Scope is not just restricted to SEO, once the algorithm is developed, it can be used for a wide range of applications including screening resumes, shortlisting perfect candidates, suggesting changes, automatically adding tags (for better SEO) and filtering job listings etc [7].

We automate the process of extracting relative keywords from job descriptions allowing for job postings to get better relevant SEO rankings. We use the same model and concepts and apply them to many more problem statements like automating resume checking, adding tags for a given block of text, etc. Most organizations rely on search engines algorithms to get their job postings listed or manually put keywords to rank it higher which is pretty time consuming. We automate this for the organizations so that it can be directly integrated with their existing system with little or no changes.

We also aim to make the process hassle-free and very easy to implement by providing a simple to use REST API and gRPC servers which can allow this to be integrated into existing system very easily by organizations. To further make this easily accessible by administrators and HRs, we also provide them with a UI where they could simply paste their job descriptions and generate the relevant keywords making the process a lot easier for them and save human efforts as well.

Several neural models comprised of pre- trained word as task agnostic embedding layer EL and neural architecture being task-specific were proposed for the keyword as original or key-phrase extraction problem [8–11] [1] whereas such models

The authors contributed equally to this work.

[1]Due to limited space, we do not list all of the existing works here, please refer to the surveys [12, 13] for more related papers.

improvement measured through the correctness or score of F1 has arrived to a bottleneck. One of the reasons in which the EL as task-agnostic is typically of layer linearly prepared along with Word2Vec [14] or GloVe [15], just offers context-independent word- level characteristics that is inadequate to capture the dependencies as complex semantic in such sentence.

In this paper we also investigate the modeling power of BERT (Bidirectional Encoder Representations from Transformers), one of the most popular pre-trained language model with Transformers [16], on the task of keyword extraction from job descriptions. We also make comparisons between BERT-based models and and those keeping BERT component fixed. We find out that the BERT-based models perform a lot better than those trained keeping BERT component fixed for the downstream tasks. So, we perform task-specific fine-tuning allowing us to make the best use of the BERT strengths for performance improvement [17]

We also propose a way to keep updating the model behind this in real-time through user feedback. We further also provide ways in which this model and data apart from adding keywords to job descriptions can also be used to derive insights from resumes or automatically add tags for a certain block of text. When performing this on a large number of similar documents like resumes or scientific articles we also try to extract keywords from particular sections to prevent analyzing the full text of an article requires more disk space and the analysis needs more computational capacity [18]. Apart from this, with enough information we could also use these models and data to perform summarization of articles [19], job descriptions and general text too [20].

The remainder of this article is organized as follows. In II section the system design and the model architecture is presented. In III section the data used for this paper and our collection methods are presented. The IV section experimental results of the model and the API are presented. The V section concludes the article and gives future works.

II. System Design and Models

The main purpose of the proposed system is to generate keywords from a corpus of text and to also eliminate the physical keywords extraction and other hassles and make the system completely hassle-free. The architecture as overall for the adopted model is represented in Fig. 1.

A. BERT as an embedding layer

We first perform word embedding of the job description text to represent them as low-dimensional vectors. A popular implementation of embeddings word are the Word2Vec [14], GloVe [15], and the models of fastText [21]. However, a major problem we face with these traditional models or ELs is that we get a single context-independent representation for each token which results in losing the correct sense of the token. This also proves to be highly inadequate to capture the dependencies as complex semantic in a sentence which would be very much needed in this use case. To preserve the context

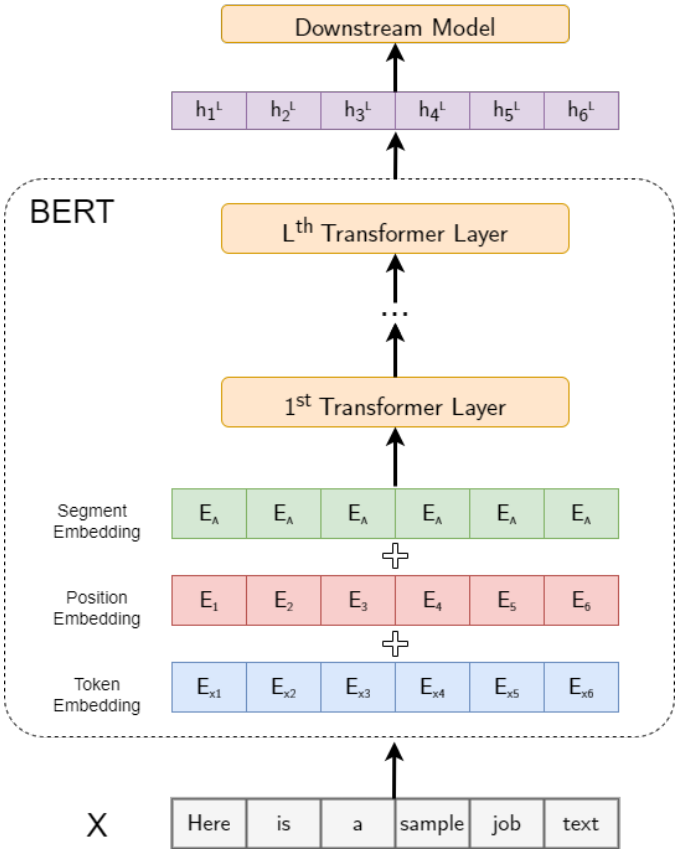

Fig. 1. Topic scores for an example job description.

in which the token is used and perform embedding in that sense we make use of the BERT model [22].

The model architecture of BERT is based on a Transformer encoder of multiple layers that was applied originally through Vaswani et al. [23] . Devlin et al. [22] presented the BERT Transformer according to the utilization of self-attention as bidirectional. Such a mechanism as bidirectional eliminates the limitations that self-attention can just integrate the one side context: right or left. BERT makes use of Transformer which eschews recurrence and is based solely on attention mechanisms which have become an integral part [24] which learns relations as contextual between text sub-words or words [23]. In its vanilla form, the Transformer includes two separate mechanisms, an encoder which reads the input of text and a decoder that creates task prediction.

Compared to the traditional layers embedded that just offers representation as context-independent being single for every token, the BERT EL considers the sentence as input and figures the representations of token-level utilizing the data from the whole sentence [25]. Given the input token sequence $x = \{x_1, \cdots, x_T\}$ of length T, we firstly employ the BERT component with L transformer layers to calculate the corresponding contextualized representations $H^L = \{h_1^L, \cdots, h_T^L\}$ for the input tokens. Specifically, the representations $H^l = \{h_1^l, \cdots, h_t^l\}$ at the l -th for $l \in [1, L]$ layer are figured as

$$H^l = Transformer_l(H^{l-1}) \quad (1)$$

Here we regard H^L as the contextualized input tokens representations and utilize them for downstream task performance. We also use BERT-based models and those models keeping the BERT component fixed. Comparing these models, we end up performing task-specific fine-tuning to the BERT model giving us a lot better performance and allowing us to exploit the power of BERT [26–28].

B. Fine tuning BERT

We also investigate the fine-tuning impact on the performances as final. Precisely, we implement BERT to figure the contextualized representations token-level and keeping the BERT component parameters fixed at the phase of training and compared them with models fine-tuning BERT. We notice that the overall representation tenacity of BERT is away from acceptable for the tasks as downstream and model fine-tuning for Keyword extraction from job descriptions is essential for us to exploit the BERT strengths for performance improvement [29].

C. Downstream Model

Following obtaining the BERT vector representations, we build a downstream model on top of the BERT EL as shown in Fig. 1 To find out the most relevant keywords we make use of topic modeling techniques [30]. Topic modeling is frequently utilized if a large text collection cannot be read and sorted reasonably through a person. As a corpus covered documents, model of a topic tries to discover the structure as latent semantic, or topics in the documents are presenting [31]. These capable further also be utilized to group similar documents. Our major aim here is to catch topics of high- level represent existing information summary in the documents.

We then use a class-based variant of TF-IDF (Term Frequency - Inverse Document Frequency) which helps in calculating the degree of similarity among multiple documents. The TF (Term Frequency) in TF-IDF signifies the occurrence of the specific word in documents. Words of a high value of TF are of significance in documents. Nevertheless, the DF (Document Frequency) involves the number of times in which a particular word is appearing in the document collection. It can briefly be said as a statistic being numerical which exhibits the relevance of the keywords to few particular documents [32]. It figures the word existence in documents as multiple, not in just one document [33]. Thus, we try to compare the importance of words between multiple documents.

We also show a visualization of the BERT Embeddings for our data, this allows us to find and evaluate relationships for a word in a document. We also demonstrate this for an example token in the data in Fig. 2. These have also been made public at this link [2]

[2]https://iali.page.link/jd

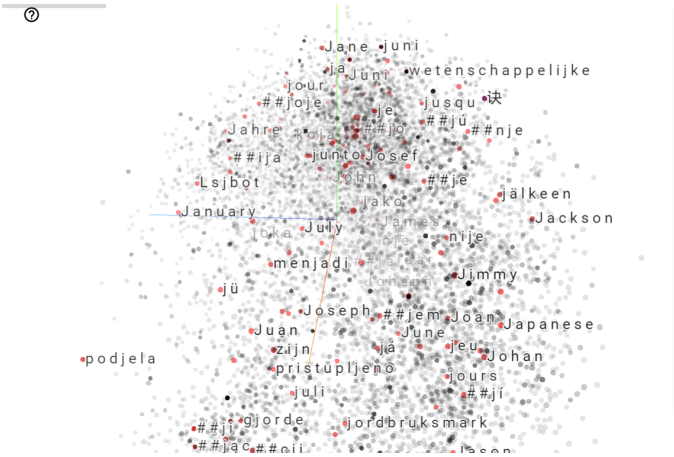

Fig. 2. BERT Embeddings for an example token in the data.

The most widely used topic modeling methods are Latent Dirichlet Allocation (LDA) [34] and Probabilistic Latent Semantic Analysis (PLSA) [35]. LDA is a generative probabilistic model which describes each document as a mixture of topics and each topic as a distribution of words that can extract latent topics from a collection of documents. We represent the documents as random mixtures over latent topics, characterizing each topic by a distribution over words [34, 36].

We make use of LDA to build the downstream model as it is generalises over PLSA by addition of distribution as Dirichlet prior over distributions of topic-word and document-topic. LDA suffers from "order effects"; for example, diverse topics are formed when the training data order is shuffled. Utilizing the default LDA settings can often cause systematic errors because of instability of topic modeling. To prevent doing so we use a a topic modeling combination (along with LDA) along with optimizer (DE or evolution as differential) which regulates the LDA parameters for similarity scores optimization using some techniques mentioned by Agrawal et al. [37].

D. The API

As mentioned earlier we aim to make this job descriptions keyword extractor to be hassle-free and very easy to implement by providing a simple to use REST API and gRPC servers which can allow this to be integrated into the existing system very easily by organizations [38]. To demonstrate the working of this we deploy our models using Flask [3] which is a lightweight WSGI (Web Server Gateway Interface) [39] web application framework.

In Fig. 3 we show the model API being tested on a real-life job description for demonstration purposes. The API developed by us for this model can also be easily tested. [4]

III. Data Set

In this paper we collect the data majorly through scraping the web for job descriptions which are available to be used

[3]https://palletsprojects.com/p/flask/

[4]https://iali.page.link/jd-demo

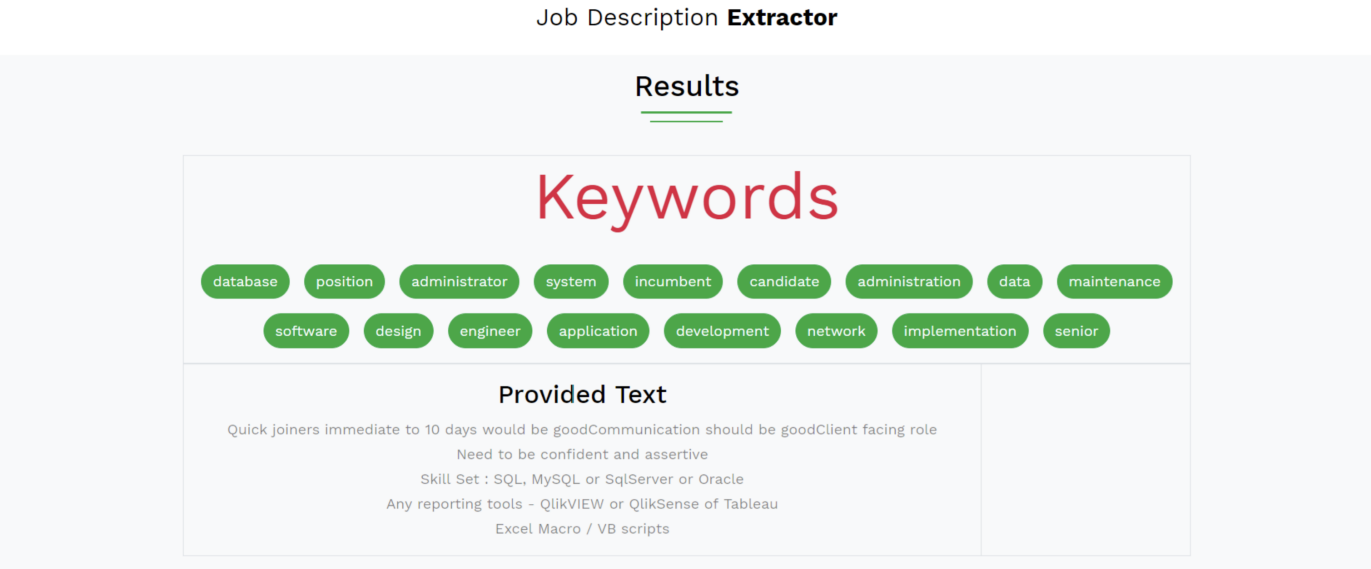

Fig. 3. Example of output from the model API.

freely and also use web content extraction for extracting less structured data [40], using combination of density sum shown by Sun et al. [41] and CSS features.

We then label this data using some crowd sourcing techniques [42], to make sure that users have truthfully submitted data we make use of the techniques mentioned by Zhao et al. [43] and also make sure to control the quality of data [44] we gather by crowd sourcing it. We further make sure to follow some additional tips and best practices that are crucial to the success of any project that uses crowd sourced data mentioned by Vaughan et al. [45].

We originally started developing and implementing the ideas on "Online Job Postings" data set [46] but soon found the data to not be enough and outdated so we also made use of scraping approaches to gather data apart of this.

IV. Experimental Results

As we mention earlier we use the pre-trained "bert-base-uncased" model by HuggingFace Transformers [47] where the number of transformer layers we use is $L = 32$ and the hidden size dim is 1024.

We compare our results with baseline TF-IDF results which as expected our model performs a lot better than baseline models. We also compare our model with other variants of BERT specifically: BERT-GRU, BERT-CRF and ALBERT [48] observing the F1 scores on the development set. We observe no significant difference in the performance of these models except using ALBERT which performs significantly lower. However, we also observe that using BERT-base uncased is quite robust to overfitting on comparing with other variants.

We also observe the impact of fine-tuning BERT for our use-case and impact of it on the final performance. We, use BERT to calculate the token-level representations while keeping the other parameters of the BERT component constant. We observe that the pre-trained BERT representation was far from the fine tuned results with almost 25 - 35 % improvement across all variants of BERT and approximately 27% improvement on our best performing BERT-base uncased model.

The general purpose representations are far from satisfactory and task-specific fine-tuning was essential to exploit the strengths of BERT and perform better on downstream tasks improving the overall performance.

V. Conclusion and Future Work

At the current work, we study the BERT effectiveness as a component being embedded on the Keyword extraction task from Job Descriptions. Precisely, we search coupling BERT as component being embedded to different other techniques to build an end-to-end keyword extractor. The obtained results exhibit BERT-based models' superiority in identifying aspect-based keywords along their robustness to over-fitting. We also build an API allowing this to be easily used or integrated into organizations and even be used by HRs and administrators.

The future work includes using the same model making minor changes to the downstream models according to the task use it for resume screening, automatic keyword adding for SEO or summarizing corporate text.

Conflict of Interest

The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

This research was supported with Cloud TPUs from Google's TensorFlow Research Cloud (TFRC) [5].

[5]https://www.tensorflow.org/tfrc

References

[1] Stuart Rose et al. "Automatic keyword extraction from individual documents". In: *Text mining: applications and theory* 1 (2010), pp. 1–20.

[2] Erion Çano and Ondřej Bojar. "Keyphrase Generation: A Text Summarization Struggle". In: *Proceedings of the 2019 Conference of the North* (2019). DOI: 10.18653/v1/n19-1070. URL: http://dx.doi.org/10.18653/v1/N19-1070.

[3] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. "Finding Advertising Keywords on Web Pages". In: *Proceedings of the 15th International Conference on World Wide Web*. WWW '06. Edinburgh, Scotland: Association for Computing Machinery, 2006, pp. 213–222. ISBN: 1595933239. DOI: 10.1145/1135777.1135813. URL: https://doi.org/10.1145/1135777.1135813.

[4] Stella Tomasi and Xiaolin Li. "Influences of Search Engine Optimization on Performance of SMEs: A Qualitative Perceptive". In: *Journal of Electronic Commerce in Organizations* 13 (2015), pp. 27–49. DOI: 10.4018/jeco.2015010103.

[5] Ron Berman and Zsolt Katona. "The Role of Search Engine Optimization in Search Marketing". In: *Marketing Science* 32.4 (2013), pp. 644–651. DOI: 10.1287/mksc.2013.0783. URL: https://doi.org/10.1287/mksc.2013.0783.

[6] Lin Liu et al. "An overview of topic modeling and its current applications in bioinformatics". In: *SpringerPlus* 5.1 (Sept. 2016), p. 1608. ISSN: 2193-1801. DOI: 10.1186/s40064-016-3252-8. URL: https://doi.org/10.1186/s40064-016-3252-8.

[7] Dhana Rao Venkat N. Gudivada and Jordan Paris. "Understanding Search-Engine Optimization". In: *Computer* 48.10 (2015), pp. 43–52. DOI: 10.1109/MC.2015.297.

[8] Peter D Turney. "Learning algorithms for keyphrase extraction". In: *Information retrieval* 2.4 (2000), pp. 303–336.

[9] Kai Hu et al. "A domain keyword analysis approach extending Term Frequency-Keyword Active Index with Google Word2Vec model". In: *Scientometrics* 114.3 (2018), pp. 1031–1068.

[10] Yujun Wen, Hui Yuan, and Pengzhou Zhang. "Research on keyword extraction based on word2vec weighted textrank". In: *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*. IEEE. 2016, pp. 2109–2113.

[11] Ning Jianfei and Liu Jiangzhen. "Using Word2vec with TextRank to extract keywords". In: *Data Analysis and Knowledge Discovery* 32.6 (2016), pp. 20–27.

[12] Kazi Saidul Hasan and Vincent 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 (Volume 1: Long Papers)*. 2014, pp. 1262–1273.

[13] Sifatullah Siddiqi and Aditi Sharan. "Keyword and keyphrase extraction techniques: a literature review". In: *International Journal of Computer Applications* 109.2 (2015).

[14] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[15] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[16] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[17] Matthew Tang et al. "Progress Notes Classification and Keyword Extraction using Attention-based Deep Learning Models with BERT". In: *CoRR* abs/1910.05786 (2019). arXiv: 1910.05786. URL: http://arxiv.org/abs/1910.05786.

[18] Parantu K. Shah et al. "Information extraction from full text scientific articles: Where are the keywords?" In: *BMC Bioinformatics* 4.1 (May 2003), p. 20. ISSN: 1471-2105. DOI: 10.1186/1471-2105-4-20. URL: https://doi.org/10.1186/1471-2105-4-20.

[19] Rafeeq Al-Hashemi. "Text Summarization Extraction System (TSES) Using Extracted Keywords." In: *Int. Arab. J. e Technol.* 1.4 (2010), pp. 164–168.

[20] Santosh Kumar Bharti and Korra Sathya Babu. *Automatic Keyword Extraction for Text Summarization: A Survey*. 2017. arXiv: 1704.03242 [cs.CL].

[21] Piotr Bojanowski et al. "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. ISSN: 2307-387X.

[22] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[23] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[24] Yoon Kim et al. *Structured Attention Networks*. 2017. arXiv: 1702.00887 [cs.CL].

[25] Xin Li et al. *Exploiting BERT for End-to-End Aspect-based Sentiment Analysis*. 2019. arXiv: 1910.00883 [cs.CL].

[26] Henry Tsai et al. *Small and Practical BERT Models for Sequence Labeling*. 2019. arXiv: 1909.00100 [cs.CL].

[27] Chi Sun et al. "How to Fine-Tune BERT for Text Classification?" In: *Chinese Computational Linguistics*. Ed. by Maosong Sun et al. Cham: Springer International Publishing, 2019, pp. 194–206. ISBN: 978-3-030-32381-3.

[28] Yang Liu. *Fine-tune BERT for Extractive Summarization*. 2019. arXiv: 1903.10318 [cs.CL].

[29] Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. *To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks*. 2019. arXiv: 1903.05987 [cs.CL].

[30] J. W. Uys, N. D. du Preez, and E. W. Uys. "Leveraging unstructured information using topic modelling". In: *PICMET '08 - 2008 Portland International Conference on Management of Engineering Technology*. 2008, pp. 955–961. DOI: 10.1109/PICMET.2008.4599703.

[31] Dimo Angelov. *Top2Vec: Distributed Representations of Topics*. 2020. arXiv: 2008.09470 [cs.CL].

[32] Shahzad Qaiser and Ramsha Ali. "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents". In: *International Journal of Computer Applications* 181.1 (July 2018), pp. 25–29. ISSN: 0975-8887. DOI: 10.5120/ijca2018917395. URL: http://www.ijcaonline.org/archives/volume181/number1/29681-2018917395.

[33] Sang-Woon Kim and Joon-Min Gil. "Research paper classification systems based on TF-IDF and LDA schemes". In: *Human-centric Computing and Information Sciences* 9.1 (Aug. 2019), p. 30. ISSN: 2192-1962. DOI: 10.1186/s13673-019-0192-7. URL: https://doi.org/10.1186/s13673-019-0192-7.

[34] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[35] Thomas Hoffman. "Probabilistic latent semantic indexing". In: *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval, 1999*. 1999, pp. 50–57.

[36] Yichen Jiang et al. "Recommending Academic Papers via Users' Reading Purposes". In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. RecSys '12. Dublin, Ireland: Association for Computing Machinery, 2012, pp. 241–244. ISBN: 9781450312707. DOI: 10.1145/2365952.2366004. URL: https://doi.org/10.1145/2365952.2366004.

[37] Amritanshu Agrawal, Wei Fu, and Tim Menzies. "What is wrong with topic modeling? And how to fix it using search-based software engineering". In: *Information and Software Technology* 98 (July 2018), pp. 74–88. ISSN: 0950-5849. DOI: 10.1016/j.infsof.2018.02.005. URL: http://dx.doi.org/10.1016/j.infsof.2018.02.005.

[38] Florian Tramèr et al. "Stealing Machine Learning Models via Prediction APIs". In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 601–618. ISBN: 978-1-931971-32-4. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/tramer.

[39] James Gardner. "The web server gateway interface (wsgi)". In: *The Definitive Guide to Pylons* (2009), pp. 369–388.

[40] Z. Zhou and Muntasir Mashuq. "Web Content Extraction Through Machine Learning". In: 2013.

[41] Fei Sun, Dandan Song, and Lejian Liao. "DOM based content extraction via text density". In: Jan. 2011, pp. 245–254. DOI: 10.1145/2009916.2009952.

[42] Ece Kamar, Severin Hacker, and Eric Horvitz. "Combining human and machine intelligence in large-scale crowdsourcing." In: *AAMAS*. Vol. 12. 2012, pp. 467–474.

[43] D. Zhao, X. Li, and H. Ma. "How to crowdsource tasks truthfully without sacrificing utility: Online incentive mechanisms with budget constraint". In: *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. 2014, pp. 1213–1221. DOI: 10.1109/INFOCOM.2014.6848053.

[44] Matthew Lease. "On Quality Control and Machine Learning in Crowdsourcing." In: Jan. 2011.

[45] Jennifer Wortman Vaughan. "Making Better Use of the Crowd: How Crowdsourcing Can Advance Machine Learning Research". In: *J. Mach. Learn. Res.* 18.1 (Jan. 2017), pp. 7026–7071. ISSN: 1532-4435.

[46] Habet Madoyan. *Online Job Postings, Version 1*. Apr. 2017. URL: https://www.kaggle.com/madhab/jobposts/version/1.

[47] Thomas Wolf et al. "HuggingFace's Transformers: State-of-the-art natural language processing". In: *arXiv preprint arXiv:1910.03771* (2019).

[48] Zhenzhong Lan et al. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. 2020. arXiv: 1909.11942 [cs.CL].