

# NLP-based Screening for IT Job Vacancies: A Case Study

Natalia Vanetik<sup>1,\*</sup>, Alona Kolesnev<sup>1</sup> and Genady Kogan<sup>1</sup>

<sup>1</sup>Department of Software Engineering, Shamoon College of Engineering (SCE), Beer-Sheva, Israel

## Abstract

The process of comparing a candidate's curriculum vitae (CV) or resume against a job description or a list of employment requirements is known as resume matching. The goal of this procedure is to determine how closely a candidate's abilities, credentials, experience, and other pertinent qualities meet the requirements of the position. There are employment courses that teach the applicants how to find the main requirements in a job description, and to describe their experience with an emphasis on these parts. On the other hand, Human Relations specialists are taught to extract the main information from hundreds of submitted resumes to find the best fit for their organization.

Typically, an automated system compares the text of the resume and vacancies and then gives a score or ranking indicating how closely the two match. However, this process may be time-consuming with a large number of applicants and lengthy vacancy descriptions.

In this paper, we introduce a novel dataset of software developer resumes extracted from Telegram. Additionally, we derived ground truth rankings for a subset of the resumes in this dataset with the assistance of expertly-guided human annotators. Furthermore, we propose an approach to simplify and enhance existing practices in resume vacancy matching, intending to streamline the recruitment process for both employers and applicants. We evaluate our approach and demonstrate its superiority over the leading resume-matching algorithm, OKAPI BM25 [1].

## Keywords

resume matching, extractive summarization, semantic vectors

## 1. Introduction

Employment is an indicator controlled at the government level, which affects the country's economic growth rate. The job search and hiring process is a part of the employment. Public and private companies provide their services to facilitate each stage for both the applicant and the employer. However, despite the involvement of many specialists and prepared manuals, it takes a lot of time. COVID has complicated and lengthened the resume-matching process due to the influx of job applications, increased competition, and the need for additional considerations such as remote work capabilities and health safety measures

---

In: M. Litvak, I.Rabaev, R. Campos, A. Jorge, A. Jatowt (eds.): *Proceedings of the IACT'23 Workshop, Taipei, Taiwan, 27-July-2023*

\*Corresponding author.


†These authors contributed equally.

✉ natalyav@sce.ac.il (N. Vanetik); alonako1@ac.sce.ac.il (A. Kolesnev); genadko@ac.sce.ac.il (G. Kogan)

🆔 0000-0002-4939-1415 (N. Vanetik); 0009-0003-8032-9403 (A. Kolesnev); 0009-0004-2974-1547 (G. Kogan)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

But how to know if a resume fits the job? Employment courses teach how to highlight the main thing in a job description and describe an experience in this regard. Such examination is subjective and may be unreliable. On the job seeker's side filters are available from the selection of vacancies, such as work experience, location, and desired salary (the salary is often not indicated on the job search sites.) It is possible to specify synonyms, as well as words that should not occur. Sorting is available by date or by the highest match to a basic or advanced query. Even after these adjustments, a lot of extra effort may be required to find a suitable vacancy. Therefore, using automatic tools for a vacancy or resume ranking can be beneficial for both sides. However, several ethical concerns arise including bias and discrimination, lack of transparency, limited context, exclusion of certain candidates, and the lack of human connection and empathy [2].

In this paper, we present a new dataset of tech-related resumes extracted from Telegram [3]. The data is anonymized and contains software developer resumes. With the aid of two human annotators who followed expert recommendations, we also determined the ranking of five developer job vacancies for 20 resumes in this dataset. We also propose a method to simplify and improve existing practices in the area of resume vacancy matching by representing both resumes and vacancies as semantic vectors and using their similarity to rank the documents and perform a case study by evaluating our method on the annotated part of our dataset. We address the following research questions to study how resume-vacancy matching can be made better, faster, and simpler:

**RQ1:** does summarization of resumes and vacancies improve ranking results?

**RQ2:** does text preprocessing improve ranking results?

**RQ3:** do semantic vectors produce better rankings than OKAPI BM25 algorithm?

The rest of the paper is structured as follows. Section 2 addresses related work, while Section 3 describes vacancy and resume data structure, collection, and annotation details. Section 4 describes text processing, construction of semantic vectors, competing methods, and the metrics we use to compare resume-vacancy rankings. In Section 5 we describe the evaluation setup and evaluation results. Finally, Section 6 describes conclusions and future work.

## 2. Related work

The task addressed in this paper belongs to the ranking and best-matching problem groups. Pudasaini et al. [4] strive to assist employers and applicants in making the right choice. They use word embeddings [5] and bag-of-words (BOW) model as text representations and compute job description and resume similarity as cosine similarity [6] and Gale-Shapley algorithm[7] that solves a stable matching problem. Tejaswini et al. [8] used the BOW model with TF-IDF weights [9] to represent the text, and the K-Nearest Neighbors (KNN) algorithm was applied to identify the resumes which are closely matching with the Job description. Kawan and Mohanty [10] have presented an approach for resume categorization using 600 resumes which were divided into three subfields – Work Experience, Academia, and Skills Certification. They used a support vector machine (SVM) as a model and TF-IDF vectors to represent the text. Joseph [11] focused on the problem of learning an end-user-specific ranking for CVs. They transformed all the data to text format by Textract [12] and deleted stop words using NLTK

Python package [13]. KNN algorithm was used to rank the CVs, and the dataset includes about 10,000 resumes enriched by the candidate's social profiles like LinkedIn and Github. Stefanovič et al. [14] use word-level n-grams-based [15] approach to finding similarity between texts by observing four similarity measures: cosine similarity, Dice similarity [16], extended Jaccard's similarity and overlap.

OKAPI BM25 is a similarity/scoring/ranking model which defines how matching documents are scored. It is one of the most accurate computational methods using a bag-of-words paradigm and was proven a useful baseline for resume ranking experiments [17]. The OKAPI BM25 algorithm takes into account the TF-IDF weight of every word in a query and calculates a score for each document in the collection based on the relevance of the document to the query. The document with the highest score is ranked at the top of the search results.

### 3. The data

#### 3.1. Vacancies

We downloaded job postings titled 'software developer' in the United States during August 2019 [18]. This data was extracted using JobsPikr - a job data delivery platform that extracts job data from job boards across the globe [19]. JobsPikr crawls over 70,000 employer websites daily to process over one million job postings; its API provides access to a database of over 500 million jobs over the past 3 years [20]. We denote the dataset **EDES** for SoftwarE DEveloper vacancies DataSet. The data was downloaded in .csv format, and it contains 10,000 vacancies with a combined size of 41.51 MB. The following data issues were found: (1) the same vacancy can appear more than once with a different unique id, and (2) portions of text in the job description field are repeated in many cases. We extracted the names of vacancies using the 'find-job-titles' library [21]. From the entire array of vacancies, we select those which include the word 'developer' or 'full-stack'.

#### 3.2. Resumes

As we did not succeed to find a good-quality resume dataset, we have constructed our own by downloading real resumes from the Telegram [3] group "HighTech Israel Jobs" [22] that freely publishes resumes and vacancies. We denote this dataset **TERT** for TELEgram Resume dataset.

All resumes in the group are written in a free format. Most often they consist of one page, but some have two or more pages. There is no clear structure, marks, or mandatory sections. The original resumes are in Russian and English; it also happens that one person uploads several versions of his resume. In addition, there are vacancies in the original dataset for both developers and testers, designers, project managers, etc. In connection with the above, the following data pre-processing steps were made.

1. Exclusion of duplicate records from the dataset with the help of the 'hashlib' library [23].
2. Exclusion resumes that are not in English with the help of the 'langdetect' library [24] - we keep the English-language files only.
3. Exclusion of resumes of non-developers.

More than 90% of the resumes were eliminated at the third stage because the dataset

**Table 1**  
The TERT dataset

format	Before pre-processing		After pre-processing	
	amount	size	amount	size
doc	3	175KB	0	0MB
docx	32	1,58MB	13	0.99MB
pdf	184	38,4MB	53	7,91MB

contained a lot of resumes of testers, project managers, etc. We used a custom-made list of keywords to search for in candidates' resumes using the 'flashtext' library [25].

#### 4. Anonymization.

Even though the resumes are published in an open Telegram group, we masked the names, phone numbers, and addresses of applicants.

Statistics for resume files in the TERT dataset before and after preprocessing appear in Table 1. Files in the .doc format were eliminated because they contained resumes of non-developers – Product and Marketing Managers, DevOps Engineers, QA Automation Engineers, and so on. The data is freely available on Google Drive at [https://drive.google.com/drive/folders/1VvrL5q6rczFZt\\_M713N3ejsQX85KJu65?usp=share\\_link](https://drive.google.com/drive/folders/1VvrL5q6rczFZt_M713N3ejsQX85KJu65?usp=share_link).

### 3.3. Human annotation

To determine the success of our methods, we need an evaluation system. In projects with similar tasks, authors used historical data about vacancies and selected candidates accordingly [11]. This is the easiest evaluation method; however, in our work, we cannot use it due to a lack of data. Another way to evaluate the system's efficiency is to compare the rankings produced by it to human rankings or rankings produced by another method that is known, widely used, and proven effective. We utilized this approach and selected a random subset of the TERT dataset containing 20 resumes and a subset of the EDES dataset containing five relevant vacancies (we took into account the time and effort load on our annotators). Our annotators have received detailed guidelines from Mrs. Yaarit Katzav, a manager of recruitment, organizational development, and welfare in an HR department of a large academic institution. Mrs. Katzav is a senior HR manager with over 13 years of experience who is responsible, among other things, for recruiting information systems engineers. The guidelines are available at [https://drive.google.com/drive/folders/1VvrL5q6rczFZt\\_M713N3ejsQX85KJu65?usp=share\\_link](https://drive.google.com/drive/folders/1VvrL5q6rczFZt_M713N3ejsQX85KJu65?usp=share_link).

## 4. Method

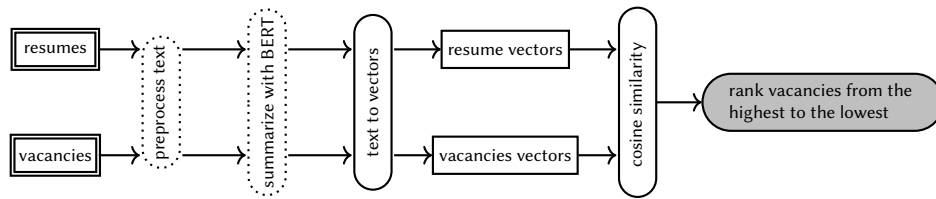
### 4.1. Text representations

We used the following text representations in our work: (1) tf\*idf vectors, where every resume and vacancy is treated as a separate document; (2) word and character n-grams with  $1 \leq n \leq 3$ ; (3) BERT sentence vectors obtained from a pre-trained English BERT model 'bert-base-uncased' [26]; (4) concatenation of all of the above (tf-idf, character and word n-grams, and BERT sentence vectors).

## 4.2. The pipeline

To match CVs and vacancies, we perform the following steps. First, we optionally preprocess the text by removing numbers and converting it to lowercase. Next, we either summarize the vacancies using BERT-based extractive summarization [27] with a pre-trained English model bert-base-uncased or leave them unchanged. Then, we represent the text of vacancies and CVs using the methods described below in Section 4.1. The maximum size of a summary is set to 10 sentences, and the ELBOW method of BERT summarizer is used to determine the optimal summary length.

Then we represent texts by numeric vectors obtained from the chosen text representation  $T$ . Therefore, we have a resume  $R$  represented by vector  $R_T$  and a vacancy  $V$  by vector  $V_T$ . Then we compute a match score of a resume and a vacancy as cosine similarity between those vectors. To obtain the final ranking, we sort cosine similarity scores in decreasing order. A pipeline of our approach is given in Figure 1.



**Figure 1:** Vacancy ranking pipeline

## 4.3. OKAPI BM25 and a BERT-based baseline

OKAPI BM25[28] is a similarity/scoring/ranking model which defines how matching documents are scored; BM stands for “best matching”. It is one of the most accurate computational methods using a bag-of-words paradigm and has proven a useful method for experiments and features for ranking [17, 29, 30]. A score of a document  $D$  is computed as

$$score(D, Q) = \sum_{i=1}^n IDF(q_i) \frac{f(q_i, D)(k_1 + 1)}{f(q_i, D) + k_1(1 - b + b \frac{|D|}{avgdl})} \quad (1)$$

where  $D$  is the document,  $Q$  is the query that consists of words  $q_1, \dots, q_n$   $f(q_i, D)$  is the number of times that a word  $q_i$  occurs in the document  $D$ . The average document length in the text collection is  $avgdl$ , and  $IDF$  is the IDF measure of a query word  $q_i$ . Parameters  $k_1$  and  $b$  are empirically-set/  $k_1$  is the term frequency saturation characteristic - the higher the value, the slower the saturation. In OKAPI BM25  $k_1 = 1.2$ . Parameter  $b$  (with range  $[0.0, 1.0]$ ) in the denominator is multiplied by the ratio of the document length. In OKAPI BM25  $b = 0.75$ . The coefficient values are default in the BM25Similarity method of Apache [31]. These values are also included in the range of recommended coefficient values [32].

We use the method introduced in [33] as a neural baseline which we denote by BERT-rank. This method improves the performance of neural ranking models by using cross-architecture knowledge distillation.

**Table 2**

Comparison of rankings produced by two human annotators

comparing	Krippendorff alpha	Cohen’s kappa	Spearman’s correlation
<i>A1 vs A2</i>	0.2191	0.1250	0.1309
<i>A1 vs averageA</i>	0.9557	0.9583	0.9513
<i>A2 vs averageA</i>	0.1847	0.1000	0.0873

#### 4.4. Metrics

To evaluate how the results of the developed methods correspond to the survey ranking results, it is necessary to calculate the correlation between automated and human rankings. Recall, F1, and similar metrics are not suitable for comparing rankings because they only measure the proportion of relevant items that are retrieved, without considering the order or position of those items in the ranking. Krippendorff’s alpha is a measure of inter-rater reliability or the degree of agreement among multiple raters ranging from 0 to 1 [34], computed with the help of ReCal web service [35]. We also use Spearman’s rank correlation coefficient (a statistical measure of the degree of association between two variables [36]) and Cohen’s kappa (a measure that evaluates how much two raters agree beyond what would be expected by chance [37]).

## 5. Evaluation

### 5.1. Setup

Experiments were performed on Google Colab [38] with Pro settings. BERT-based extractive summarizer [27] was used to produce resume summaries with ELBOW setting and maximum size of a summary set to 10 sentences. Sklearn and scipy SW packages were used to compute evaluation metrics and text representations.

### 5.2. Annotation and annotators’ agreement

Due to limited resources and the requirement to obtain at least two rankings for each resume, 20 resumes were randomly selected from the datasets. Every one of the 20 resumes was evaluated against five relevant vacancies selected for this purpose. The evaluators were asked to arrange the vacancies from the most relevant to the least relevant for a given resume.

To evaluate agreement among annotators (denoted by *A1* and *A2*), we have compared their rankings. We also compared the rankings of every annotator to their average rankings (denoted by *averageA*). Full results of the test are shown in Table 2. We can see that agreement is low for all of the reported parameters, which implies that the task of ranking is quite difficult even for humans.

### 5.3. Methods and metrics

As a topline, we run the OKAPI BM25 algorithm [1]. We also run BERT-rank [33] as a baseline. OKAPI BM25 is applied to original and preprocessed texts, before and after the summarization.

**Table 3**  
OKAPI BM25 and BERT-rank results

Algorithm	Text	Krippendorf alpha	Cohen's kappa	Spearman's correlation
OKAPI BM25	preprocessed text	0.4486	0.1625	0.3859
OKAPI BM25	original text	0.3924	0.0875	0.3274
OKAPI BM25	summary	0.1509	0.0625	0.0810
OKAPI BM25	preprocessed summary	-0.2902	-0.0750	-0.4354
BERT-rank	summary	-0.2203	-0.0375	-0.3627
BERT-rank	preprocessed summary	-0.2722	-0.1375	-0.4205

BERT-rank is applied to summarized texts only because of document size limitations of its implementation [33].

For our vector-based method, we use the following text forms – original and preprocessed full texts, and original and preprocessed summaries. We use text representations described in Section 4.1: tf-idf vectors, word, and character n-grams, BERT sentence embeddings, and their combinations.

#### 5.4. OKAPI BM25 results

Table 3 contains the results produced by the OKAPI BM25 algorithm on four data types, arranged by decreasing the value of Krippendorf's alpha. We can see that preprocessing of text improves its performance for all three metrics. Summarizing texts, however, decreases the performance, but again preprocessed summaries produce a better ranking than those produced from the original text. We can see the BERT-rank method produces a negative correlation in all three metrics, and therefore this method is not suitable for our domain.

#### 5.5. Semantic vectors ranking results

Table 4 shows the results of ranking with semantic vectors for all the text forms and representations discussed in Section 4.1. We can see that the top four methods in this table outperform OKAPI BM25 top method from Table 3. The top two methods use preprocessed text to produce semantic vectors, which complies with the results obtained for the OKAPI BM25 algorithm. We can also deduce that using summarization is not beneficial in this case because all semantic vectors produced from summaries, preprocessed or not, show a negative correlation with the ground truth results. BERT sentence embeddings alone do not produce good results. Character n-grams and combined vectors of n-grams, tf-idf, and BERT sentence embeddings are responsible for the top four results. Additionally, Tables 3-4 indicate that the three metrics we use to compare rankings strongly correlate one with another. Krippendorf's alpha consistently shows higher values than Cohen's kappa and Spearman's correlation, which be attributed to the fact that it takes into account the ordinal nature of the rankings and considers the agreement beyond chance [39]. In contrast, Cohen's kappa focuses on the pairwise agreement between two raters and may be less indicative when comparing rankings, and Spearman's correlation does not directly capture the level of agreement. Therefore, in this particular scenario, Krippendorf's alpha appears to be the preferable measure for comparing rankings.

**Table 4**  
Semantic vectors results

Text form	Text representation	Krippendorf alpha	Cohen's kappa	Spearman's correlation
preprocessed text	char ngrams	0.5285	0.0750	0.4703
preprocessed text	tfidf+char ngrams+word ngrams+BERT SE	0.4880	0.0625	0.4253
original text	tfidf+char ngrams+word ngrams+BERT SE	0.4686	0.0625	0.4183
original text	char ngrams	0.4686	0.0625	0.4183
preprocessed text	tfidf	0.1117	0.0000	0.0082
original text	tfidf	0.1117	0.0000	0.0082
summary	char ngrams	0.0899	-0.0500	-0.0055
preprocessed text	word ngrams	0.0497	-0.0250	-0.0627
original text	word ngrams	0.0497	-0.0250	-0.0627
preprocessed text	BERT SE	0.0491	-0.0250	-0.0574
summary	tfidf+char ngrams+word ngrams+BERT SE	0.0314	-0.0625	-0.0705
preprocessed summary	word ngrams	-0.1009	-0.0375	-0.2236
preprocessed summary	BERT SE	-0.1323	-0.1500	-0.2623
preprocessed summary	tfidf	-0.2494	-0.1625	-0.3886
preprocessed summary	tfidf+char ngrams+word ngrams+ BERT SE	-0.2539	-0.0875	-0.3936
summary	BERT SE	-0.2798	-0.1625	-0.4282
preprocessed summary	char ngrams	-0.2809	-0.1250	-0.4236
original text	BERT SE	-0.2839	-0.0875	-0.4274
summary	tfidf	-0.2989	-0.1250	-0.4436
summary	word ngrams	-0.2989	-0.1000	-0.4436

## 6. Conclusions and limitations

In this paper, we presented a TERT dataset of hi-tech resumes obtained from a Telegram group "HighTech Israel Jobs". We also evaluated this dataset with OKAPI BM25 and a neural baseline and suggested a new method for vacancy ranking that relies on the semantic vector representation of texts. We have observed that using summarization does not improve the results of automated vacancy ranking for all the methods, thus answering research question RQ1 negatively. However, a simple text preprocessing does give an improvement in all three metrics used to evaluate them, and we can answer positively to RQ2. We also find that using character n-grams vectors as semantic representation, with or without text preprocessing, produces a ranking superior to those of the OKAPI BM25, showing that the answer to RQ3 is positive as well.

Our method as suggested applies to IT vacancies and resumes only, and it has not been evaluated in other domains, specifically SW developers and full-stack SW engineers. In its current form, it applies to resumes and vacancies written in English only.

## Acknowledgments

We wish to express our gratitude to Mrs. Yaarit Katzav for taking the time to explain to us how to match vacancies to resumes. Her guidance and insights have been extremely helpful in our research.



## References

- [1] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford, et al., Okapi at trec-3, Nist Special Publication Sp 109 (1995) 109.
- [2] B. Dattner, T. Chamorro-Premuzic, R. Buchband, L. Schettler, The legal and ethical implications of using ai in hiring, *Harvard Business Review* 25 (2019).
- [3] Telegram FZ LLC and Telegram Messenger Inc., Telegram, 2019. URL: <https://telegram.org>.
- [4] S. Pudasaini, S. Shakya, S. Lamichhane, S. Adhikari, A. Tamang, S. Adhikari, Scoring of resume and job description using word2vec and matching them using gale-shapley algorithm, 2022. doi:10.1007/978-981-16-2126-0\_55.
- [5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, *Advances in neural information processing systems* 26 (2013).
- [6] F. Rahutomo, T. Kitasuka, M. Aritsugi, Semantic cosine similarity, in: *The 7th international student conference on advanced science and technology ICAST*, volume 4, 2012, p. 1.
- [7] L. E. Dubins, D. A. Freedman, Machiavelli and the gale-shapley algorithm, *The American Mathematical Monthly* 88 (1981) 485–494.
- [8] K. Tejaswini, V. Umadevi, S. Kadiwal, S. Revanna, Design and development of machine learning based resume ranking system, *Global Transitions Proceedings* (2021). doi:10.1016/j.gltp.2021.10.002.
- [9] C. Sammut, G. I. Webb, *Encyclopedia of machine learning*, Springer Science & Business Media, 2011.
- [10] S. Kawan, M. N. Mohanty, Multiclass resume categorization using data mining, *International Journal of Electrical Engineering and Technology* 11 (2020) 267–274.
- [11] A. Joseph, Resume analyser: Automated resume ranking software, *International Journal for Research in Applied Science and Engineering Technology* 8 (2020) 896–899. doi:10.22214/ijraset.2020.30378.
- [12] M. S. Neff, R. J. Byrd, B. K. Boguraev, The talent system: Texttract architecture and data model, *Natural Language Engineering* 10 (2004) 307–326.
- [13] E. Loper, S. Bird, Nltk: The natural language toolkit, *arXiv preprint cs/0205028* (2002).
- [14] P. Stefanovič, O. Kurasova, R. Štrimaitis, The n-grams based text similarity detection approach using self-organizing maps and similarity measures, *Applied Sciences* 9 (2019) 1870. doi:10.3390/app9091870.
- [15] P. F. Brown, V. J. Della Pietra, P. V. Desouza, J. C. Lai, R. L. Mercer, Class-based n-gram models of natural language, *Computational linguistics* 18 (1992) 467–480.
- [16] L. R. Dice, Measures of the amount of ecologic association between species, *Ecology* 26 (1945) 297–302.
- [17] J. Martinez-Gil, A. Paoletti, M. Pichler, A novel approach for learning how to automatically match job offers and candidate profiles, *Information Systems Frontiers* 22 (2020) 10 1007 10796–019–09929–7.
- [18] A. Jha, P. Kumar, 2017. URL: <https://data.world/jobspikr/software-developer-job-listings-from-usa>, jobsPikr datasets.
- [19] A. Jha, P. Kumar, JobsPikr, 2017. URL: <https://www.jobspikr.com/>.
- [20] J. F. API, 2017. URL: <https://www.jobspikr.com/job-feed-api/>, accessed 22 October 2022.

- [21] J. Ahlmann, find\_job\_titles python sw package, 2017. URL: <https://pypi.org/project/find-job-titles/>.
- [22] Telegram FZ LLC and Telegram Messenger Inc., Hightech israel jobs telegram channel, 2019-09-05. URL: <https://tgstat.ru/en/channel/@israjobs>.
- [23] docs.python.org, hashlib – secure hashes and message digests, 2022. URL: <https://docs.python.org/3/library/hashlib.html>, accessed 22 October 2022).
- [24] M. M. Danilak, langdetect pypi, 2022. URL: <https://pypi.org/project/langdetect/>, accessed 22 October 2022).
- [25] V. Singh, flashtext pypi, 2020. URL: <https://pypi.org/project/flashtext>, accessed 22 October 2022.
- [26] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT, Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
- [27] D. Miller, bert-extractive-summarizer pypi, 2019. URL: <https://pypi.org/project/bert-extractive-summarizer/>, accessed 22 October 2022).
- [28] G. Amati, Bm25, in: L. LIU, M. ÖZSU (Eds.), Encyclopedia of Database Systems, Springer, Boston, MA, 2009, pp. 59–61. doi:10.1007/978-0-387-39940-9\_921, in:.
- [29] S. Guo, F. Alamudun, T. Hammond, Résumatcher: A personalized résumé-job matching system, Expert Systems with Applications 60 (2016) 169–182.
- [30] P. Xu, D. Barbosa, Matching résumés to job descriptions with stacked models, in: Advances in Artificial Intelligence: 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, May 8–11, 2018, Proceedings 31, Springer, 2018, pp. 304–309.
- [31] T. A. L. Project, Bm25similarity - apache lucene 8.0.0 core api, [https://lucene.apache.org/core/8\\_0\\_0/core/org/apache/lucene/search/similarities/BM25Similarity.html](https://lucene.apache.org/core/8_0_0/core/org/apache/lucene/search/similarities/BM25Similarity.html), Year of Access. Accessed on Day Month Year.
- [32] Minerazzi, Minerazzi tutorials, <http://www.minerazzi.com/tutorials/index.php>, Year of Access. Accessed on Day Month Year.
- [33] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, A. Hanbury, Improving efficient neural ranking models with cross-architecture knowledge distillation, in: Proceedings of the 2021 Conference on Human Information Interaction and Retrieval, ACM, 2021, pp. 84–93.
- [34] K. Krippendorff, Computing krippendorff's alpha-reliability, 2011.
- [35] D. Freelon, Recal: Intercoder reliability calculation as a web service, International Journal of Internet Science 5 (2010) 20–33.
- [36] L. Myers, M. J. Sirois, Spearman correlation coefficients, differences between, Encyclopedia of statistical sciences 12 (2004).
- [37] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1960) 37–46.
- [38] D. Kim, K. Chai, J. Kim, H. Lee, J. Lee, J. Kim, Colaboratory: An educational research environment for machine learning using jupyter notebooks, Journal of Educational Resources in Computing (JERC) 16 (2017) 1–10.
- [39] A. de Raadt, M. J. Warrens, R. J. Bosker, H. A. Kiers, A comparison of reliability coefficients for ordinal rating scales, Journal of Classification (2021) 1–25.