big data and cognitive computing

MDPI

Article

NLP-Based Bi-Directional Recommendation System: Towards Recommending Jobs to Job Seekers and Resumes to Recruiters

Suleiman Ali Alsaif, Minyar Sassi Hidri *, Imen Ferjani, Hassan Ahmed Eleraky and Adel Hidri

Computer Department, Deanship of Preparatory Year and Supporting Studies, Imam Abdulrahman Bin Faisal University, Dammam 31441, Saudi Arabia
* Correspondence: mmsassi@iau.edu.sa

Abstract: For more than ten years, online job boards have provided their services to both job seekers and employers who want to hire potential candidates. The provided services are generally based on traditional information retrieval techniques, which may not be appropriate for both job seekers and employers. The reason is that the number of produced results for job seekers may be enormous. Therefore, they are required to spend time reading and reviewing their finding criteria. Reciprocally, recruitment is a crucial process for every organization. Identifying potential candidates and matching them with job offers requires a wide range of expertise and knowledge. This article proposes a reciprocal recommendation based on bi-directional correspondence as a way to support both recruiters' and job seekers' work. Recruiters can find the best-fit candidates for every job position in their job postings, and job seekers can find the best-match jobs to match their resumes. We show how machine learning can solve problems in natural language processing of text content and similarity scores depending on job offers in major Saudi cities scraped from Indeed. For bi-directional matching, a similarity calculation based on the integration of explicit and implicit job information from two sides (recruiters and job seekers) has been used. The proposed system is evaluated using a resume/job offer dataset. The performance of generated recommendations is evaluated using decision support measures. Obtained results confirm that the proposed system can not only solve the problem of bi-directional recommendation, but also improve the prediction accuracy.

Keywords: information system management; bi-directional recommender system; natural language processing; content-based filtering

Citation: Alsaif, S.A.; Sassi Hidri, M.; Ferjani, I.; Eleraky, H.A.; Hidri, A. NLP-Based Bi-Directional Recommendation System: Towards Recommending Jobs to Job Seekers and Resumes to Recruiters. *Big Data Cogn. Comput.* **2022**, *6*, 147. https://doi.org/10.3390/bdcc6040147

Academic Editor: Fabrizio Marozzo

Received: 13 October 2022
Accepted: 24 November 2022
Published: 1 December 2022

1. Introduction

Frictional unemployment is linked to informational imperfections, due to the costs of collecting, processing, and disseminating information, as well as to the asymmetry of information between suppliers and applicants, and the cognitive limitations of individuals. These imperfections are one of the reasons why some jobs remain unoccupied even though significant unemployment is observed in the same employment sectors.

Although the problems of job search and of finding potential candidates are identical from an abstract point of view, they nevertheless differ greatly in practice. Classically, a job seeker writes a resume that will be sent to the human resources (HR) departments of the hundreds of companies to which he or she applies. The quality of a resume plays a decisive role in recruitment, particularly in the context of high unemployment, where an HR director receives hundreds of resumes for each open position. The situation has evolved with the influx of companies, particularly on the Web (LinkedIn, Indeed, Monster, Qapa, MeteoJob, Keljob), offering employment access assistance services.

Several recommendations based on job descriptions and resumes using keywords or information retrieval techniques have been proposed (e.g., [1–6]). The problems encountered are linked, among other things, to the different nature of the languages used to

describe the users (job seekers) and the objects (job offers). One solution involves defining standard skills, ideally constituting a common language between offers and resumes.

However, this list of skills only partially answers the problem posed. First of all, it is a list that evolves more slowly than the professions and skills required in all sectors (from low-skilled social professions to high-tech professions). Secondly, the identification by users of terms corresponding to their real skills leads to some noise; symmetrically, recruiters may also only partially identify the relevant skills for the positions to be filled.

In addition, other factors come into play during a direct match between resumes and offers. Indeed, offers and resumes involve semi-structured information, containing, in addition to skills and experience, geographical notions, a salary range, and development prospects. The importance of this information depends on the context, the sector of employment and the position to be filled.

Following this analysis of the need and the evolution of uses, the implementation of systems of personalized reciprocal recommendation of job offers to job seekers and potential candidates to recruiters thus appear as one of the keys to combat frictional unemployment.

Our work is based on the idea that better access to information would improve the situation in terms of finding jobs for job seekers and potential candidates for recruiters, would reduce the cost in terms of time and opportunity, and would make it possible to remedy to a certain extent the phenomenon of frictional unemployment.

The main contributions of this article can be summarized as:

- Helping job seekers to find how their skills stack up against the most sought jobs and also recommend job postings that match the skills reflected in their resumes.
- Offering recruiters an engine that facilitates the identification and hiring of candidates.
- Improving the matching between candidates and job offers.
- Improving the computerized recruitment process.

Figure 1 shows the flowchart of the bi-directional recommender system. The inputs of our process are job listing and user profiles data. Pre-processing and training are performed to generate candidate profiles and jobs. Finally, matching is performed to generate a list of recommended jobs for job seekers and a list of recommended profiles for employers.

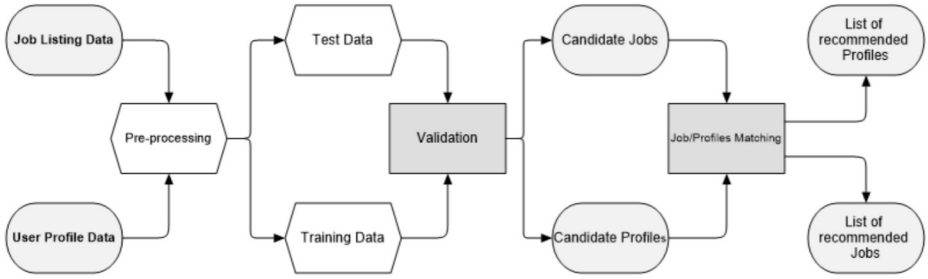

Figure 1. Bi-directional recommender system flowchart.

The remainder of this paper consists of seven sections. Related work is highlighted in Section 2. Natural Language Processing (NLP) and representations are presented in Section 3. The proposed bi-directional recommender system is detailed in Section 4. The methods used for building the recommender model are presented and subsequently described in Section 5. Discussion is presented in Section 6 followed by a brief summary of the paper's contributions and future works.

2. Related Work

Recommender systems are designed to reduce the amount of information that a user has to wade through to find the information or item that he is interested in; which could be frustrating [7,8].

The literature has proposed a variety of approaches to recommendation, but we are mainly focusing on the most recent recommendations.

The collaborative filtering (CF) system identifies neighbors who have similar expectations and offers items based on their ratings [9,10]. In online bookstores, for example, users will be offered book suggestions based on what their closest neighbors are interested in, what they have purchased, and what they have rated [11–13].

Recruiters can use this strategy if they want more candidates. Their preferences would be analyzed for a few candidates and more candidates would be offered to the recruiters. Other recruiters would retain these candidates if they share the recruiter's preferences.

An expectation-maximization (EM) algorithm was used by the authors in [13] as the basis for their bi-directional collaborative filtering method. Through iteration of this process, they combined user ratings with item ratings to build a general rating matrix.

In order to provide relevant elements to the user, content-based filtering (CBF) relies heavily on the information the user provides (his profile, for example) [14–18]. This is the approach we used in our study.

In these systems, a set of items and/or descriptions previously preferred by a user is analyzed, and a model is developed based on those items' characteristics to model the user's interests [19,20].

In traditional recommendation systems like CF and CBF models, item interactions are static and only capture the user's general preferences, but in the sequential recommendation system (SRS), the item interaction is a dynamic sequence [21–25]. To model user behavior sequences, the authors of [24] proposed a sequential recommendation model that employs deep bidirectional self-attention. In e-commerce, these systems can be used to predict the continuous change in preferences based on a person's history.

The conversational recommender system (CRS) allows users and the system to interact in real-time through natural language interactions, giving users [26–28] unprecedented opportunities to explicitly find out what users prefer. In a conversational recommendation, users can communicate more efficiently with the system by using natural language conversations to find or recommend the most relevant information [7,27,29,30].

Whatever recommendation approach (CF, CBF, SRS, or CRS) is applied, each type of system is based on quite different data structures and each has its behavior, so the choice of the approach to adopt is mainly linked to the use case study and the specifics of the data to be processed.

3. Natural Language Processing & Representations

This section focuses on the representation of texts in natural language except for many other applications of NLP.

3.1. Usual Pre-Processing

After data acquisition, the first phase of processing consists of cleaning the text. Below, we list the different usual pre-processing methods:

- Clean Tags: Remove special, unrecognized characters and HTML tags.
- Tokenization: Segmentation of data into meaningful units: terms, sometimes called *tokens*. Words are character strings separated by spaces and punctuation marks. A term is defined as a word or a series of consecutive words. More generally, n-gram designates a term formed of n consecutive words.
- Lemmatization: Normalization of terms in order to disregard variations in inflection (essentially related to conjugation, capital letters, and declension). We keep the masculine singular for an adjective and the infinitive for a conjugated verb. Lemmatization makes it possible to unify the different inflections of the same word (at the risk of unifying independent words, for example, the conjugated verb "played" and the noun "played"). Lemmatization also causes a loss of information; for example, words can have different meanings depending on whether they are singular or plural.
- Stop words: filtering of terms containing no semantic information (e.g., "of", "and", "or", "but", "we", "for"); the list of these tool words is determined manually for each language. The pruning of the most frequent terms (terms present in all the texts

and therefore without impact on their characterization) is also carried out; the only frequency is a hyper-parameter of the approach.

Pre-processing natural language documents often use methods such as: (i) morpho-syntactic tagging (part-of-speech tagging) associating each term with its grammatical category (noun, verb, determiner) in the statement or (ii) identification of the syntactic relations or dependencies of the terms in the sentence (subject, object, complement). These two methods are not considered in the rest of the paper, because the sentences of job offers and resumes are often short, with poor grammar (see Table 1).

Table 1. NLP and representations.

	Bag of Words	**Language Model**
Text pre-processing	Cleaning. Removed infrequent stop words	Cleaning. Removed infrequent stop words
Selection of terms	Word or n-gram [31]	Word or n-gram [31]
Vector representation	tf-idf [32]	Representation of terms
Projection in a latent space	LSA [33], LDA [34]	Word2Vec [35], RNN [36]
Similarity	Cosine [37], Jaccard [38]	Cosine [37], Jaccard [38]

3.2. Vector Representation—Bag of Words

The phase following the cleaning phase associates a vector representation with a sequence of terms. Two approaches are distinguished. The first considers bags of words (or bags of terms), where the order of the terms is not taken into account. The second approach keeps the sequential information in the sentences.

In the *bag of words* approach, a set of n_d documents, called a corpus, is represented by a $documents \times terms$ matrix denoted D, where a row corresponds to a document and a column to a term. The cell i, j of this matrix represents the number of occurrences of the term j in the document i (or a binary value of the presence of the term in the document).

3.3. Language Model

A language model is a probability distribution over a sequence of symbols. It is an alternative approach to bags of words, taking into account the order of terms. A language model is used to assess the plausibility of a sentence. It can also be used to generate a sequence of words or the next word in a sentence. The first language models are based on n-gram statistics [31,39,40].

However, these models do not generate vector representations for words or documents. The first continuous language model proposed in 2003 is based on neural networks [41]. This approach generates a vector representation for each word. Recent language models emphasize the representation of words [35]. These word representations are then evaluated on the target task, or used for continuous document representation. Note that some approaches directly define a continuous representation of documents.

A classic evaluation of the quality of a language model is based on the corpus "One Billion Words" [42]. In this case, the models must find the randomly removed words in the corpus. The n-gram statistical approach [31] remains a good reference measure, but is overtaken by neural approaches [41].

Word embedding is an encoding method that aims to represent the words or sentences of a text by vectors of real numbers, described in a vector model (or vector space model) [43]. It is then possible to make an inventory of all the words used in the text to create a dictionary. The text will therefore be represented in the form of a vector which will indicate the frequency at which each of the words appears. This step is decisive for the performance of NLP.

Word2Vec is a family of model architectures and optimizations that can be used to determine word embeddings from large datasets. Two very popular architectures belonging to this family are the Continuous Bag-of-Words (CBOW) and the skip-gram [44].

With the CBOW model, we start by choosing a word (the *i*th word) and then we select a certain number of neighbors (on the left and on the right), and we try to train the model of so that it would be able to predict word number "*i*" if given only its neighbors [45]. Indeed, these neighboring words represent the context, it means that to predict any word, its neighbors are taken into consideration, so the projection keeps the information and the context of the word (see Figure 2).

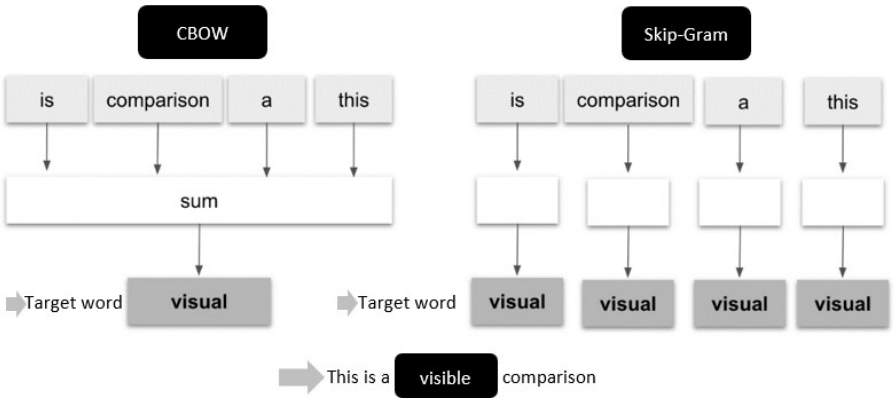

Figure 2. CBOW vs. skip-gram.

The skip-gram model is very similar, the model seeks to determine the associated context.

Skip-gram works well when the training data is not enough and it represents rare words or phrases very well. Conversely, CBOW is faster in training than skip-gram and is more accurate for frequent words [44].

4. Methodology

We aim at analyzing the explicit property of the content using the NLP technique. All the words in the job description were scraped from *sa.indeed.com* and jib seekers resumes.

4.1. Model Overview

Figure 3 shows the overview of the proposed bi-directional recommender system. Four steps distinguish the proposed system:

- Web scrapping: job offers are scrapped from *sa.indeed.com*.
- Pre-processing:
  - Converting dataturks annotated data to *spaCy* format to be used as training data.
  - Removing leading and trailing white spaces from entity spans.
- Training: Train *spaCy* named entity recognition (NER):
  - Creating blank language class.
  - Creating built-in pipeline components and adding them to the pipeline.
  - Testing the model.
  - Validating the prediction.
- Matching:
  - Creating job list.
  - Creating resume list.
  - Matching both list resumes and jobs.
  - Finding the most matching Job.
  - Finding most matching resumes.

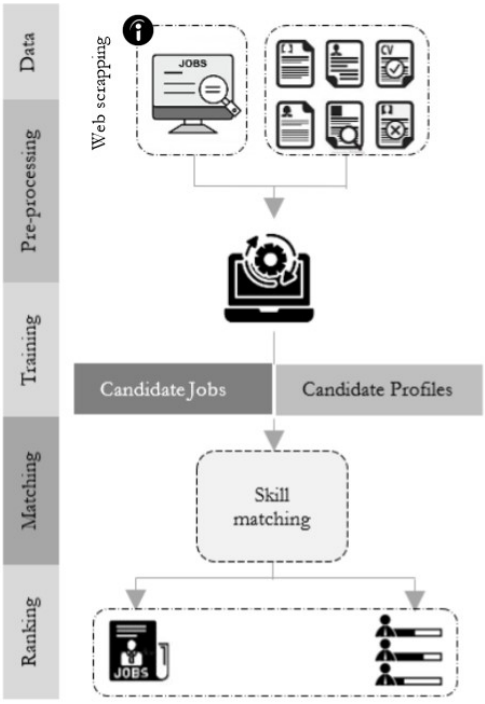

Figure 3. Overview of Bi-directional JRS.

4.2. Data Scrapping

Web scraper collects data from web sources and stores it locally in many formats so they can be used for further analysis [46].

Due to the fact that every website has its own structure, it is nearly impossible to generalize the method of web scraping for every website. To accomplish this, we use Python programming to automate or create web crawlers. Data is parsed using *BeautifulSoup*.

We scraped *PHP Developer*, *Data Scientist*, *Data Engineer*, *Java Developer*, and *Back-end Developer* job profiles from https://sa.indeed.com (accessed on 1 September 2022). We gathered jobs posted in the last 30 days (1 September 2022–1 October 2022), in 3 major Saudi Arabia regions (Jeddah, Riyadh, and Dammam). We used Selenium Webdriver to automate web scraping and save results in a local JSON file.

4.3. Pre-Processing

A major concern of any research project is to collect data that is beneficial to the research project at hand. Internet data can be found in different forms a variety of formats such as tables, comments, articles, job postings, together with different HTML tags that are embedded within them. In order to collect data, we use methods such as web scraping. Using web scraping, descriptive data is collected, as well as user data (from resumes).

Due to the fact that both data do not seem to have any history of previous interactions between them, the study continued to analyze the content explicitly in an attempt to find any obvious associations.

To categorize all the job listings into different categories, as well as compare user data and job listing data to identify similarities, the current study required a method to analyze text data.

The collected dataset has an attributes-filled string column with symbols and stop words. Especially in the column where skill details are present in both datasets (resumes and jobs). Data cleaning is our first process (see Figure 4).

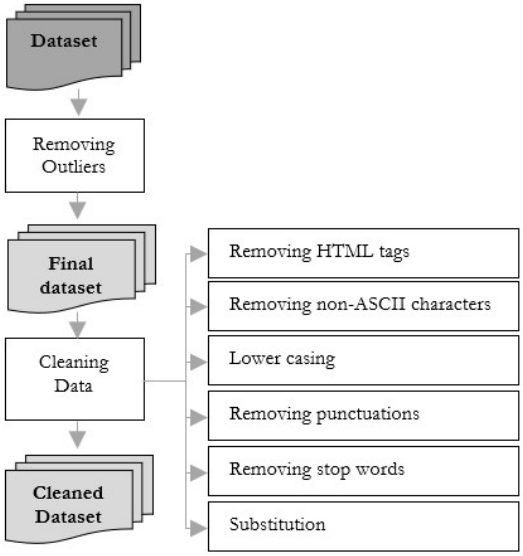

Figure 4. Overall cleaning process.

4.4. Training the Model to Auto-Detect Named Entities

The named-entity recognition (NER) process entails identifying the entities discussed in a text and categorizing them according to predefined categories in accordance with their significance to the text [47]. In our approach, categories could be entities like *Name*, *Location*, *Skills*, and *Education*. *spaCy* is used in extracting these named entities from the job description and resumes [47,48]. Figure 5 shows the NLP-based training diagram.

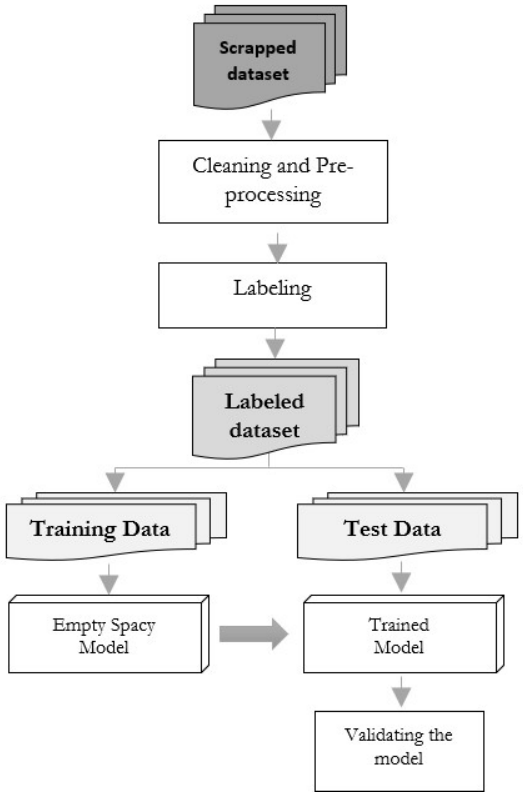

Figure 5. NLP-based training diagram.

Training is an iterative process in which the model's predictions are compared against the reference annotations to estimate the gradient of the loss [47]. The gradient of the loss is then used to calculate the gradient of the weights through back-propagation. The gradients indicate how the weight values should be changed so that the model's predictions become more similar to the reference labels over time.

4.5. Validating the Model

Predictive accuracy is measured based on the difference between the observed values and predicted values. Decision support metrics quantify recommendation success and error rates, considering that a recommender system generates a list of object recommendations for each user. The following metrics are used [49]:

- The *Precision* allows to know the number of positive predictions well made. In other words, it is the number of well-predicted positives (True Positives (*TP*)) divided by all the predicted positives (True Positives (*TP*) + False Positives (*FP*)). It is given by Equation (1).

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

- The *Recall* metric makes it possible to know the percentage of positives well predicted by our model. In other words, it is the number of well-predicted positives (True Positives (*TP*)) divided by all the positives (True Positives (*TP*) + False Negatives (*FN*)). It is given by Equation (2).

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

- The *F1-score* is a metric used to evaluate the performance of classification models with 2 or more classes. It summarizes the values of *Precision* and *Recall* in a single metric. Mathematically, this measure is defined as the harmonic mean of *Precision* and *Recall*. It is given by Equation (3).

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

4.6. Bi-Directional Matching between Resumes and Jobs

Figure 6 shows an overview of matching jobs and candidate profiles using the trained model.

Word2vec is used to retrieve terms that are based on a similarity between two terms. Then, using cosine similarity to measure the similarity within these two terms can be used to retrieve terms that are found using a similarity between these two terms [44,50]. It produces a vector space with dimensions in hundreds based on a large set of words, called the corpus. To measure the distance between words, we can generate a vector space model, and then we use the similarity measuring method.

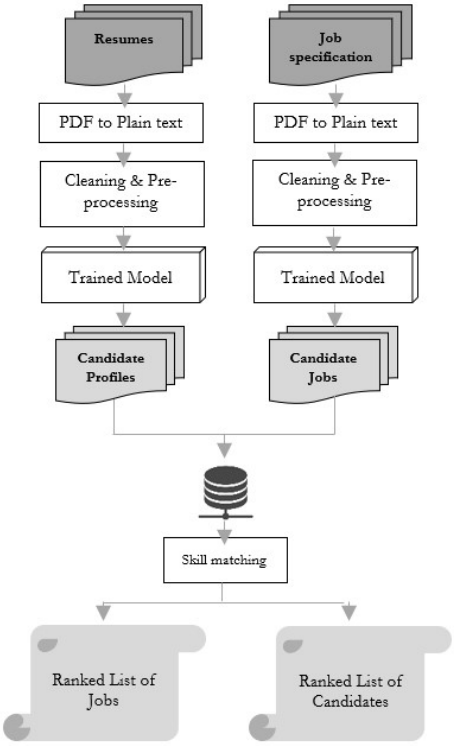

Figure 6. Matching jobs and candidate profiles using the trained model.

5. Results

5.1. Datasets Description

We collected data from two different sources for this study. We have two sets of data, one relating to user profiles and another relating to job profiles. We train the model with 138 resume data and test it on 25 resume data and 250 job descriptions.

The user profiles data we used to train our NLP model were acquired from https://github.com/DataTurks-Engg/Entity-Recognition-In-Resumes-SpaCy (accessed on 1 September 2022). The data, which consists of 138 resumes, was annotated using NER. A resume's skills are extracted from its content as entities. Figure 7 shows a snapshot of the dataset.


```
[
{
"label":["Name"],
"points":[{"start":3970, "end":3987,
"text": "Utkarsh Chaturvedi"}]
},
{
"label":["Skills"],
"points":[{"start":3721, "end":3721,
"text": "R"}]
},
{
"label":["Skills"],
"points":[{"start":3713, "end":3713,
"text": "Machine Learning"}]},
{
"label":["Skills"],
"points":[{"start":3665, "end":3665,
"text":" SQL"}
]
```


Figure 7. User profiles data description.

A total of 25 resumes were collected from https://www.hireitpeople.com/resume-database/totestourmodel (accessed on 1 September 2022).

As job listing data was scraped from published jobs in *sa.indeed.com*, the extracted data was then saved to JSON files. There are five files, each containing nearly 50 job descriptions for one IT job (from our selected jobs) published during a month. Job descriptions contain six key pieces of information: the job URL link, the company name, the location, the salary, and the full job description text.

Figure 8 shows the job dataset's structure.


```
{
"link":"https://sa.indeed.com/rc/clk?jk=e8013b5fc2044550&fccid=b7c2b45e2f6dc3fb&vjs=3",
"location": "Dammam",
"title": "Administrator",
"company": "Element Materials Technology",
"salary": NaN,
"desc": "Overview:\nElement has an opportunity for a Administrator to join our growing
team\nThis is a great opportunity to develop your Administration career within a Global TIC
business and in this role and in this role , you will manage administrative tasks and assist with
the Reception desk as required and other associated tasks\nThis position is based in Dammam ,
Second Industrial area in KSA\n\nResponsibilities:\nProvide administrative support to
laboratory operations and facility management while supporting the receptionist activities
when required\nAssist HR Manager and Operations Managers in preparing month end
reports\nCoordination of inter lab transactions \u2013 Sending LPO\u2019s and follow up for
invoices\nResponsible for arranging hotel accommodation and transportation for
guests\nManage interoffice queries and ensure that there is effective communication of
operational data to the management section\nSkills / Qualifications:\nIdeal candidate must be
a Bachelor degree holder with good communication skills in Arabic and English\nShould have
well developed skills in excel and word\nHave confident telephone manner and ready to follow
flexible timing\nThis is a KSA national role\n#LI-SR1
}
```


Figure 8. Job profiles data description.

The information from job specifications will be extracted and analyzed using NLP based on NER techniques to find skills.

5.2. Testing the Model and Validating the Prediction

A predictive model's performance can be easily evaluated by calculating the percentage of its predictions that are accurate [51]. A model's accuracy is a metric used to measure how many predictions it correctly predicted. Essentially, this value is calculated based on the ground truth classes and the predictions of the model. It is calculated by dividing the number of correct predictions by the total prediction number (Equation (4)).

$$Accuracy = \frac{Number\ \ of\ \ Correct\ \ Predictions}{Total\ \ Number\ \ of\ \ Predictions} \quad (4)$$

Since our predictive model deals with multi-class classification, the accuracy algorithm steps are as follows:

1. Get predictions from the model.
2. Calculate the number of correct predictions.
3. Divide it by the total prediction number.
4. Analyze the obtained value.

The higher the metric value, the better the prediction. The best possible value is 1 (if a model made all the predictions correctly), and the worst is 0 (if a model did not make a single correct prediction).

Table 2 presents the decision support metrics for *Name*, *Location*, *Skills*, and *Education* entities.

Table 2. Validating the prediction.

Entity	Accuracy	Precision	Recall	F1-Score
Name	99.885%	0.9988545291	0.9988532110	0.9987388618
Location	99.77%	0.9977064220	0.9977064220	0.9977064220
Skills	99.08%	0.9895468905	0.9908256880	0.9898169508
Education	100%	1.0	1.0	1.0

The obtained results show good values of accuracy for the Name (99.88%), Location (99.77%), Skills (99.08%), and Education (100%) entities. These values show how many times our ML model was correct overall.

Precision values are also very satisfying. This also proves how good our model is at predicting Name (∼1), Location (∼1), Skills (∼1), and Education (0.99) entities.

Recall values (∼1) obtained for Name, Location, Skills, and Education proves that the proposed model was able to detect these entities.

As mentioned previously, *PHP Developer*, *Data Scientist*, *Data Engineer*, *Java Developer*, and *Back-end Developer* job profile descriptions are scrapped from *sa.indeed.com* then trained to create a list of skills associated to each job. As presented in Table 3, the skills of the *Data Scientist* and *Data Engineer* job profiles are very similar and overlap, we will show that the proposed bi-directional recommendation system can distinguishes between job roles with close similarity even if their skills overlap.

Table 3. Creating list of Saudi job profiles from scrapped data (*sa.indeed.com*).

Job Profile	Skills
Data Scientist	['Image Processing', 'Computer Science', 'Speech Processing', 'C', 'Deep Learning', 'NLP', 'Machine Learning', 'R', 'Marine Studies', 'Python', 'Python', 'Machine Learning', 'SAS', 'Java', 'Scala', 'Hadoop', 'Hive', 'Big data', 'Programming', 'SQL server reporting', 'MSBI', 'SSRS', 'SQL Reporting', 'Artificial Intelligence', 'Pandas', 'Pyspark', 'Sklearn', 'Flask', 'Django', 'Map Reduce', 'Parametric Design', 'Modeling', 'Regression', 'Patterns', 'Data Mining', 'Text Mining', 'Oops', 'Deep Learning', 'Web Analytics', 'Time Series', 'Regression', 'Tensorflow', 'Azure', 'Linear Regression', 'Logistic Regression', 'Decision Tree', 'Random Forest', 'Data Structure', 'Computer Vision']
Back-end Developer	['Data Analytics', 'Javascript', 'Bash/Shell', 'PHP', 'Data Scientists', 'Python', ['MySQL', 'PostgreSQL', 'Microsoft Access', 'SQL Server', 'FileMaker', 'Oracle', 'RDBMS', 'dBASE', 'Clipper', 'FoxPro', 'Firebase', 'MongoDB']
Data Engineer	['C++', 'Discipline Manager', 'Data Services', 'Server Virtualization', 'Big Data', 'CAD', 'Python', 'Angular', 'Rider Levett', 'Hybrid Cloud', 'Microsoft Office', 'R', 'HTML', 'SQL', 'Product Management', 'Data Centre', 'AutoCAD', 'Java', 'Business Intelligence', 'Solution Development', 'C', 'Oracle Financials', 'CPA', 'Java', 'J2EE', 'Oracle Fusion', 'Oracle Cloud', 'Devops Android', 'Business Analyst', 'UI Developer', 'DBAs', 'Embedded Systems', '.NET', 'Hadoop', 'SQL Developer', 'Big Data', 'Tableau', 'Networking', 'Etl', 'Informatica', 'Ios', 'Quality Analyst', 'Project Manager', 'Python']
PHP Developer	['JAVA', 'Javascript', 'PHP', 'Angular', 'HTML', 'Laravel', 'HTML5', 'Python', 'Django', 'CSS', 'HTML/CSS', 'JavaScript', 'MongoDB', 'SQL', 'CodeIgniter', 'Symfony', 'Zend', 'Phalcon', 'CakePHP', 'Yii', 'FuelPHP', 'React', 'Ember']
Java Developer	['Java', 'Computer Science', 'Speech Processing', 'Deep Learning', 'C++', 'NLP', 'Machine Learning', 'Image Processing', 'Python', 'IHS', 'WAS', 'Java EE', 'SQL Server', '.NET core', 'C#', 'ASP.NET', 'RDLC', 'Linq', 'SQL', 'Web API', 'MVC', 'Javascript', 'Web Services', 'Oracle', 'MS SQL']

5.3. Finding Most Matching Jobs

The most matching jobs for 25 resumes are listed in Table 4.

Table 4. Finding most matching jobs.

Resume N°	Similarity (%)				
	Data Scientist	**Back-End Developer**	**Data Engineer**	**PHP Developer**	**Java Developer**
1	40	31.43	15.35	0	0
2	40	31.43	15.35	0	0
3	40	31.43	15.35	0	0
4	20	0	15.35	0	0
5	31.83	50.02	24.43	0	0
6	40	32.43	30.7	28.7	18.59
7	20	0	15.35	0	9.3
8	20	0	15.35	0	9.3
9	20	0	15.35	0	9.3
10	20	0	15.35	0	9.3
11	100	62.86	30.7	28.7	55.78
12	100	62.86	30.7	28.7	55.77
13	100	62.86	30.7	28.7	55.77
14	40	31.43	15.35	0	27.89
15	30.7	30.72	15.35	0	27.89
16	65.15	0	100	0	30.28
17	65.15	0	100	0	30.28
18	65.15	0	100	0	30.38
19	65.15	30.28	100	0	30.28
20	20	0	0	0	0
21	0	54.73	53.46	100	16.19
22	0	54.76	53.49	100	16.2
23	0	54.76	53.49	100	16.2
24	0	0	26.73	49.98	0
25	0	0	0	0	0

According to Table 4, the similarity of the resumes varies according to the job profiles. The value 0 obtained for the resumes 21, 22, 23, 24, and 25 for *Data Scientist* means that the job seekers having these resumes cannot succeed as *Data Scientist* and can strongly succeed as *PHP Developer*, moderately successful as *Data Engineer* and *Back-end Developer* profiles, and they will very low successful as *Java Developer*.

Job seekers with resumes, 2, and 3 cannot be successful as *Java Developer* (0%) and *PHP Developer* (0%). They can be successful as *Data Scientist* (40%) and *Back-end Developer* (31.43%). They will not be very successful as *Data Engineer* (15.35%).

From this table, we can conclude that the *Data Engineer* profile in Saudi Arabia is open to almost all job seekers who have *Data Scientist*, *Back-end Developer*, *Data Engineer*, *PHP Developer*, and *Java Developer* profiles with different similarities.

5.4. Finding Most Matching Resumes

The most matching resumes for *Data Engineer*, *Java Developer*, and *Back End Developer* profiles are listed in Table 5.

According to Table 5, Saudi recruiters who search for *PHP Developer* can strongly consider resumes 23 (100%), 22 (88.54%), and 21 (75.89%). If they seek Java developer profiles, they can consider resume 19.

Recruiters who seek *Data Scientist* profiles can consider resumes 6 (100%), 13 (95.03%), 12 (85.03%), 11 (82.85%), 20 (80.78%), and 3 (80.78%).

Recruiters who seek *Data Engineer* profiles can strongly consider resumes 23 (80%), 22 (77.78%), and 19 (76.19%).

If recruiters are looking for a profile that can be a *PHP Developer*, *Java developer*, *Back-end Developer*, and *Data Engineer*, they can consider 19 resumes (17.86%, 57.14%, 57.14%, and 76.19 resp.), 23 (100%, 14.4%, 40%, and 80% resp.), and 18 (12.5%, 40%, 40%, and 53.33% resp.).

Table 5. Finding the top-10 most matching resumes for PHP Developer, Java Developer, Data Scientist, Back-end Developer, and Data Engineer profiles.

PHP Developer		Java Developer		Data Scientist		Back-End Developer		Data Engineer	
R.N°	Similarity (%)	R.N°	Similarity (%)	R.N°	Similarity (%)	R.N°	Similarity (%)	R.N°	Similarity (%)
23	100	19	57.14	6	100	19	57.14	23	80
22	88.54	18	40	13	95.03	13	47.06	22	77.78
21	75.89	17	28.57	12	85.03	6	44.44	19	76.19
24	39.06	15	20.31	11	82.85	12	42.11	21	66.67
25	31.25	13	19.29	20	80.78	11	41.03	3	66.67
10	31.25	12	17.26	3	80.78	18	40	4	66.67
19	17.86	14	17.03	5	71.80	23	40	5	59.26
9	15.63	11	16.82	2	64.62	22	33.33	18	53.33
18	12.5	23	14.4	1	53.85	3	33.33	2	53.33
8	12.5	10	13.33	10	53.85	15	30.77	13	50.93

Given the overlap of the *Data Scientist* and *Data Engineer* profiles, recruiters will have the choice between a pure *Data Scientist* or a *Data Engineer* who can exercise the profession of a *Data Scientist* profile. For example, resumes 2, 3, 5, and 13 can help the company find these two types of profiles.

5.5. Recommender System Evaluation

To validate the prediction, we have used the confusion matrix to highlight not only the correct and incorrect predictions but, also, to give us an indication of the type of errors made. We find good recommendations on the diagonal of the confusion matrix (see Figure 9).

Figure 9. Confusion matrix.

The results show that:

- Five resumes have been classified as belonging to the *PHP Developer* job profile out of a total of five resumes, which is very satisfactory.
- For the *Data Scientist* job profile, five out of five were identified as belonging to this job profile, which is very satisfactory.
- For the *Data Engineer* job profile, two out of five resumes have been identified. This is due to the natural overlap between the skills of these two profiles.
- Four out of five resumes have been classified as belonging to the *Java Developer*, which is quite satisfactory.
- Four resumes have been classified as belonging to the *Back-end Developer* job profile out of a total of five, which is also quite satisfactory.

The number of TP is therefore 20 out of a total of 25 resumes, which is very satisfactory.

We also calculated metrics necessary for the analysis of this matrix such as precision, recall, F1-score, and accuracy.

According to Table 6, we can observe a precision of one for the PHP and Java Developer profiles. This proves that the proposed system succeeded in the recommendation since the skills required for these two profiles do not overlap. The precision value obtained for the *Data Engineer* profile is a bit low (0.5). This is due to the strong overlap of skills with the *Data Scientist* profile and a weak overlap with the *Back-end Developer* profile. The *Data Scientist* profile has an acceptable precision of 0.71 given the prediction obtained from the resumes belonging to the *Data Engineer* profile, but which were predicted in this profile.

Table 6. Validating the bi-directional recommendation using decision support metrics.

Job Class	Precision	Recall	F1-Score	Support
PHP Developer	1	1	1	5
Data Scientist	0.71	1	0.83	5
Data Engineer	0.5	0.4	0.44	5
Java Developer	1	0.8	0.89	5
Back-end Developer	0.8	0.8	0.8	5
Macro AVG	0.8	0.8	0.79	25
Weighted AVG	0.8	0.8	0.79	25

Because all *PHP Developer* and *Data Scientist* resumes were predicted as belonging to these same two profiles, the obtained recall values are equal to 1. Contrary, for *Data Engineer*, *Java Developer*, and *Back-end Developer*, two, four, and four resumes, respectively, were correctly assigned out of five, five, and five resumes and which gave us a recall of 0.4, 0.8, and 0.8, respectively.

The overall accuracy of 0.8 is obtained by our recommendation system. This value shows how the proposed recommender system was correct overall.

6. Discussion

The proposed NLP-based recommender system is based on the idea that better access to information would improve the situation in terms of finding jobs for job seekers and potential candidates for recruiters, would reduce the cost in terms of time and opportunity, and would make it possible to remedy a certain extent the phenomenon of frictional unemployment.

Because ML is at the core of many recent advances in science and technology, the proposed recommender system is evaluated using a resume/job offer dataset. The performance of generated recommendations is evaluated using ML such as accuracy, precision, recall, and F1-score.

The obtained results in matching resumes to Saudi jobs and Saudi job offers to profiles are very encouraging for job seekers as well as for Saudi recruiters. The proposed system is part of a framework for improving the computerized recruitment process. It can help job seekers to find how their skills stack up against the most sought jobs and also recommend job postings that match the skills reflected in their resumes. It also offers recruiters an engine that facilitates the identification and hiring of candidates.

This work will benefit from several improvements. In particular, we will have to study the dates of the relevant experience for the offer in order not to offer profiles with too old an experience. The detection of skills in an offer will benefit from the work presented in [52], who have developed a system that automatically builds a skills ontology for a certain number of jobs, distributed by universe. Geo-location could also extend to neighboring towns of those sought.

7. Conclusions and Future Works

Recruitment is one of the most important tasks in job seekers' and also organizations' point of view. If there could be a model that could recommend the possibility of a job

seeker being recruited or placed in an organization or not, it can save a lot of time for HR managers, staffs and placement committees.

The proposed bi-directional JRS would ease the burden of a recruiter by reducing the number of irrelevant applicants and a job seeker by offering recommended jobs. It focused on a resumes/job offers (resp. job offers/potential candidates) matching task while helping recruiters and job seekers to support their work through recommendations and help job seekers to find suitable job offers. Potential candidates are identified for recruiters and job offers are recommended for job seekers. Model prediction is evaluated decision support metrics such as accuracy, precision, recall, and F1-score.

Author Contributions: Conceptualization, S.A.A., M.S.H. and H.A.E.; methodology, S.A.A., I.F. and A.H.; formal analysis, S.A.A., I.F. and H.A.E.; investigation, A.H.; resources, S.A.A. and I.F; data curation, S.A.A. and I.F.; writing—original draft preparation, S.A.A. and H.A.E.; writing—review and editing, M.S.H. and A.H.; visualization, H.A.E.; supervision, M.S.H.; validation, M.S.H.; project administration, M.S.H.; funding acquisition, M.S.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research study was funded by Deanship of Scientific Research, Imam Abdulrahman Bin Faisal University [2022-017-PYSS].

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [https://github.com/imenFerjani/NLP_Job_Skills_match].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CBF	Content-Based Filtering
CBOW	Continuous Bag-Of-Words
CF	Collaborative Filtering
CRS	Conversational Recommender System
EM	Expectation-Maximization
HR	Human Resources
LSA	Latent Semantic Analysis
NER	Named Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
RNN	Recurrent Neural Network
RS	Recommender System
SRS	Sequential Recommendation Systems

References

1. Catherine, R.; Visweswariah, K.; Chenthamarakshan, V.; Kambhatla, N. PROSPECT: A system for screening candidates for recruitment. In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, Toronto, ON, Canada, 26–30 October 2010; pp. 659–668. [CrossRef]
2. Parida, B.; KumarPatra, P.; Mohanty, S. Prediction of recommendations for employment utilizing machine learning procedures and geo-area based recommender framework. *Sustain. Oper. Comput.* **2022**, *3*, 83–92. [CrossRef]
3. Kokkodis, M.; Ipeirotis, P.G. Demand-Aware Career Path Recommendations: A Reinforcement Learning Approach. *Manag. Sci.* **2021**, *67*, 4362–4383. [CrossRef]
4. Lacic, E.; Reiter-Haas, M.; Kowald, D.; Dareddy, M.R.; Cho, J.; Lex, E. Using autoencoders for session-based job recommendations. *User Model. User Adapt. Interact.* **2020**, *30*, 617–658. [CrossRef]
5. Saeed, T.; Sufian, M.; Ali, M.; Rehman, A.U. Convolutional Neural Network Based Career Recommender System for Pakistani Engineering Students. In Proceedings of the 2021 International Conference on Innovative Computing (ICIC), Lahore, Pakistan, 9–10 November 2021; pp. 1–10. [CrossRef]
6. Zhu, G.; Chen, Y.; Wang, S. Graph-Community-Enabled Personalized Course-Job Recommendations with Cross-Domain Data Integration. *Sustainability* **2022**, *14*, 7439. [CrossRef]
7. Jannach, D.; Manzoor, A.; Cai, W.; Chen, L. A Survey on Conversational Recommender Systems. *ACM Comput. Surv.* **2021**, *54*, 1–36. [CrossRef]

8. Syed, M.H.; Huy, T.Q.B.; Chung, S. Context-Aware Explainable Recommendation Based on Domain Knowledge Graph. *Big Data Cogn. Comput.* **2022**, *6*, 11. [CrossRef]
9. Wu, L. Collaborative Filtering Recommendation Algorithm for MOOC Resources Based on Deep Learning. *Complexity* **2021**, *2021*, 5555226. [CrossRef]
10. Liu, X.; Li, S. Collaborative Filtering Recommendation Algorithm Based on Similarity of Co-Rating Sequence. In Proceedings of the International Symposium on Electrical, Electronics and Information Engineering, Chiang Mai, Thailand, 25–27 February 2022; Association for Computing Machinery: New York, NY, USA, 2021; pp. 458–463. [CrossRef]
11. Sun, N.; Chen, T.; Guo, W.; Ran, L. Enhanced Collaborative Filtering for Personalized E-Government Recommendation. *Appl. Sci.* **2021**, *11*, 12119. [CrossRef]
12. Hu, B.; Long, Z. Collaborative Filtering Recommendation Algorithm Based on User Explicit Preference. In Proceedings of the IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 28–30 June 2021; pp. 1041–1043. [CrossRef]
13. Mao, Y.; Zhang, F.; Xu, L.; Zhang, D.; Yang, H. A Bidirectional Collaborative Filtering Recommender System Based on EM Algorithm. In Proceedings of the Advances in Smart Vehicular Technology, Transportation, Communication and Applications, Mount Emei, China, 25–28 October 2018; Pan, J.S., Wu, T.Y., Zhao, Y., Jain, L.C., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 265–273. [CrossRef]
14. Javed, U.; Shaukat, K.; Hameed, I.A.; Iqbal, F.; Alam, T.M.; Luo, S. A Review of Content-Based and Context-Based Recommendation Systems. *Int. J. Emerg. Technol. Learn. iJET* **2021**, *16*, 274–306. [CrossRef]
15. Li, L.; Wang, Z.; Li, C.; Chen, L.; Wang, Y. Collaborative filtering recommendation using fusing criteria against shilling attacks. *Connect. Sci.* **2022**, *34*, 1678–1696. [CrossRef]
16. Tai, Y.; Sun, Z.; Yao, Z. Content-Based Recommendation Using Machine Learning. In Proceedings of the IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP), Gold Coast, Australia, 20–23 September 2021; pp. 1–4. [CrossRef]
17. Gu, Y.; Zhao, B.; Hardtke, D.; Sun, Y. Learning Global Term Weights for Content-Based Recommender Systems. In Proceedings of the 25th International Conference on World Wide Web, Montreal, QC, Canada, 11–15 April 2016; pp. 391–400. [CrossRef]
18. Alsaif, S.A.; Sassi Hidri, M.; Eleraky, H.A.; Ferjani, I.; Amami, R. Learning-Based Matched Representation System for Job Recommendation. *Computers* **2022**, *11*, 161. [CrossRef]
19. Joseph, A.; Benjamin, M.J. Movie Recommendation System Using Content-Based Filtering and Cosine Similarity. In Proceedings of the National Conference on Emerging Computer Applications (NCECA), Kerala, India, 22 October 2022; pp. 405–408. [CrossRef]
20. Pérez-Almaguer, Y.; Yera, R.; Alzahrani, A.A.; Martínez, L. Content-based group recommender systems: A general taxonomy and further improvements. *Expert Syst. Appl.* **2021**, *184*, 115444. [CrossRef]
21. Ni, J.; Tang, G.; Shen, T.; Cai, Y.; Cao, W. An Improved Sequential Recommendation Algorithm based on Short-Sequence Enhancement and Temporal Self-Attention Mechanism. *Complexity* **2022**, *2022*, 4275868. [CrossRef]
22. Jiang, W.; Lin, F.; Zhang, J.; Yang, C.; Zhang, H.; Cui, Z. Dynamic Sequential Recommendation: Decoupling User Intent from Temporal Context. In Proceedings of the 2021 International Conference on Data Mining Workshops (ICDMW), Auckland, New Zealand, 7–10 December 2021; pp. 18–26. [CrossRef]
23. Latifi, S.; Jannach, D.; Ferraro, A. Sequential recommendation: A study on transformers, nearest neighbors and sampled metrics. *Inf. Sci.* **2022**, *609*, 660–678. [CrossRef]
24. Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; Jiang, P. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1441–1450. [CrossRef]
25. Wu, C.; Wu, F.; Qi, T.; Li, C.; Huang, Y. Is News Recommendation a Sequential Recommendation Task? In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'22, Madrid, Spain, 11–15 July 2022; Association for Computing Machinery: New York, NY, USA, 2022; pp. 2382–2386. [CrossRef]
26. Mentec, F.; Miklós, Z.; Hervieu, S.; Roger, T. Conversational recommendations for job recruiters. In Proceedings of the Knowledge-aware and Conversational Recommender Systems, Amsterdam, The Netherlands, 27 September–1 October 2021.
27. Manzoor, A.; Jannach, D. Towards retrieval-based conversational recommendation. *Inf. Syst.* **2022**, *109*, 102083. [CrossRef]
28. Manzoor, A.; Jannach, D. Revisiting Retrieval-based Approaches for Conversational Recommender Systems. In Proceedings of the 12th Italian Information Retrieval Workshop 2022, Milan, Italy, 29–30 June 2022.
29. Manzoor, A.; Jannach, D. Generation-based vs. Retrieval-based Conversational Recommendation: A User-Centric Comparison. In Proceedings of the RecSys'21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September–1 October 2021; Pampín, H.J.C., Larson, M.A., Willemsen, M.C., Konstan, J.A., McAuley, J.J., Garcia-Gathright, J., Huurnink, B., Oldridge, E., Eds.; ACM: New York, NY, USA, 2021; pp. 515–520. [CrossRef]
30. Wu, Y.; Macdonald, C.; Ounis, I. Multimodal Conversational Fashion Recommendation with Positive and Negative Natural-Language Feedback. In Proceedings of the 4th Conference on Conversational User Interfaces, Glasgow, UK, 26–28 July 2022; pp. 1–10. [CrossRef]
31. Kneser, R.; Ney, H. Improved backing-off for M-gram language modeling. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Detroit, Michigan, 9–12 May 1995; pp. 181–184. [CrossRef]

32. Sammut, C.; Webb, G.I. (Eds.) TF-IDF. In *Encyclopedia of Machine Learning*; Springer: Boston, MA, USA, 2010; pp. 986–987. [CrossRef]
33. Deerwester, S.; Dumais, S.; Furnas, G.; Landauer, T.; Harshman, R. Indexing by latent semantic analysis. *J. Am. Soc. Inf. Sci.* **1990**, *41*, 391–407. [CrossRef]
34. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022. . jmlr.2003.3.4-5.993. [CrossRef]
35. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
36. Kim, Y.; Jernite, Y.; Sontag, D.; Rush, A.M. Character-Aware Neural Language Models. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 2741–2749. [CrossRef]
37. Li, B.; Han, L. Distance Weighted Cosine Similarity Measure for Text Classification. In Proceedings of the 14th International Conference on Intelligent Data Engineering and Automated Learning, Hefei, China, 20–23 October 2013; pp. 611–618. [CrossRef]
38. Sternitzke, C.; Bergmann, I. Similarity measures for document mapping: A comparative study on the level of an individual scientist. *Scientometrics* **2007**, *78*, 113–130. [CrossRef]
39. Good, I.J. The Population Frequencies of species and the estimation of population parameters. *Biometrika* **1953**, *40*, 16–264. [CrossRef]
40. Witten, I.H.; Bell, T.C. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Trans. Inf. Theory* **1991**, *37*, 1085–1094. [CrossRef]
41. Bengio, Y.; Ducharme, R.; Vincent, P.; Janvin, C. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155. [CrossRef]
42. Chelba, C.; Mikolov, T.; Schuster, M.; Ge, Q.; Brants, T.; Koehn, P. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *arXiv* **2013**, arXiv:1312.3005.
43. Ammar, M.; Hidri, A.; Sassi Hidri, M. Time-sensitive clustering evolving textual data streams. *Int. J. Comput. Appl. Technol.* **2020**, *63*, 25–40. [CrossRef]
44. Kenter, T.; de Rijke, M. Short Text Similarity with Word Embeddings. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, Melbourne, Australia, 19–23 October 2015; ACM: New York, NY, USA, 2015; pp. 1411–1420. [CrossRef]
45. İrsoy, O.; Benton, A.; Stratos, K. Corrected CBOW Performs as well as Skip-gram. In Proceedings of the Second Workshop on Insights from Negative Results in NLP, Online, Punta Cana, Dominican Republic, 10 November 2021; pp. 1–8. [CrossRef]
46. Egger, R.; Kroner, M.; Stöckl, A. Web Scraping. In *Applied Data Science in Tourism: Interdisciplinary Approaches, Methodologies, and Applications*; Egger, R., Ed.; Springer International Publishing: Berlin/Heidelberg, Germany, 2022; pp. 67–82. [CrossRef]
47. Fantechi, A.; Gnesi, S.; Livi, S.; Semini, L. A spaCy-based tool for extracting variability from NL requirements. In Proceedings of the 25th ACM International Systems and Software Product Line Conference, New York, NY, USA, 6–11 September 2021; pp. 32–35. [CrossRef]
48. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537. [CrossRef]
49. Ferjani, I.; Sassi Hidri, M.; Frihida, A. SiNoptiC: Swarm intelligence optimisation of convolutional neural network architectures for text classification. *Int. J. Comput. Appl. Technol.* **2022**, *68*, 82–100. [CrossRef]
50. Barzilay, R.; Elhadad, N. Sentence Alignment for Monolingual Comparable Corpora. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Sapporo, Japan, 11–12 July 2003; pp. 25–32. [CrossRef]
51. Ferjani, I.; Sassi Hidri, M.; Frihida, A. Multi-GPU-based Convolutional Neural Networks Training for Text Classification. In Proceedings of the Intelligent Systems Conference, IntelliSys, Amsterdam, The Netherlands, 2–3 September 2021; pp. 72–84. [CrossRef]
52. Boudjedar, S.; Bouhenniche, S.; Mokeddem, H.; Benachour, H. Automatic Human Resources Ontology Generation from the Data of an E-Recruitment Platform. *Metadata Semant. Res.* **2020**, *1355*, 97–109. [CrossRef]