mathematics

MDPI

Article

Data Mining of Job Requirements in Online Job Advertisements Using Machine Learning and SDCA Logistic Regression

Bogdan Walek * and Ondrej Pektor

Department of Informatics and Computers, University of Ostrava, 30 dubna 22, 70103 Ostrava, Czech Republic; ondrej.pektor@osu.cz

* Correspondence: bogdan.walek@osu.cz

Abstract: There are currently many job portals offering job positions in the form of job advertisements. In this article, we are proposing an approach to mine data from job advertisements on job portals. Mainly, it would concern job requirements mining from individual job advertisements. Our proposed system consists of a data mining module, a machine learning module, and a postprocessing module. The machine learning module is based on the SDCA logistic regression. The postprocessing module includes several approaches to increase the success rate of the job requirements identification. The proposed system was verified on 20 most searched IT job positions from the selected job portal. In total, 9971 job advertisements were analyzed. Our system's verification is finding all job requirements in 80% of analyzed advertisements. The detected job requirements were also compared with the Open Skills database. Based on this database and the extension of IT job positions with other typical job skills, we created a list of the most frequent job skills in selected IT job positions. The main contribution is the development of a universal system to detect job requirements in job advertisements. The proposed approach can be used not only for IT positions, but also for various job positions. The presented data mining module can also be used for various job portals.

Keywords: data mining; data mining approach; machine learning; SDCA logistic regression; job requirements; job advertisements

Citation: Walek, B.; Pektor, O. Data Mining of Job Requirements in Online Job Advertisements Using Machine Learning and SDCA Logistic Regression. *Mathematics* **2021**, *9*, 2475. https://doi.org/10.3390/math9192475

Academic Editor: Anatoliy Swishchuk

Received: 31 August 2021
Accepted: 25 September 2021
Published: 3 October 2021

1. Introduction

There are currently many job portals offering job positions in the form of job advertisements. Information in such advertisements and requirements for job applicants are usually structured in a certain way. However, some job advertisements contain unstructured information, which hampers orientation and automated processing of their content. Job advertisements consist of a job position name and advertisement description. The advertisement description usually contains some of the following information.

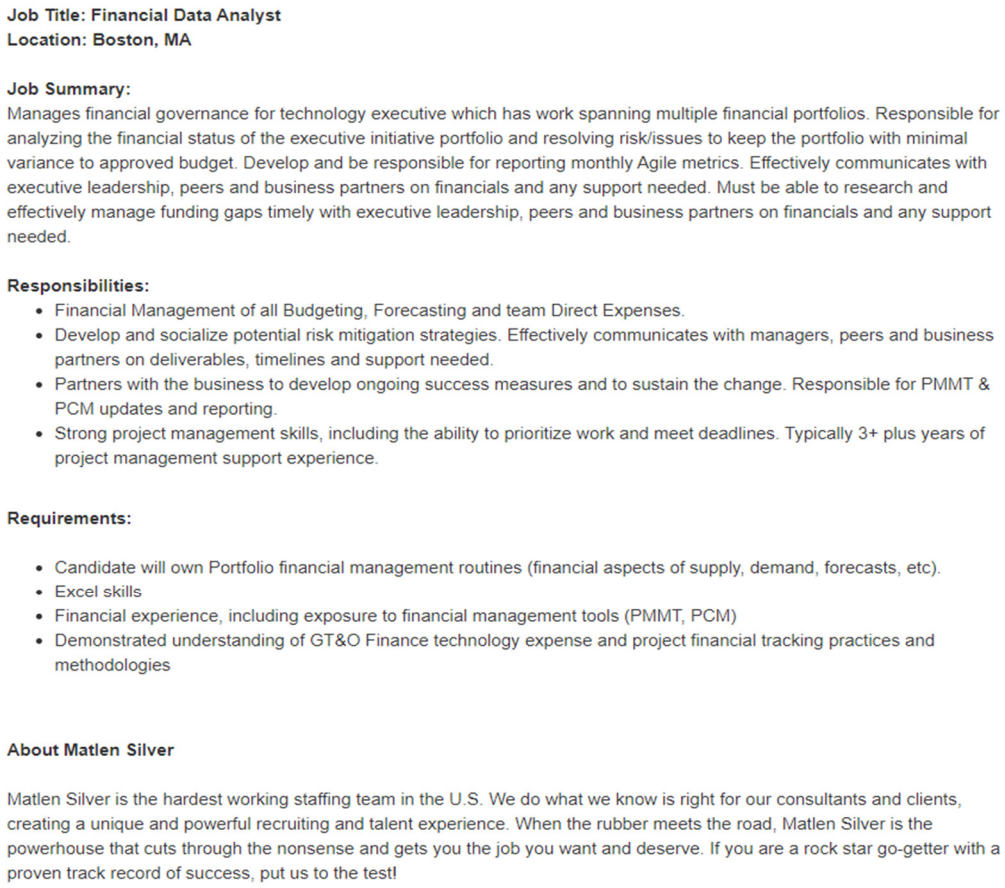
The first is job summary, and next is responsibilities, which is a list describing the scope of the job position, job requirements, required skills, a list of competencies of an ideal job applicant, required qualification, required work experience, necessary knowledge, good-to-have skills, salary, benefits, and information about the employer.

Some job advertisements combine the information in larger units. For instance, Requirements and Skills contains both requirements and skills for the given job position. Similarly, education, work experience, and knowledge are sometimes listed together [1–5].

Figure 1 contains a description of a sample job advertisement.

As Figure 1 reveals, a significant part of the job advertisement includes a description of job requirements. Job requirements are employers' requirements for the job position [6]. Such requirements often appear not only in the Requirements section, but in others as well, e.g., Competencies, Skills, Qualification, Work Experience, etc. This means that, if we

wanted to get all job requirements for the given job position, we would need to process the whole advertisement.



Job Title: Financial Data Analyst
Location: Boston, MA

Job Summary:
Manages financial governance for technology executive which has work spanning multiple financial portfolios. Responsible for analyzing the financial status of the executive initiative portfolio and resolving risk/issues to keep the portfolio with minimal variance to approved budget. Develop and be responsible for reporting monthly Agile metrics. Effectively communicates with executive leadership, peers and business partners on financials and any support needed. Must be able to research and effectively manage funding gaps timely with executive leadership, peers and business partners on financials and any support needed.

Responsibilities:
- Financial Management of all Budgeting, Forecasting and team Direct Expenses.
- Develop and socialize potential risk mitigation strategies. Effectively communicates with managers, peers and business partners on deliverables, timelines and support needed.
- Partners with the business to develop ongoing success measures and to sustain the change. Responsible for PMMT & PCM updates and reporting.
- Strong project management skills, including the ability to prioritize work and meet deadlines. Typically 3+ plus years of project management support experience.

Requirements:

- Candidate will own Portfolio financial management routines (financial aspects of supply, demand, forecasts, etc).
- Excel skills
- Financial experience, including exposure to financial management tools (PMMT, PCM)
- Demonstrated understanding of GT&O Finance technology expense and project financial tracking practices and methodologies

About Matlen Silver

Matlen Silver is the hardest working staffing team in the U.S. We do what we know is right for our consultants and clients, creating a unique and powerful recruiting and talent experience. When the rubber meets the road, Matlen Silver is the powerhouse that cuts through the nonsense and gets you the job you want and deserve. If you are a rock star go-getter with a proven track record of success, put us to the test!

Figure 1. Example of a job advertisement on the job portal www.dice.com (accessed on 31 August 2021).

Motivation

The primary motivation of this research is to find and correctly mark all job requirements in job advertisements. Correct detection of all job requirements not only enables their automated finding, but it also provides the opportunity for their further processing. It primarily concerns the possibility to extract job requirements for a job position, find current trends in the requirements, and find the requirements on qualification and necessary experience for a given job position, etc.

In this article, we are proposing an approach for automated detection of job requirements from job advertisements. The approach is based on machine learning using the SDCA logistic regression. We continuously tested and improved the approach based on experiments performed on selected job positions and related job advertisements.

The main contributions of the article are:

- Development of a machine learning module based on the SDCA logistic regression for automated detection of job requirements in job advertisements; the module is connected to data mining and implemented in the form of a fully functional system;
- Development and comparison of several postprocessing methods to improve marking of job requirements in job positions;
- The developed approach has been verified on 19 job positions and a large number of job advertisements; the success rate of marked job requirements has been assessed;
- Creation of a list of the most frequent job skills in job advertisements based on the Open Skills database extended with other typical job skills of selected IT positions.

2. Related Work

2.1. Analysis of Job Requirements and Skills

Recently, numerous research papers dealing with the analysis of job requirements and skills in job advertisements have appeared.

Kwon Lee and Han [7] carried out an analysis of the US labor market, where they showed that most universities emphasize skills in the area of hardware and operating systems. Nevertheless, employers do not usually demand them. Next, the results indicated that the job position "programmer/analyst" is expected to have skills in the area of software development, technical skills, and even business skills.

A study [8] created the ontology-based information extraction (OBIE) method to identify skills necessary for a job position described in a job advertisement. The developed method works with skills and requirements ontology (SARO), proposed by the authors.

In publication [9] web and text mining techniques are used to analyze a dataset of 244,460 job positions. Based on the analysis, the most frequent job requirements and job positions were detected. In addition, it enabled the detection of five main groups of job positions, where the processed job positions were classified.

Authors in study [10] analyzed 1216 job advertisements, aiming to find and classify required skills. The authors used their own framework based on the consensus-based pile-sort method (CPSP).

In an article [11], business and data analytics positions based on job advertisements were analyzed. The authors developed a rated list of relevant skills related to particular groups of skills for the analyzed job positions. They also found that decision-making, organization, communication, and structured data management are essential skills for all analyzed job position categories.

A journal paper [12], based on results published in [11], carried out an analysis of skill requirements in artificial intelligence and machine learning job advertisements. The result showed that technical skills, such as data mining, programing, statistics, and big data are more valued for ML positions than for AI positions. Conversely, AI positions tend to be more general from the point of view of skills description, although emphasizing communication skills.

In [13], the authors focused on job advertisements in the area of digital humanities (DH). The objective was to determine current required skills and knowledge appearing in job advertisements in DH. The authors experimented on a dataset of 72 job advertisements. They used various methods to analyze the frequency of keywords and phrases, and applied cluster analysis to detect job position requirements.

2.2. Machine Learning

Processing of job advertisements or their parts often uses one of the artificial intelligence tools—machine learning. Machine learning can be defined as computations using experience to improve performance or make accurate predictions [14,15].

Machine learning can be applied in various areas, including, among others [14]:

- Text or document classification;
- Natural language processing (NLP);
- Speech processing applications;
- Computer vision applications;
- Computational biology applications;
- Many other problems (fraud detection for a credit card, telephone, or insurance companies; network intrusion; learning to play games, such as chess or Go; medical diagnosis; the design of recommendation systems or search engines, etc.).

Using various machine learning approaches and algorithms, it is possible to classify job advertisements, identify and extract job requirements and skills from job advertisements, or search for and analyze competencies in job advertisements.

A journal article [16] carried out an automated classification of many job advertisements into various categories. They also extracted skills from the advertisements.

A study [17] downloaded Industry 4.0 advertisements from LinkedIn and used text mining and the TF-IDF algorithm to extract the most frequently used phrases representing skills and competencies.

Authors in study [18] classified jobs based on job requirements in job advertisements. They used semi-supervised machine learning techniques on a dataset of 37 million UK online job adverts collected by Burning Glass Technologies.

In paper [19], the authors created a system for automated identification of skills for German-language job advertisements. They used and compared several machine learning approaches for identification of the job position requirements. The result of their work was a list of the most required competencies in the dataset of 491 job positions.

A publication [20] used machine learning and natural language processing to analyze an Australian labor market's demand for skills of a PhD. The authors show how machine learning methods enable processing of large datasets, whose processing is not cost-efficient for human work.

In conference article [21], a high-performing machine learning approach to predict job skill shortages was implemented. A dataset of job data in Australia from 2012 to 2018 (7.7 million job advertisements) was used. The results show that job advertisements and employment statistics are the most valuable sources of information for predicting interannual changes in the job skill shortage.

The state-of-the-art analysis implies the following findings:

- Processing of job advertisements primarily emphasizes correct detection, extraction, and classification of job requirements and competencies for a given job position or a given job position dataset.
- Another objective is to find groups or clusters of related job requirements, skills, and competencies.
- Processing of job advertisements uses various methods of web mining, text mining, and machine learning.

The state-of-the-art analysis also implies the following, and limitations of machine learning techniques:

- SVM classifiers have good generalization ability; they are well suited to the particular characteristics of texts, namely, high dimensional feature spaces, few irrelevant features (dense concept vector), and sparse instance vectors [16].
- TF-IDF algorithm is a suitable method for text classification; limitations are: the algorithm cannot identify the words, even with a slight change in its tense, TF-IDF cannot check the semantic of the text in documents; to improve the performance and accuracy of the algorithm, other methods can be used (decision trees, rule-based classifiers, SVM classifiers, etc.).
- Natural language processing (NLP) methods are very suitable for information retrieval and information extraction in text and documents. The limitations of these methods are higher computational requirements and complexity of training sets.

3. System for Job Advertisement Processing

Our proposed system to process job advertisements serves to mine information on job requirements from job advertisements. The system connects to selected job portals containing various types of job positions. The system consists of the following modules:

- User interface;
- Data mining module (connects to web job portals);
- Machine learning module (extracts texts from job advertisements containing job requirements);
- Postprocessing module (follows up machine learning methods and tries to maximize the reliability and accuracy of the whole process based on information on the job advertisement text structure).

The architecture of the proposed system is depicted in Figure 2. The user interface serves as a gate for a user and communication between individual modules. The data mining module serves as a connection to web job portals. It mines data from job advertisements. The machine learning module uses machine learning methods to extract sentences containing job requirements. The postprocessing module follows up the machine learning module and increases the reliability and accuracy of the whole process based on information on the job advertisement text structure.

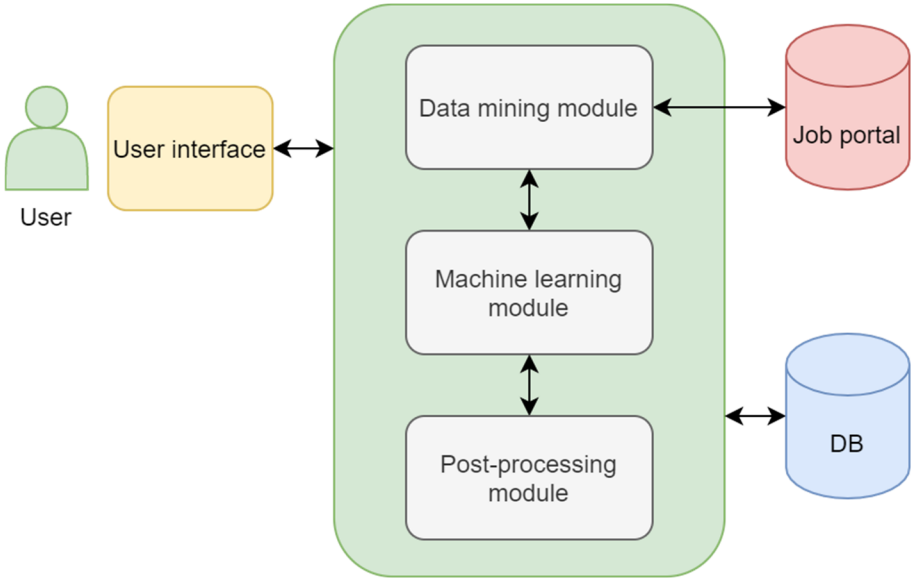


Figure 2. Architecture of the proposed system.

3.1. User Interface

The user communicates with the system through the user interface. This interface primarily contains an entry form to enter job position names, select the number of downloaded job advertisements, select the classification model, and select the postprocessing methods. A queue is then run to launch individual system modules. Once the data are processed, the results are stored in a database.

Other essential parts of the user interface display processed statistics for individual job positions and their advertisements, as well as statistics of a comparison with the Open Skills database and with the user's own database of skills.

3.2. Data Mining Module

The data mining module deals with loading and processing advertisements from a job portal. Searching for specific advertisements takes advantage of searching provided by the job portal. The input for this module is the job position name, which is used for searching through advertisements. There is also a possibility to reduce the maximum number of advertisements downloaded by the data mining module. The output is a list of objects containing the URL from which it was downloaded, its name, HTML of the advertisement text, and, finally, the advertisement text ready to be divided into individual sentences and classified into groups according to the membership to identified lists in the text.

The whole data mining process is described in the flowchart depicted in Figure 3. It is composed of the following steps:

1. Loading the job position name from the user.
2. Loading the HTML page of the job portal, where the job position name from the user is used as the search query in the URL (Figure 4).
3. The loaded HTML enables determination of the section with paging (the size of pages can be set using the query parameter of the user when loading the whole page), which

provides parsed information on the total number of pages found. This information is stored in the variable **count**.

4. The loaded HTML enables identification of links to advertisements from the first page of results. Those are stored in collection advertisements.
5. A cycle is run, with a counter starting at two up to the number of results in the count; in each iteration, an HTML page is downloaded from the job portal (Figure 5), where the page query parameter is the counter. All links to specific advertisements are stored, again, in collection **advertisements**.
6. Another cycle is run for all objects in the collection **advertisements**. First, it checks whether the advertisement has already been downloaded (using a database query). If yes, the advertisement text and HTML are loaded from the database. If not, a query is sent to the portal using the advertisement URL, and the advertisement text is loaded from the response HTML code to the query. Some job portals store advertisement data in the HTML page data, as well as in the JSON format (it is loaded, then, by JavaScript into the HTML page data). The JSON format contains, among other things, the job position name and a job position description in the HTML format (Figure 6). These two pieces of information are then stored for each advertisement into collection **advertisements**. The last step is to process this structured HTML using an algorithm, which is described in the following subchapter.

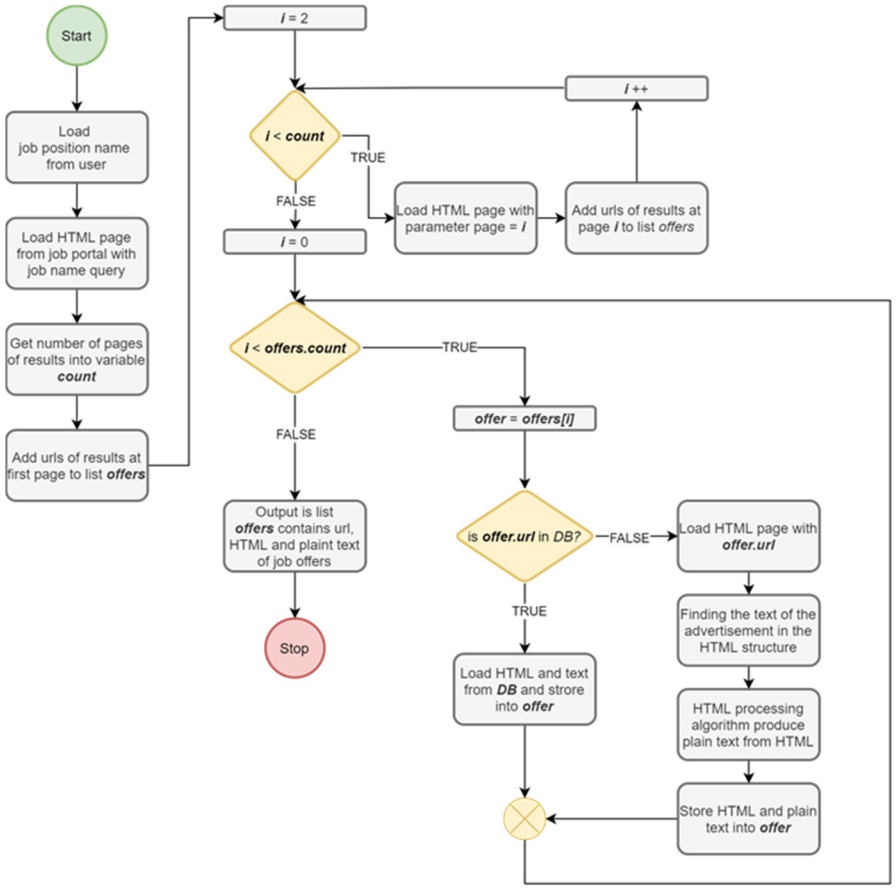


Figure 3. Data mining process from job advertisements.

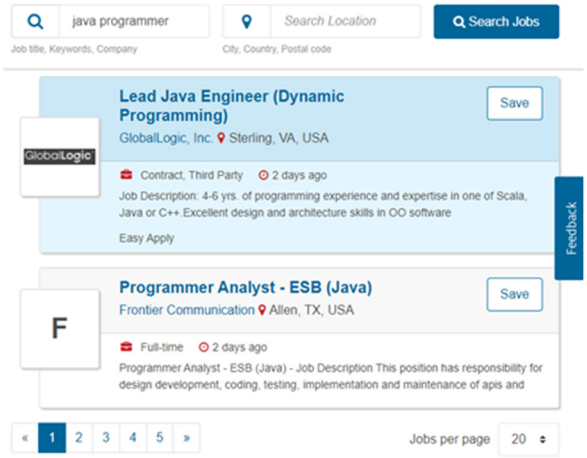


Figure 4. Example of search results using a search engine on a job portal www.dice.com (accessed on 31 August 2021).

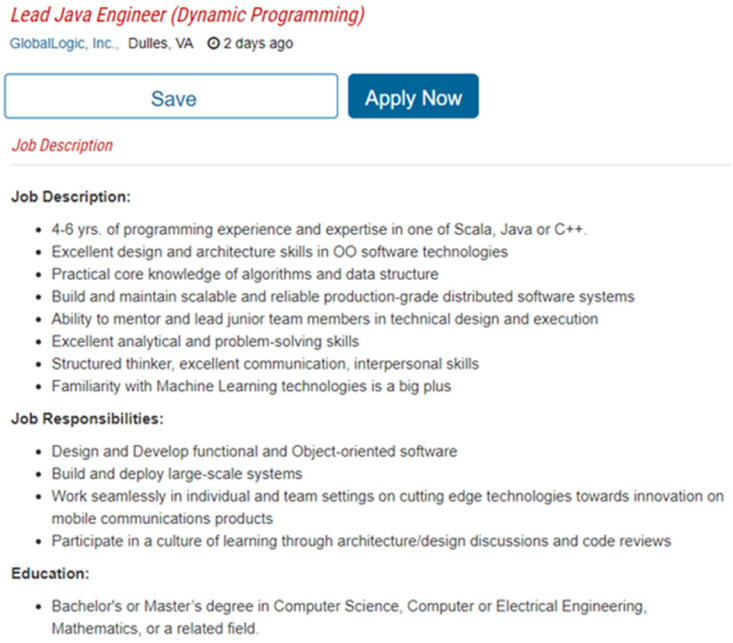


Figure 5. Sample job advertisement on a job portal www.dice.com (accessed on 31 August 2021).

Processing HTML Containing Job Advertisement Data

In this process, it is necessary to process an HTML page containing job advertisement data. The objective is to acquire clean text of the given job advertisement. The input is the job advertisement text in the HTML format. The output is a clean text divided into sentences separated by full stops. All HTML marks are removed, and marks for the beginnings and ends of item lists are added. The whole process is described in Algorithm 1.

```
{
    "url": "https://www.dice.com/jobs/detail/
    Lead-Java-Engineer-%28Dynamic-Programming%29-GlobalLogic%2C-Inc.
    -Dulles-VA-20101/RTL65472/6828362",
    "educationRequirements": "IT Diploma/Degree/Certification",
    "industry": "IT Software",
    "title": "Lead Java Engineer (Dynamic Programming)",
    "hiringOrganization": {
        "name": "GlobalLogic, Inc."
    },
    "skills":"Java, Design Pattern, Spring, Data Structure, Algorithms,
    Software development",
    "description": "<!DOCTYPE html><html><head></head><body><p><strong>Job
    Description:</strong></p><ul><li>4-6 yrs. of programming experience
    and expertise in one of Scala, Java or C++.</li><li>Excellent design
    and architecture skills in OO software technologies</li><li>Practical
    core knowledge of algorithms and data structure</li><li>Build and
    maintain scalable and reliable production-grade distributed software
    systems</li><li>Ability to mentor and lead junior team members in
    technical design and execution</li><li>Excellent analytical and
    problem-solving skills</li><li>Structured thinker, excellent
    communication, interpersonal skills</li><li>Familiarity with Machine
    Learning technologies is a big plus</li></ul><p><strong> </
    strong></p><p><strong>Job Responsibilities:</strong></p><ul><li>Design
    and Develop functional and Object-oriented software</li><li>Build and
    deploy large-scale systems</li><li>Work seamlessly in individual and
    team settings on cutting edge technologies towards innovation on
    mobile communications products</li><li>Participate in a culture of
    learning through architecture/design discussions and code reviews</
    li></ul><p><strong> </strong></p><p><strong>Education: </strong></
    p><ul><li>Bachelor's or Master&rsquo;s degree in Computer Science,
    Computer or Electrical Engineering, Mathematics, or a related field.</
    li></ul></body></html>"
}
```


Figure 6. Example of job advertisement data in the JSON format on a job portal www.dice.com (accessed on 31 August 2021).

Algorithm 1. Processing an HTML page containing job advertisement data.

Input:
B represents a set of HTML tags with no meaning regarding text division into sentences containing individual information on job requirements {"b", "strong", "div", "em"}
S represent a set of HTML tags usually separating sentences {"p", "li", "br"}
L represents a set of HTML tags indicating standard bulleted lists {"ul", "ol"}
N represents a set of strings that can bullet an item of a manually created list {"o", "•", "*", "-", "•"}
AD represents a set of texts that will be substituted by a full stop in the advertisement text {" .", "•", "..", ": ", ":.", ". ."}
T represents an HTML text of the job advertisement
Paragraphs are all HTML nodes "p" from **T**, whose internal text begins with "-"
Output:
Clean text divided into sentences (separated by full stops).
foreach(B_i) in B do
{
  remove all occurrences of $\mathbf{B_i}$ in **T** with preserving the contents of the tag
}

Algorithm 1. *Cont*.

```
if(Paragraphs != null and Paragraphs.Count > 1)
{
   foreach(Pi) in Paragraphs do
   {
      if(Paragraphs do not contain the previous HTML node from Pi)
      {
         add text ". $begin$. " to T before Pi
      }
      else if(Paragraphs do not contain the following HTML node from Pi)
      {
         add text ". $end$. " to T after Pi
      }
      removing the "p" tag with preserving the internal content
   }
}
foreach(Si) in S do
{
   Nodes = all HTML nodes Si from T
   foreach(NDi) in Nodes do
   {
      add text ". " to T before NDi
      add text ". " to T after NDi
      removing the tag NDi with preserving the internal content
   }
}
foreach(Li) in L do
{
Nodes = all HTML nodes Li from T
foreach(NDi) in Nodes do
{
      add text ". $end$. " to T before NDi
      add text ". $begin$. " to T after NDi
      removing the tag NDi with preserving the internal content
   }
}
ListSigh = null
StartAdded = false
ListNodes = list
foreach(NDi) in T do
{
   Text = internal text NDi
      if(ListSign != null and Text starts with ListSign)
      {
         if(StartAdded == false)
         {
            StartAdded = true
            add text ". $begin$. " before ListNodes[first] to T
            }
            add NDi to ListNodes
   }
   else
```

Algorithm 1. *Cont.*

```
    {
      if(ListSign != null)
      {
        ListSign = null
        StartAdded = false
        if(ListNodes.Count > 1)
        {
            add text ". $end$. " after ListNodes[last] to T
          }
          clean ListNodes
    }
    foreach(Ni) in N do
    {
    if(Text starts with Ni)
    {
      ListSign = Ni
      add NDi to ListNodes
      break
        }
      }
    }
  }
  foreach(Ti) in T do
  {
  remove the remaining HTML tags from Ti
  }
  foreach(Ti) in T do
  {
  replacement of all occurrences of texts from AD to "."
  }
  foreach(Ti) in T do
  {
  decoding the remaining HTML characters to UTF-8 character
  }
```

In addition, the sentences are separated using full stops and stored in a collection of objects. When creating the collection, the sentence between texts ". $begin$. " and ". end. " are assigned with a group ID (UUID), which is then used during postprocessing.

3.3. Machine Learning Module

The machine learning module serves to mark job advertisement sentences containing job requirements. It concerns the task of binary classification. The input into the predictive part is an individual sentence of job advertisements, and the output is their binary membership to a set of sentences containing job requirements.

3.3.1. Training Set

This module works with a training set containing sentences and their membership to a set of job requirements. We used a training set in the TSV format, which is a text file with each line containing a sentence, tabulator, and flag 0 or 1, which determines if the given sentence contains information on job requirements or not. Figure 7 depicts part of the training set.

```
Install, maintain, and upgrade as necessary all Company computer software.  1
7+ years experience in software development.    1
3+ years Java development experience    1
3+ years of web development experience  1
Programming skills using Java EE, Tomcat, HTML5, CSS, JavaScript, AngularJS, JSON.  1
Looking for a Senior Full Stack Java consultant to assist with refinement of requirements,
design, development and support of complex Java/JEE programs.    0
Talent Software Services is in search of a Full Stack Java Programmer for a contract
position in Richmond, VA   0
Opportunity will be five months with a strong chance for a long-term extension. 0
```

Figure 7. Part of the training set to verify if the sentence contains job requirements.

The figure clearly shows that sentences marked with flag 1 contain job requirements. Sentences marked with flag 0 do not contain job requirements, although some contain keywords or phrases used in job requirements. A sample sentence: "Looking for a Senior Full Stack Java consultant to assist with refinement of requirements, design, development and support of complex Java/JEE programs".

There are several strategies for the creation of the initial training set. In the case of a generally used domain, it is possible to find a ready set. For the case of sentences containing job requirements, there is no such generally used dataset.

The most straightforward and universally usable strategy is a manual creation of the training set. Another possibility is to use already existing resources, such as Open Skills, in the case of information on job requirements, even though it might raise some issues.

The first issue is that the Open Skills database (Crockett, Lin, Gee, and Sung, 2018) contains a database of competencies, and skills and their relations, which does not correspond to whole sentences processed from job advertisements. The second issue is that the correct functionality of the training within the machine learning methods requires that the training set includes several negative examples.

Therefore, we created the initial training set manually using real data acquired directly from job advertisements.

3.3.2. Machine Learning Methods

The type of machine learning that is necessary to solve in this system is binary classification. Several machine learning methods can solve this type of task:

- Averaged perceptron;
- L-BFGS;
- Symbolic stochastic gradient descent;
- SDCA logistic regression.

An averaged perceptron is considered suitable for text classification, although it requires refining of input parameters, which is impossible considering a small training set and its gradual extension.

L-BFGS is recommended in the case of a large number of characteristics in the input data.

Symbolic stochastic gradient descent is considered a fast and accurate method, although it also requires refining of the input parameters.

SDCA logistic regression provides the best results in its default setting, which is crucial for the given application. That is why we have implemented it in our proposed system.

SDCA

Stochastic dual co-ordinate ascent (SDCA) is a general machine learning method for (generalized) linear models, such as linear regression or logistic regression [22]. It regards a popular method for solving regularized loss minimization for the case of convex losses [23].

Moreover, the SDCA method provides more accurate results than other machine learning methods, such as PLS, PLS-DA, and SIMCA [24]. So, the SDCA method is appropriate for our proposed system.

In this article, we work with the SDCA method for logistic regression.

Logistic regression is a statistical method dealing with probability estimation of a specific phenomenon (dependent variable) based on known facts (independent variables) that can affect the phenomenon occurrence.

The logistic regression method assumes that, under conditions defined by vector x, a random quantity $Y(x)$ will equal 1, with a probability P whose dependence on x can be expressed using a so-called logistic function, written as:

$$P[Y(x) = 1] = \frac{\exp(\beta\prime x)}{1 + \exp(\beta\prime x)}$$

Vector β is a vector of unknown parameters and exp is an exponential function. To estimate vector β means to estimate the searched probability of the occurrence of the investigated phenomenon (supposing parametrization by logistic function). The SDCA method was applied to solve various problems [25–28].

In our work, we use logistic regression to determine which sentences or other separable parts of the job advertisement text are job requirements. This approach was selected due to a constantly changing training set (model, respectively). Considering the nature of the investigated problem, no generally accepted dataset would be usable for the training phase of the machine learning methods. Therefore, the algorithm must achieve good results without the need to refine various input parameters. It is the SDCA logistic regression that possesses the required attributes. SDCA logistic regression uses several hyperparameters (parameters whose value is used to control the learning process). In our implementation, we use hyperparameters L1Regularization and L2Regularization. These parameters were set to default values.

3.3.3. Prediction Improvement by a Feedback Mechanism

Prediction improvement is achieved by using a simple feedback mechanism. The user uses it on a special system page which serves to test and improve this module functionality.

The whole process begins with marking (highlighting) sentences that have been marked by the system as job requirements when processing a job advertisement. Such an adapted text is displayed to the user (Figure 8).

Our client is seeking a Java Programmer to join their team to help move their existing applications to the cloud. This is an excellent opportunity to gain great experience in an enterprise environment and develop your career.

Job Requirements:

1. 3 years of Java experience developing in a formal enterprise environment.
2. 3 years of experience writing API
3. 2 years of experience with JavaScript, Spring and Struts

Preferred Experience:

1. AWS or Azure experience
2. Insurance background

EEO Employer

Apex Systems is an equal opportunity employer. We do not discriminate or allow discrimination on the basis of race, color, religion, creed, sex (including pregnancy, childbirth, breastfeeding, or related medical conditions), age, sexual orientation, gender identity, national origin, ancestry, citizenship, genetic information, registered domestic partner status, marital status, disability, status as a crime victim, protected veteran status, political affiliation, union membership, or any other characteristic protected by law. Apex will consider qualified applicants with criminal histories in a manner consistent with the requirements of applicable law. If you have visited our website in search of information on employment opportunities or to apply for a position, and you require an accommodation in using our website for a search or application, please contact our Employee Services Department at or

Figure 8. Processed job advertisement containing green-marked sentences with job requirements.

Having marked the selected sentences by a mouse cursor, a panel (Figure 9) is displayed. The panel contains buttons to mark the sentence positively or negatively. In

addition, it has a counter of positively marked sentences in the selection and a counter of all sentences in the selection.

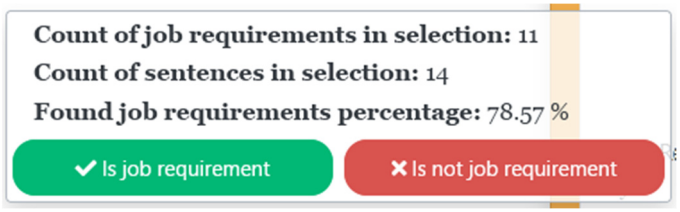


Figure 9. Panel to determine if a sentence contains job requirements or not.

For the feedback purposes, there is a database table created which contains all user interventions into the training dataset. Once a button is pressed, the table is searched through. If it contains a record with a currently highlighted sentence, the counter of positively or negatively marked sentences increases by 1. This can filter out the effects of a human factor on the whole model. If one sentence is, for example, marked three times positive (as containing job requirements) and the user makes a mistake and marks it once negatively, it does not affect the training dataset. Part of the database table containing user feedback records is depicted in Table 1.

Table 1. Part of the database table containing user feedback records.

ID	Text	Positive Counts	Negative Counts
1	kindly research us at	0	1
2	job id	0	1
3	assist in the selection of computer software	1	0
4	m/f/disability/veterans	0	1
5	required skills	0	2
6	works with necessary personnel to determine if modifications are necessary with interested personnel to determine the necessity for modifications or enhancements	1	0
7	design, develop, document, analyse, create, test and modify computer programs.	1	0
8	codes enhancement and development programs and/or required fixes to production problems using the functional and technical programming standards	1	0
9	business group highlights	0	1
10	install, maintain, and upgrade as necessary all company computer software	1	0
11	title	0	2

After a query over the table, the sentence in the training set is adapted (if the positive/negative markings ratio changes) or added to the training set. Then, the training process is performed, and a new model is created. This model serves to mark job requirements in a job advertisement.

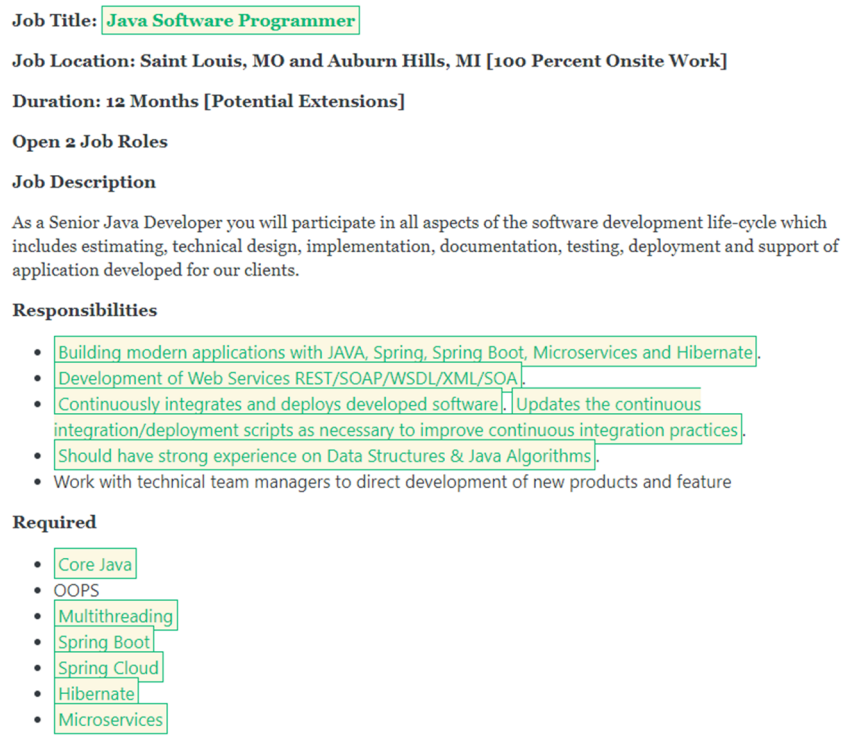
3.4. Postprocessing Module

The postprocessing module contains several approaches to improve the overall success rate when extracting sentences containing job requirements from job advertisement texts. It uses information on the text structure.

Before describing individual methods, it is necessary to describe a term, denoted as so-called lists containing job requirements. The aim is to detect these lists in advertisements correctly and to mark all job requirements in these lists.

Lists containing requirements

This concerns lists containing job requirements. Figure 10 contains data of a job advertisement with lists called Responsibilities and Required. Both of these lists describe necessary job requirements for the job position. Our objective is to use machine learning and postprocessing methods to correctly detect all such lists in a job advertisement and mark all job requirements in these lists. Correct detection of these lists is crucial, as advertisements can also contain lists without job requirements (e.g., information on salary conditions, benefits, and other auxiliary information). An example of such lists is shown in Figure 11 (lists called Tech Breakdown, The offer, You will receive the following benefits).


Job Title: Java Software Programmer

Job Location: Saint Louis, MO and Auburn Hills, MI [100 Percent Onsite Work]

Duration: 12 Months [Potential Extensions]

Open 2 Job Roles

Job Description

As a Senior Java Developer you will participate in all aspects of the software development life-cycle which includes estimating, technical design, implementation, documentation, testing, deployment and support of application developed for our clients.

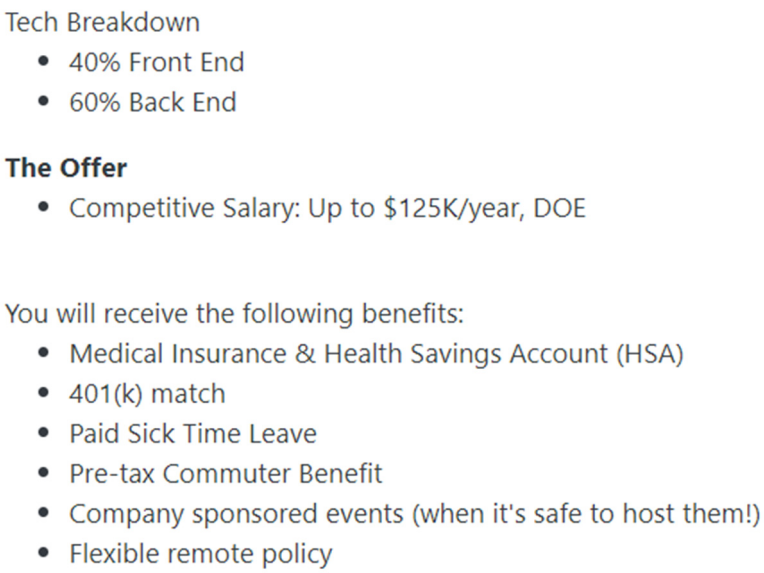
Responsibilities

- Building modern applications with JAVA, Spring, Spring Boot, Microservices and Hibernate.
- Development of Web Services REST/SOAP/WSDL/XML/SOA.
- Continuously integrates and deploys developed software. Updates the continuous integration/deployment scripts as necessary to improve continuous integration practices.
- Should have strong experience on Data Structures & Java Algorithms.
- Work with technical team managers to direct development of new products and feature

Required

- Core Java
- OOPS
- Multithreading
- Spring Boot
- Spring Cloud
- Hibernate
- Microservices

Figure 10. A job advertisement with two lists containing job requirements.


Tech Breakdown

- 40% Front End
- 60% Back End

The Offer

- Competitive Salary: Up to $125K/year, DOE

You will receive the following benefits:

- Medical Insurance & Health Savings Account (HSA)
- 401(k) match
- Paid Sick Time Leave
- Pre-tax Commuter Benefit
- Company sponsored events (when it's safe to host them!)
- Flexible remote policy

Figure 11. A job advertisement with lists missing job requirements.

3.4.1. Basic Machine Learning

This approach uses an SDCA-based machine learning with the help of a training set. This method does not use any other postprocessing method to increase the success rate of detected and marked job requirements in job advertisements. Figure 12 shows a job advertisement processed by the basic machine learning method.

Figure 12. Part of a job advertisement processed by the basic machine learning method.

The figure clearly shows that processing a job advertisement by this method provides promising results. Most job requirements were marked correctly. However, some job requirements were not marked in the lists.

Job requirements not marked in **Required skills/Certifications** lists were as follows:

- Authentication/Authorization framework;
- DevOps, Continuous Integration/Build/Quality Tools;
- Web Security;
- Terms: Redis, JUnit, Mockito.

Job requirements not marked in **Desired skills/Certifications** lists were as follows:

- Responsive Web, CSS3, HTML5;
- MEAN stack/REACT;
- AWS/Cloud.

The item lists reveal that the not-marked items were mostly names of technologies and software tools for software development.

3.4.2. Machine Learning with Marking of Sentences

In this case, sentences in the advertisement are searched through, and, if there is one negatively marked sentence between two positively marked sentences, it is automatically marked positive as well.

In addition, isolated positive occurrences are marked negatively. The idea is that job requirements are generally grouped in a job advertisement.

Figure 13 shows an identical job advertisement text, but this time the marking took advantage not only of machine learning, but an intelligent algorithm as well. It marks previously not-marked sentences in the lists (e.g., Web Security, Redis, JUnit, Mockito). It also marks the item called technical emphasis in the following areas, although it does not contain job requirements.

Figure 13. Part of a job advertisement processed by the method called machine learning with marking of sentences between positive occurrences.

3.4.3. Machine Learning with Marking Lists

This approach continues in using the idea that job requirements in job advertisements usually appear in lists. List processing also takes advantage of the fact that job advertisements are in the form of an HTML format, which enables bulleted text based on specific tags (e.g., "<ul>" and "<li>") to be found. The process of tag processing and sentence grouping is described in the chapter "processing an HTML page containing job advertisement data".

Nevertheless, not all job advertisements contain lists created by valid HTML marks. Thus, we tried to solve various nonstandard situations when processing a job advertisement text and performed several experiments to achieve as accurate a list detection as possible. Another critical factor is that some lists do not contain job requirements (e.g., information on benefits, company management, salary conditions, etc.).

Thus, this approach works with detected lists containing job requirements. When analyzing each list, we determine whether more than 50% of sentences are marked as job requirements. If yes, all sentences in the list are marked as job requirements.

An example can be a list called **Required Skills/Certifications** and its sub-list **Technical emphasis in the following areas**, shown in Figure 12.

In total, 10 out of 13 items in the list are marked as job requirements, and three items are not marked. These concern the following:

- Authentication/Authorization framework;
- DevOps, Continuous Integration/Build/Quality Tools;
- Web Security.

As it concerns a list containing job requirements, based on the fulfilled condition (50% and more sentences marked as job requirements), the remaining items will also be marked as job requirements. The result is depicted in Figure 14.

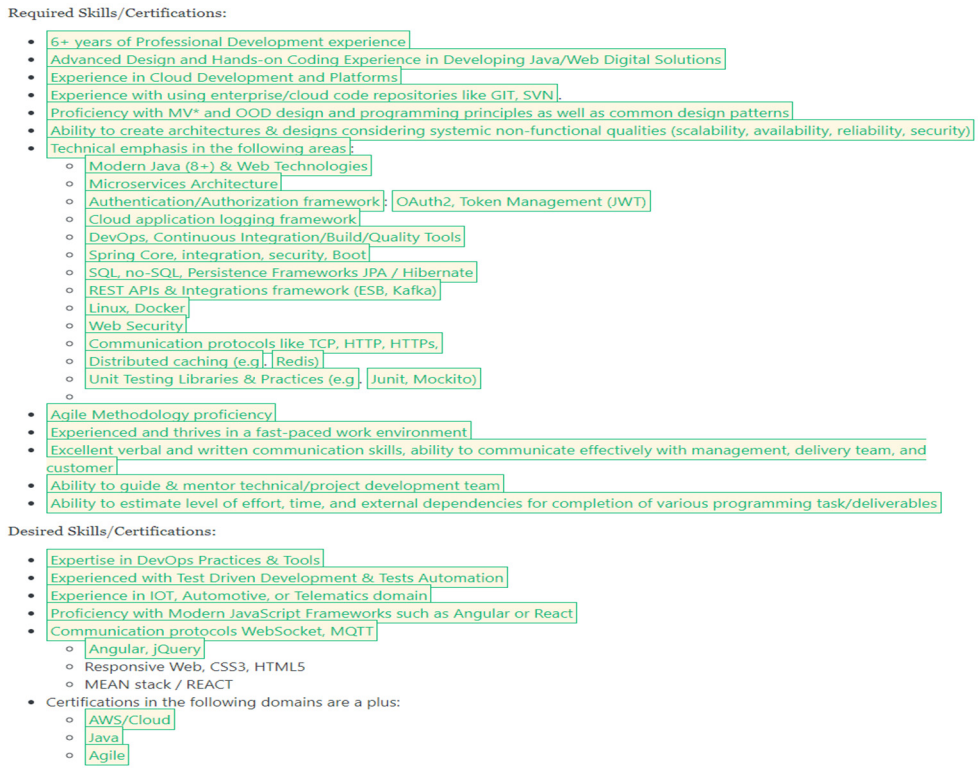


Figure 14. Part of a job advertisement processed by the method called machine learning with marking lists.

The figure reveals that all items are marked in the list **Required Skills/Certifications**. The same holds for sub-list **Certifications in the following domains are plus** (unlike in the basic machine learning approach, it lacked AWS/Cloud). We can also see that the list **Desired skills/Certifications** did not have the following job requirements marked:

- Responsive Web, CSS3, HTML5
- MEAN stack/REACT

This is caused by the fact that part of the sub-list contains another two items that were not marked (Angular, jQuery). In this case, the condition for marking the whole list is not fulfilled, as those two items remain unmarked.

3.4.4. Machine Learning with Improved Marking Lists

This approach is similar to the machine learning with marking lists. The only difference is a condition for marking all items on the list containing job requirements. The

condition is: all items on the list are marked if the list contains one or more items (sentences) marked as a job requirement. The result is depicted in Figure 15.

Figure 15. Part of a job advertisement processed by the method called machine learning with improved marking lists.

The figure clearly reveals that, in this approach, all items on the lists containing job requirements were marked. In addition, it can also be seen that, when processing the HTML text of the advertisement, the main item lists were detected (**Required skills/Certifications** or **Desired skills/Certifications**) together with sub-lists (e.g., **Certifications in the following domains are a plus**). Only item "Technical emphasis in the following areas" is marked incorrectly.

In general, this approach is the best of all that were tested and experimented with in the postprocessing module. The validation of the marked job requirements lists in the job advertisement showed that this approach functions properly and correctly marks job requirements in lists containing only one item marked as a job requirement.

4. Results and Discussion

In order to verify the mining system, we performed several experiments. In the performed experiments, we worked with the following input data:

- The 20 most searched IT job positions from job portal Dice.com (www.dice.com—accessed on 31 August 2021);
- A total of 9971 job advertisements analyzed;
- The SDCA logistic regression method was used as a classification model of the machine learning.

The system verification worked with the following data processing approaches:

- Basic machine learning;

- Machine learning with marking of sentences between positive occurrences;
- Machine learning with marking lists;
- Machine learning with improved marking lists.

These approaches are described in detail in Section 3.4 Postprocessing Module.

During the experiments, we discovered that only five job advertisements were loaded for one job position, namely, "game developer". Therefore, this job position was removed from further experiments.

Table 2 shows the results of the basic machine learning approach. The table contains the following columns:

- **Name of the job position**—name of the job position that was analyzed and whose job advertisements were processed.
- **Number of processed job advertisements**—number of advertisements found and processed for a given job position.
- **Number of job advertisements with all job requirements found**—number of job advertisements where all job requirements were found. This means all items of the detected lists containing job requirements that were marked as job requirements.
- **Ratio of the job advertisements with found job requirements to all job advertisements**—the ratio of the advertisements with all job requirements to all processed advertisements found, i.e., the success rate of finding all requirements in the advertisements.

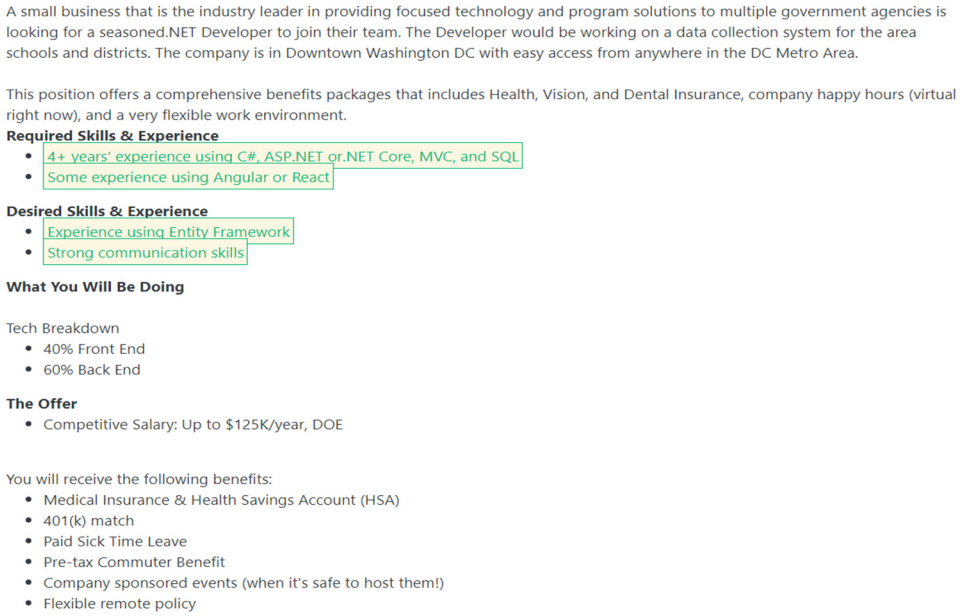
Table 2. Results of processing job advertisements using the basic machine learning approach.

Name of the Job Position	Number of Processed Job Advertisements	Number of Job Advertisements with All Job Requirements Found	Ratio of the Job Advertisements with Found Job Requirements to All Job Advertisements
linux administrator	137	2	**2%**
mobile developer	177	15	8%
python developer	392	40	**10%**
.net developer	991	94	9%
machine learning engineer	418	20	5%
backend developer	342	63	**18%**
network administrator	146	6	4%
solution architect	919	35	4%
frontend developer	430	44	10%
sql developer	318	29	9%
android developer	280	35	**13%**
product manager	989	27	**3%**
web developer	440	32	7%
scrum master	995	26	**3%**
project manager	999	32	3%
php developer	180	13	7%
java developer	1000	74	7%
full stack developer	998	60	6%
data analyst	824	45	5%
Average	**577**	**36**	**7%**

The table reveals that, in this approach, all job requirements were marked only in, on average, 7% of the advertisements. Three job advertisements are in bold, where the ratio of the found job requirements is the lowest (2%, 3%, and 3%); bold/underlined are the three job advertisements with the highest ratio of the found job requirements (10%, 18%, and 13%). This approach is not very successful, which is mainly caused by the lack of a postprocessing phase.

Figure 16 contain a sample job advertisement with all job requirements correctly marked and found. The advertisement has two lists that are detected as lists containing job requirements. It concerns **Required Skills & Experience** and **Desired Skills & Experience**. The figure reveals that all items marked as job requirements were marked correctly in

these lists. In addition, the advertisement text contains other lists **What you will be doing**, **The Offer**, and **You will receive the following benefits**. These lists do not contain job requirements and, thus, were not marked as lists containing job requirements, nor were any items marked as sentences containing job requirements. The same holds for another job advertisement. All job requirements in this advertisement were correctly detected and marked.


A small business that is the industry leader in providing focused technology and program solutions to multiple government agencies is looking for a seasoned.NET Developer to join their team. The Developer would be working on a data collection system for the area schools and districts. The company is in Downtown Washington DC with easy access from anywhere in the DC Metro Area.

This position offers a comprehensive benefits packages that includes Health, Vision, and Dental Insurance, company happy hours (virtual right now), and a very flexible work environment.

Required Skills & Experience

- 4+ years' experience using C#, ASP.NET or.NET Core, MVC, and SQL
- Some experience using Angular or React

Desired Skills & Experience

- Experience using Entity Framework
- Strong communication skills

What You Will Be Doing

Tech Breakdown

- 40% Front End
- 60% Back End

The Offer

- Competitive Salary: Up to $125K/year, DOE

You will receive the following benefits:

- Medical Insurance & Health Savings Account (HSA)
- 401(k) match
- Paid Sick Time Leave
- Pre-tax Commuter Benefit
- Company sponsored events (when it's safe to host them!)
- Flexible remote policy

Figure 16. Job advertisement for a .NET developer position with all job requirements found.

Figure 17 shows a graph of the success rate of finding job requirements in individual job positions. The graph shows that the highest success rate was in positions Backend developer and Android developer. On the other hand, the lowest success rate was in Linux administrator, Product manager, Scrum master, and Project manager.

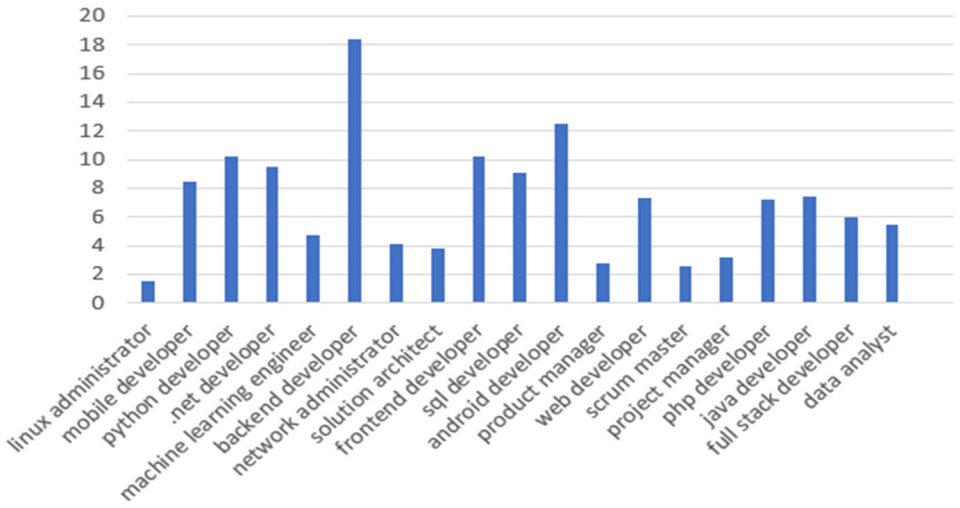


Figure 17. Success rate of finding job requirements in job positions—basic machine learning approach.

We also performed experiments with the same job positions and processed job advertisements for another approach: machine learning with marking of sentences between positive occurrences. Table 3 shows the results of this approach.

Table 3. Results of processing job advertisements using the machine with marking of sentences between positive occurrences approach.

Name of the Job Position	Number of Processed Job Advertisements	Number of Job Advertisements with All Job Requirements Found	Ratio of the Job Advertisements with Found Job Requirements to All Job Advertisements
linux administrator	137	5	**4%**
mobile developer	177	41	**23%**
python developer	392	83	**21%**
.net developer	991	180	18%
machine learning engineer	418	63	15%
backend developer	342	100	**29%**
network administrator	146	10	**7%**
solution architect	919	93	10%
frontend developer	430	79	18%
sql developer	318	50	16%
android developer	280	60	21%
product manager	989	73	7%
web developer	440	78	18%
scrum master	995	68	7%
project manager	999	95	10%
php developer	180	21	12%
java developer	1000	142	15%
full stack developer	998	59	**6%**
data analyst	824	101	12%
Average	**577**	**74**	**14%**

Using this approach led to the increase in the successfully found job requirements in 16% of the advertisements. The highest percentage of the best job positions is also higher in this approach, namely, 23%, 21%, and 29% of job advertisements with all job requirements found.

Figure 18 shows a graph of the success rate of finding job requirements in individual job positions. It is clear that four job positions scored a more than 20% success rate.

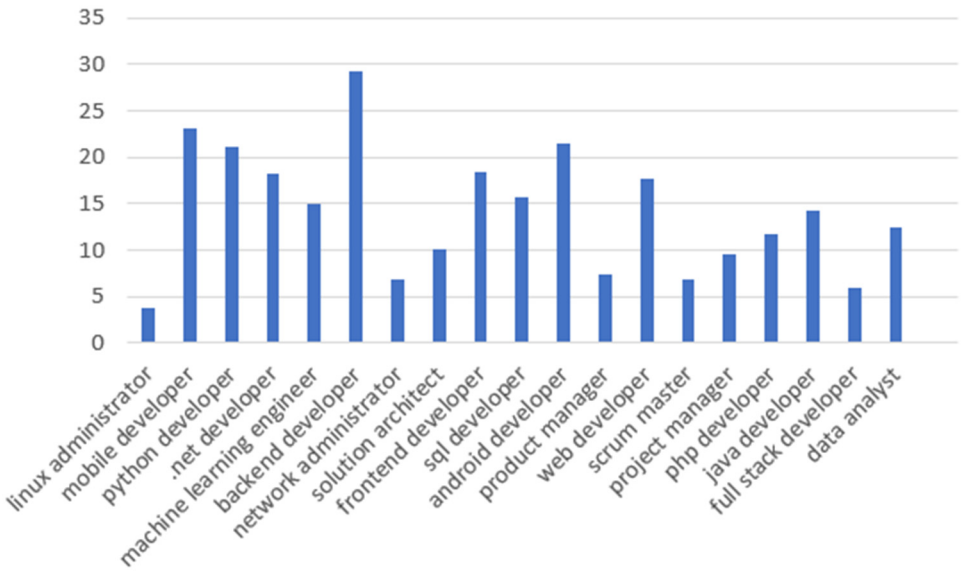


Figure 18. Success rate of finding job requirements in individual job positions—machine learning with marking of sentences between positive occurrences approach.

We also performed experiments with the machine learning with marking lists approach. Table 4 shows the results of this approach.

Table 4. Results of processing job advertisements using the machine learning with marking lists approach.

Name of the Job Position	Number of Processed Job Advertisements	Number of Job Advertisements with All Job Requirements Found	Ratio of the Job Advertisements with Found Job Requirements to All Job Advertisements
linux administrator	137	63	48%
mobile developer	177	103	**58%**
python developer	392	197	50%
.net developer	991	563	**57%**
machine learning engineer	418	171	41%
backend developer	342	195	57%
network administrator	146	72	49%
solution architect	919	360	39%
frontend developer	430	228	53%
sql developer	318	186	**58%**
android developer	280	145	52%
product manager	989	273	**28%**
web developer	440	235	53%
scrum master	995	408	41%
project manager	999	286	**29%**
php developer	180	94	52%
java developer	1000	467	47%
full stack developer	998	550	55%
data analyst	824	278	**34%**
Average	**577**	**257**	**47%**

This approach led to another increase in the success rate of found job requirements. It was in 47% of all job advertisements where all job requirements were found. This approach brought a significant improvement in marking job requirements. In 10 job positions out of 19, it concerned more than 50% of found job requirements.

Figure 19 shows a graph of the success rate of finding job requirements in individual job positions. The success rate is higher than 25% in all job positions; most of them scored more than 50%. Only three job positions are less than 30%. The best score was achieved for the job positions of Mobile Developer, .NET developer, and SQL developer.

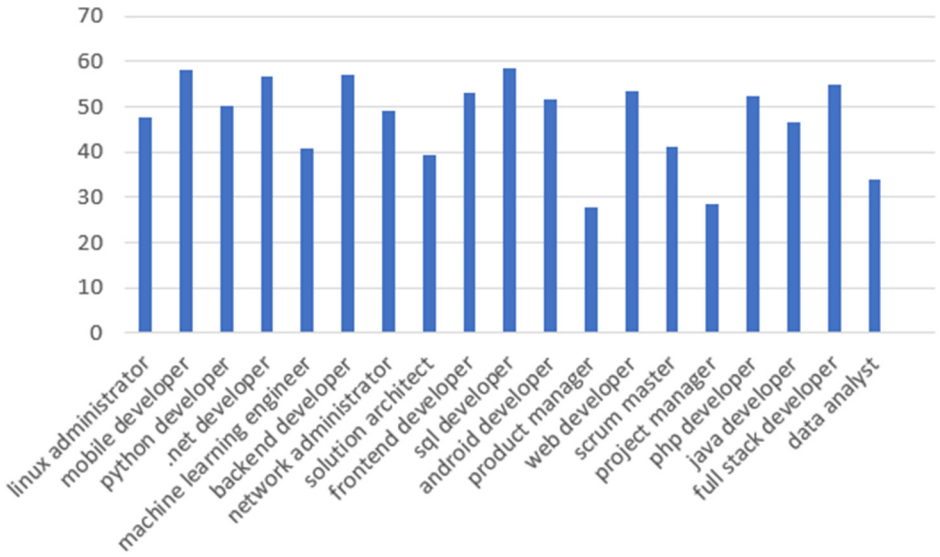


Figure 19. Success rate of finding job requirements in individual job positions—machine learning with marking lists approach.

Finally, we performed experiments with the machine learning with improved marking lists approach. Table 5 shows the results.

Table 5. Results of processing job advertisements using the machine learning with improved marking lists approach.

Name of the Job Position	Number of Processed Job Advertisements	Number of Job Advertisements with All Job Requirements Found	Ratio of the Job Advertisements with Found Job Requirements to All Job Advertisements
linux administrator	137	95	**72%**
mobile developer	177	148	**84%**
python developer	392	298	**76%**
.net developer	991	762	77%
machine learning engineer	418	329	79%
backend developer	342	287	83%
network administrator	146	116	79%
solution architect	919	731	80%
frontend developer	430	330	77%
sql developer	318	252	79%
android developer	280	198	**71%**
product manager	989	720	73%
web developer	440	331	75%
scrum master	995	788	79%
project manager	999	748	75%
php developer	180	145	81%
java developer	1000	836	**84%**
full stack developer	998	931	**93%**
data analyst	824	578	**71%**
Average	**577**	**454**	**80%**

The results of this approach are promising. This approach achieved the best results in marking job requirements. It has an average of 80% of the job advertisements with correctly marked job requirements. In total, 15 job positions scored a more than 75% success rate. The Full Stack developer even scored 93%. This approach provides the best results, and it implies its usability for the detection of job requirements in job advertisements.

Figure 20 shows a graph of the success rate of finding job requirements in individual job positions. The figure shows that the success rate is higher than 70% in all job positions. Most job positions scored more than 75%. The best results were achieved by the job positions Full Stack developer, Mobile developer, and Java developer.

A comparison of the results of finding job requirements by individual approaches is provided in Table 6. The comparison reveals that the highest success rate is achieved by the machine learning with improved marking lists approach. This approach scored an average 80% success rate in finding job requirements.

Figure 21 shows a graph of the success rate of finding job requirements in individual job positions for individual approaches. The columns represent all four tested approaches for each analyzed position. The figure reveals a significant rise in the success rate. The best results were achieved for the job positions Full Stack developer (93%), followed by Mobile developer, and Java developer (84%).

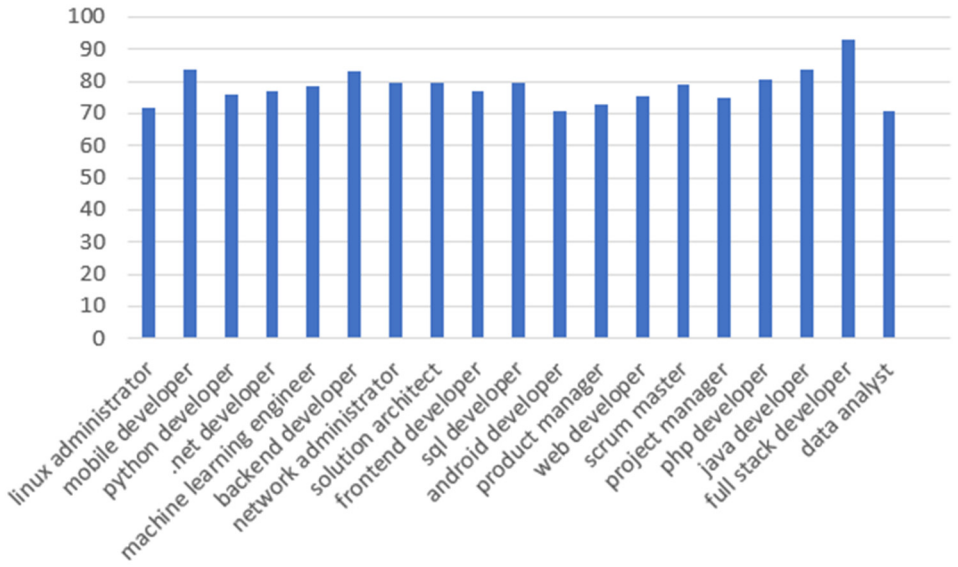


Figure 20. Success rate of finding job requirements in individual job positions—machine learning with improved marking lists approach.

Table 6. Comparison of individual approaches.

Name of the Job Position	Basic Machine Learning	Machine Learning with Marking of Sentences between Positive Occurrences	Machine Learning with Marking Lists	Machine Learning with Improved Marking Lists
linux administrator	**2%**	**4%**	48%	**72%**
mobile developer	8%	23%	**58%**	**84%**
python developer	**10%**	**21%**	50%	**76%**
.net developer	9%	18%	**57%**	77%
machine learning engineer	5%	15%	41%	79%
backend developer	**18%**	**29%**	57%	83%
network administrator	4%	7%	49%	79%
solution architect	4%	10%	39%	80%
frontend developer	10%	18%	53%	77%
sql developer	9%	16%	**58%**	79%
android developer	**13%**	21%	52%	**71%**
product manager	**3%**	7%	**28%**	73%
web developer	7%	18%	53%	75%
scrum master	**3%**	7%	41%	79%
project manager	3%	10%	**29%**	75%
php developer	7%	12%	52%	81%
java developer	7%	15%	47%	**84%**
full stack developer	6%	**6%**	55%	**93%**
data analyst	5%	12%	**34%**	**71%**
Average	**7%**	**14%**	**47%**	**80%**

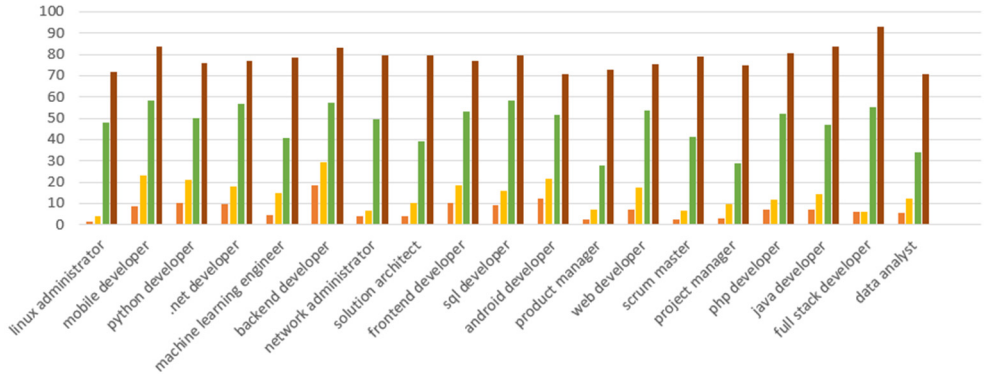


Figure 21. Success rate of finding job requirements for job positions by individual approaches.

4.1. Comparison of Found Job Requirements with the Open Skills Database

Part of the system verification included a comparison of found job requirements for individual positions with the Open Skills database. We worked with the outputs from the machine learning with improved marking lists approach for all job positions. Open Skills is an open database of job positions and skills [29].

An advantage of this database is the Open Skills API (http://api.dataatwork.org/v1/spec/—accessed on 31 August 2021), which enables us to work with job positions data; in our case, it was mainly the skills database.

Examples of several skills in the Open Skills database are presented in Table 7.

Table 7. Examples of skills in the Open Skills database.

Name of the Skill
angle dividers
angled pliers
appraisal software
approach detection systems
arbutus analyzer
archery bows
computer touch screens
lint filters
linux
linux-based email software
lion edge technologies ranch manager software
loan application processing software
regression testing software
relative humidity gauges
rigid ureteroscopes
skin staplers

The table reveals that it concerns general skills that can occur in any job position in any field. An advantage is the fact that it concerns quite a complex database of various job skills. Based on our experience, job requirements are often described in complex sentences. A job requirement might be detected correctly, although there is no match with skills in the Open Skills database. Another disadvantage is that the Open Skills database contains only a lower number of relevant IT skills that really occur on the verified job advertisements.

For illustrative purposes, we will present a job advertisement for the position Linux Systems Network Administrator with Polygraph, where our system validly marked 11 job requirements. However, only one of these job requirements matched with skills from Open Skills. This concerned the first bullet in the list containing a skill "linux" in the Open Skills database. The job advertisement with the marked job requirements is shown in Figure 22.

The Level 1 Linux System/Network Administrator (SNA) shall possess the following capabilities:

- Provide Linux and Solaris Systems Administration to local and remote servers
- Familiarization with VMware
- Experience generating and maintaining server PKI
- Experience Scripting (PowerShell, bash, etc.)
- Provide support for implementation, troubleshooting and maintenance of IT systems
- Manage the daily activities of configuration and operation of IT systems
- Provide assistance to users in accessing and using IT systems
- Provide support to IT systems including day-to-day operations, monitoring and problem resolution for all of the client/server/storage/network devices, mobile devices, etc.
- Provide Tier 1 (Help Desk) and Tier 2 (Escalation) problem identification, diagnosis and resolution
- Provide support for the escalation and communication of status to agency management and internal customers
- Optimize system operations and resource utilization, and perform system capacity analysis and planning

Qualifications: (U) Five (5) years' experience as a SA in programs and contracts of similar scope, type, and complexity is required. Bachelor's degree in a technical discipline from an accredited college or university is required. Five (5) years of additional SA experience may be substituted for a bachelor's degree.

Contractor shall currently possess and maintain one of the following certifications:

Figure 22. A job advertisement for a position Linux Systems Network Administrator with Polygraph with 0% match with the Open Skills database.

The level of the match with the Open Skills database is rather affected by the way of formulating job requirements in individual job advertisements. An important factor is a lack of a large number of typical job skills for IT positions in the Open Skills database.

A comparison of the found job requirements in job advertisements with job skills in the Open Skills database was carried out in the following steps:

- Load all job advertisements with all job requirements found in the lists;
- Load skills from the Open Skills database;
- Process all found job requirements and search for a match with at least one job skill from Open Skills;
- Mark job requirements with a match with Open Skills as relevant;
- Process the results.

This process is graphically depicted in Figure 23.

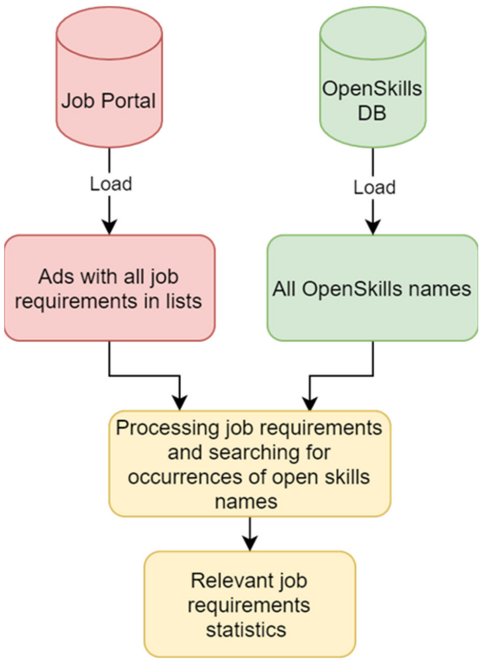


Figure 23. Visualization of the process of comparing the found job requirements with the Open Skills database.

The results include the ratio of the found matches of the analyzed job requirements with Open Skills for individual job positions (see Table 8).

Table 8. The ratio of the found relevant skills in Open Skills.

Name of the Job Position	Ratio of Relevant Skills in Open Skills
linux administrator	34%
mobile developer	44%
python developer	47%
.net developer	43%
machine learning engineer	45%
backend developer	47%
network administrator	32%
solution architect	30%
frontend developer	50%
sql developer	37%
android developer	58%
product manager	37%
web developer	52%
scrum master	28%
project manager	26%
php developer	37%
java developer	33%
full stack developer	22%
data analyst	35%
Average	39%

On average, 39% of relevant job skills in job requirements were found. This ratio of relevant skills is not very high, yet it is also caused by the fact that the Open Skills database does not contain some typical IT skills. Some examples can be: software development, computer science, security, and, also, names of technologies (Angular, ASP, .NET, HTML, CSS, etc.).

4.2. Most Frequent Skills

Another part of the research was to find the most frequent skills. First, we added the Open Skills database with other skills, primarily phrases occurring in IT job advertisements, names of technologies, and others. Here are examples of such phrases (skills):

- Software development;
- Computer science;
- Communication skills;
- Security;
- API;
- Angular;
- ASP;
- .NET;
- C#, etc.

Next, we performed a computation to find the top 20 most frequent skills (key phrases connected with skills) in all processed job advertisement across all job positions. The results are depicted in Table 9.

The list of the most frequent skills usually contains hard skills; only one skill can be marked as a soft skill—"communication skills". The most frequent skills include a large number of preferred technologies and name of programing languages (Java, JavaScript, HTML, Python, Angular, C#, Oracle, and others). The list of the most frequent skills is beneficial for HR managers or heads of IT departments in various companies.

Table 9. Top 20 most frequent skills in the analyzed job advertisements.

Skill	Number of Occurrences in Job Advertisements
Agile	3519
Java	3501
SQL	3050
Computer science	2549
Framework	2532
Communication skills	2500
API	2399
Database	2142
JavaScript	1901
Programming	1785
Certification	1776
Software development	1763
Analysis	1693
HTML	1352
Python	1271
Angular	1156
C#	1067
Security	1042
Oracle	831
Linux	466

4.3. Assessment of Relevant Job Requirements

We also performed the assessment of relevant job requirements in job advertisements where all job requirements in the lists were successfully found. The objective was to determine how many marked job requirements are relevant (i.e., they are real job requirements) and irrelevant (the marked text does not contain a job requirement). In some job advertisements, the system not only marked job requirements, but also incorrectly marked other texts as job requirements.

Examples of such irrelevant job requirements:

- Company name;
- Salary;
- Employer's contact information;
- Other job advertisement text that does not contain job requirements.

For each job position, a set of the following steps was performed in order to assess relevant job requirements:

- Loading all job advertisements where all job requirements were found on the lists;
- Loading a sample of random job advertisements;
- Loading all marked job requirements in the advertisement;
- Detecting relevant and irrelevant job requirements;
- Calculating the precision and recall metrics.

To assess the quality of performed marked job requirements in the system, standard metrics were used: precision and recall.

Precision defines the ability of the system to propose content that is relevant for a given user. It concerns a ratio of relevant recommendations with respect to all recommendations for the user. Precision can be calculated using the following:

$$\text{Precision} = \text{Correctly recommended content} / \text{Total recommended content}$$

where correctly recommended content is the number of relevant recommendations marked by the user as "correctly recommended". Total recommended content is the number of all recommendations provided to the user.

Recall defines the ability of the system to provide the user with relevant content. It concerns the number of correct recommendations in a set of relevant recommendations, i.e., top recommendations of the system. Recall can be calculated using the following:

$$Recall = Correctly\ recommended\ content / Relevant\ content$$

where correctly recommended content is the number of recommendations marked as "correctly recommended". Relevant content is a set of top recommendations based on user recommendations.

Precision signifies the ratio of job requirements marked as relevant to all marked job requirements. Recall signifies the ratio of job requirements marked as relevant to the list of the top 10 marked job requirements.

The results of assessing relevant job requirements (shown in Table 10) are promising. On average, 93% of job requirements were marked as relevant, so the precision is 93%. The recall is 91%. In most cases, all job requirements were correctly marked. The result was similar to those in Figure 14 or Figure 16. It is possible to claim that our proposed system can reliably mark job requirements in job advertisements.

Table 10. The precision and recall metrics in a random sample of job advertisements.

Name of the Job Position	Precision	Recall
linux administrator	89%	90%
mobile developer	95%	90%
python developer	93%	91%
.net developer	95%	93%
machine learning engineer	91%	88%
backend developer	92%	89%
network administrator	90%	87%
solution architect	94%	91%
frontend developer	91%	88%
sql developer	95%	93%
android developer	91%	90%
product manager	96%	94%
web developer	94%	91%
scrum master	97%	95%
project manager	97%	95%
php developer	89%	88%
java developer	93%	92%
full stack developer	93%	91%
data analyst	89%	91%
Average	**93%**	**91%**

5. Conclusions

This article dealt with a proposal of an approach to mine data from job advertisements on job portals. It mainly concerns mining job requirements from individual job advertisements. Our proposed system consists of a data mining module, machine learning module, and postprocessing module. The machine learning module is based on the SDCA logistic regression method. The postprocessing module contains several approaches to increase the success rate of marking job requirements. The objective of these approaches is to correctly detect and mark all job requirements in a given job advertisement. The success rate of these approaches is expressed by the ratio of job advertisements with all job requirements found for a given job position. We also performed a comparison of the found job requirements with the Open Skills database. Part of our proposed system is also the search for the most frequent job skills in the analyzed job positions. The list of the most frequented job skills shows the most frequently required job skills in IT job positions. We also verified correctly marked (relevant) and incorrectly marked (irrelevant) job requirements on a random sample of job advertisements.

In this article, several postprocessing approaches to improve the overall precision were developed and verified (basic machine learning, machine learning with marking of sentences between positive occurrences, machine learning with marking lists, machine learning with improved marking lists). The first two methods have limitations in correctly marking the job requirements, so the overall precision is low. The third method marks all job requirements in a list if 50% of the specific list is marked as job requirements. Using this method, the results are much better. The last method uses marks all job requirements in a list if at least one list item is marked as a job requirement. Experimental results show that this approach functions properly and correctly marks job requirements in lists. Moreover, this method has the highest overall precision.

Our proposed system was validated and verified on a set of selected IT job positions and their job advertisements. The results bring several practical implications:

- The system works with the whole text of the given job advertisement and uses the data mining module to detect the advertisement's structured text and mark lists of items containing job requirements.
- The system contains several approaches to detect and mark job requirements in job advertisements. They can also be used to detect other important information in the text (salary, benefits, employer's information, etc.), or even in other types of texts. The system would require the following modifications:
    - Creation of a training set for binary classification based on specific data which are to be found in the texts.
    - Train the model based on the training set.
    - Modification of the work with HTML structure elements based on the form of the searched information (bullets, tables, paragraphs, etc.).
- The verification of our system and the best approach, called machine learning with improved marking lists, bring promising results—an average of 80% of all job requirements found in the analyzed job advertisements.
- Certain job skills from the Open Skills database were found in 39% of the job advertisements with found job requirements.
- A list of the most frequent job skills in job advertisements was created. It was created based on the Open Skills database and added with other job skills typical for IT positions.
- The precision metric is 93% on average.

Nevertheless, we also encountered several limitations and situations that we had to resolve:

- Incorrect marking of some parts as job requirements in job advertisements:
    - This problem directly relates to the quality and size of the training set (model, respectively). A smaller tested dataset was supplied with representative examples.
- Detection of lists—some are as <li>, others as a point, etc.:
    - Advertisement lists were not often created using HTML tags <ul> and <li>, but by manually inserted points and other symbols for bullets and by manually inserted lines
 or <p>. That is why such lines had to be treated as regular bullets.

The proposed system for mining data from job advertisements and marking job requirements can be used not only for IT positions, but for various types. The only requirement is a modification of the training set and its adding with other types of job positions. In addition, the data mining module can be used for various job portals.

In the future, we would like to deal with several areas.

The first area is job skill extraction from job requirements in job advertisements. In the case of efficient extraction of the most frequent job skills from job requirements, it is possible to process these job skills further on. It is primarily suitable to classify such

job skills (hard skills, soft skills, life skills, qualification, experience). Another possibility is to cluster the detected hard skills. In the case of IT job positions, this might concern programing languages, development environments, frameworks, software tools, etc.).

The second area relates to job skills extraction. It concerns the development of a profile of a typical applicant for a given position based on data from job advertisements for this job position. This profile would include all must-have job skills, together with the necessary qualification. An applicant for a given job position would use such a profile to verify necessary job requirements. An HR manager and company management would use the profile when creating various job advertisements.

The third area is job matching. It concerns the automated comparison of an applicant's skills (based on CV data) with all job advertisements across various job positions. The system would then propose a list of the most suitable job advertisements with respect to the applicant's job skills. An HR manager would use this system to identify the most suitable applicant for the given job position (based on applicants' CVs).

In addition, we would like to verify the system on other types of job advertisements (e.g., in the area of business and marketing, HR management, etc.).

Author Contributions: Conceptualization, B.W.; methodology, B.W. and O.P.; software, O.P.; validation, B.W. and O.P.; formal analysis, B.W.; investigation, O.P.; resources, B.W. and O.P.; data curation, B.W. and O.P.; writing—original draft preparation, B.W. and O.P.; writing—review and editing, B.W. and O.P.; visualization, O.P.; supervision, B.W.; project administration, B.W.; funding acquisition B.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was created during the completion of a Student Grant SGS20/PřF-MF/2021 with student participation, supported by the Czech Ministry of Education, Youth and Sports.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Campion, M.A.; Fink, A.A.; Ruggeberg, B.J.; Carr, L.; Phillips, G.M.; Odman, R.B. Doing Competencies Well: Best Practices in Competency Modeling. *Pers. Psychol.* **2011**, *64*, 225–262. [CrossRef]
2. García-Barriocanal, E.; Sicilia, M.-A.; Sánchez-Alonso, S. Computing with competencies: Modelling organizational capacities. *Expert Syst. Appl.* **2012**, *39*, 12310–12318. [CrossRef]
3. Harzallah, M.; Vernadat, F. IT-based competency modeling and management: From theory to practice in enterprise engineering and operations. *Comput. Ind.* **2002**, *48*, 157–179. [CrossRef]
4. Shahhosseini, V.; Sebt, M. Competency-based selection and assignment of human resources to construction projects. *Sci. Iran.* **2011**, *18*, 163–180. [CrossRef]
5. Shippmann, J.S.; Ash, R.A.; Batjtsta, M.; Carr, L.; Eyde, L.D.; Hesketh, B.; Sanchez, J.I. The practice of competency modeling. *Pers. Psychol.* **2000**, *53*, 703–740. [CrossRef]
6. Fleishman, E.A.; Reilly, M.E. *Handbook of Human Abilities: Definitions, Measurements, and Job Task Requirements*; Consulting Psychologists Press: Palo Alto, CA, USA, 1992.
7. Lee, C.K.; Han, H.J. Analysis of skills requirement for entry-level programmer/analysts in Fortune 500 corporations. *J. Inf. Syst. Educ.* **2008**, *19*, 17.
8. Sibarani, E.M.; Scerri, S.; Morales, C.; Auer, S.; Collarana, D. Ontology-guided job market demand analysis: A cross-sectional study for the data science field. In Proceedings of the 13th International Conference on Semantic Systems, Amsterdam, The Netherlands, 11–14 September 2017; pp. 25–32.
9. Litecky, C.; Aken, A.; Ahmad, A.; Nelson, H.J. Mining for Computing Jobs. *IEEE Softw.* **2009**, *27*, 78–85. [CrossRef]
10. Gardiner, A.; Aasheim, C.; Rutner, P.; Williams, S. Skill requirements in big data: A content analysis of job advertisements. *J. Comput. Inf. Syst.* **2018**, *58*, 374–384. [CrossRef]
11. Verma, A.; Yurov, K.M.; Lane, P.L.; Yurova, Y.V. An investigation of skill requirements for business and data analytics positions: A content analysis of job advertisements. *J. Educ. Bus.* **2019**, *94*, 243–250. [CrossRef]
12. Verma, A.; Lamsal, K.; Verma, P. An investigation of skill requirements in artificial intelligence and machine learning job advertisements. *Ind. High. Educ.* **2021**, *11*, 0950422221990990.

13. Zhang, Y.; Su, F.; Hubschman, B. A content analysis of job advertisements for digital humanities-related positions in academic libraries. *J. Acad. Libr.* **2020**, *47*, 102275. [CrossRef]
14. Mohri, M.; Rostamizadeh, A.; Talwalkar, A. *Foundations of Machine Learning*; MIT Press: Cambridge, MA, USA, 2018.
15. Shavlik, J.W.; Dietterich, T.; Dietterich, T.G. (Eds.) *Readings in Machine Learning*; Morgan Kaufmann: Burlington, MA, USA, 1990.
16. Boselli, R.; Cesarini, M.; Mercorio, F.; Mezzanzanica, M. Classifying online Job Advertisements through Machine Learning. *Futur. Gener. Comput. Syst.* **2018**, *86*, 319–328. [CrossRef]
17. Pejic-Bach, M.; Bertoncel, T.; Meško, M.; Krstić, Ž. Text mining of industry 4.0 job advertisements. *Int. J. Inf. Manag.* **2020**, *50*, 416–431. [CrossRef]
18. Djumalieva, J.; Lima, A.; Sleeman, C. *Classifying Occupations according to Their Skill Requirements in Job Advertisements*; Economic Statistics Centre of Excellence Discussion Paper; Economic Statistics Centre: London, UK, 2018.
19. Grüger, J.; Schneider, G.J. Automated Analysis of Job Requirements for Computer Scientists in Online Job Advertisements. In Proceedings of the 15th International Conference on Web Information Systems and Technologies (WEBIST 2019), Vienna, Austria, 18–20 September 2019; pp. 226–233.
20. Mewburn, I.; Grant, W.J.; Suominen, H.; Kizimchuk, S. A machine learning analysis of the non-academic employment opportunities for Ph.D. graduates in Australia. *High. Educ. Policy* **2020**, *33*, 799–813. [CrossRef]
21. Dawson, N.; Rizoiu, M.A.; Johnston, B.; Williams, M.A. Predicting skill shortages in labor markets: A machine learning approach. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), online, 10–13 December 2020; pp. 3052–3061.
22. Csiba, D.; Qu, Z.; Richtárik, P. Stochastic dual coordinate ascent with adaptive probabilities. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 674–683.
23. Shalev-Shwartz, S. SDCA without duality. *arXiv* **2015**, arXiv:1502.06177.
24. Vikström, A. A Comparison of Different Machine Learning Algorithms Applied to Hyperspectral Data Analysis. *DiVA* **2021**.
25. Shalev-Shwartz, S.; Zhang, T. Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization. *J. Mach. Learn. Res.* **2013**, *14*, 2.
26. Suzuki, T. Stochastic dual coordinate ascent with alternating direction method of multipliers. In Proceedings of the International Conference on Machine Learning Beijing, Beijing, China, 21–26 June 2014; pp. 736–744, PMLR.
27. Tran, K.; Hosseini, S.; Xiao, L.; Finley, T.; Bilenko, M. Scaling up stochastic dual coordinate ascent. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, Australia, 10–13 August 2015; pp. 1185–1194.
28. Yang, T. Trading Computation for Communication: Distributed Stochastic Dual Coordinate Ascent. *NIPS* **2013**, *27*, 629–637.
29. Crockett, T.; Lin, E.; Gee, M.; Sung, C. DATA AT WORK. Open Skills Project. 2018. Available online: http://dataatwork.org/data/ (accessed on 3 July 2021).