A System for Detecting Professional Skills from Resumes Written in Natural Language

Emil St. Chifu, Viorica Rozina Chifu, Iulia Popa, Ioan Salomie
Department of Computer Science
Technical University of cluj-Napoca
Cluj-Napoca, Romania
{emil.chifu, viorica.chifu, iona.salomie}@cs.utcluj.ro

***Abstract*—In this paper, we present a new method for detecting professional skills (as noun phrases) from resumes written in natural language. The proposed method uses an ontology of skills, the Wikipedia encyclopedia, and a set of standard multi word part-of-speech patterns in order to detect the professional skills. First, the method checks to see if there are, in the text of the resumes, skills that are concepts in our ontology. The method also tries to identify possible new skills, which are not present in our ontology. This is done with the help of some specific, lexicalized, multi-word expression patterns (i.e. specific contexts) that could surround new, unknown skills. The specific expression patterns (specific contexts) are induced by training from a corpus of resumes. This induction of the possible specific contexts for new skills is based on a set of standard, generic part-of-speech patterns (found by hand) that usually contain the skills already present in the ontology. Hence our skill extraction method is based on a bootstrapping approach. The newly detected skills are validated by a human expert and then inserted automatically into the skill ontology. Populating the ontology with the new skills is performed with the help of the Wikipedia encyclopedia. The method proposed has been tested on a set of resumes written by users as well as on a corpus collected by automatically extracting resumes from specific Web sites.**

I. INTRODUCTION

Usually each user builds his/her CV in his/her own way. So, an application that is able to extract the professional skills from a resume written in natural language is useful for the online job portals or the recruiter workforce agencies. Such an application should provide the extracted information in a structured way in order to help the specialized portals or agencies when filtering the candidates to meet the criteria for a specific job.

There are different approaches that can be applied when extracting professional skills from resumes written in natural language. All the possible approaches try to extract specific patterns or words from text. The approaches can be named entity based, rule based, statistical, machine learning based, or hybrid ones [6]. The named entity based approaches use regular expressions or dictionaries for extracting specific patterns or words from text. The statistical approaches use statistical models such as Hidden Markov Models or Conditional Random Fields, while in the rule based approaches, grammars help identify specific patterns or words in the text. The learning based approaches use AI classification algorithms, such as decision trees, support-vector machines (SVMs), maximum entropy, or transformation-based learning, while the hybrid approaches combine some of the previously mentioned approaches when extracting specific linguistic patterns from text [6].

In this paper we present an ontology based system capable of detecting professional skills (as noun phrases) from CVs or resumes written in natural language. Our system uses a domain ontology of skills with more than 13,000 concepts, but an important aspect of our work is the ability to detect even new skills, i.e. skills that are not present in the ontology. In order to reach this goal, we have considered several aspects. First of all, we needed a varied data set to train the system. This data set is a large corpus consisting of CV summaries, especially in the field of computer science. Secondly, by using the information extant in the skill ontology[1], the diverse corpus of data, and a set of standard, generic part-of-speech patterns (found by hand) which usually contain the skills already defined in the ontology, we induce by training the possible specific contexts in which new skills could be mentioned in the text. Hence our skill extraction method is based on a bootstrapping approach. More concretely, we generate the possible specific multi word expression patterns that could precede a new competence in the text. Only some domain specific words that occur in a lexicalized multi-word expression pattern – and which define a specific context – will actually help locate a new skill and no other semantic entities. So, by having these specific contexts, our system will be able to detect possible new skills, not defined in our skill ontology. Moreover, these newly discovered skills are then inserted into the ontology. For the skill insertion process an issue occurs, i.e., finding the right place in our ontology for the new skill. To solve this problem, we have a matching algorithm for finding the right taxonomic parent for the new skill.

The paper is structured as follows. Section 2 presents related work in the field of extracting relevant information from text. Section 3 presents the architecture of our system as well as the algorithms involved, while section 4 shows the experimental results together with the information retrieval metrics used for validating the performance of our system. We end our paper with some conclusions.

[1] i.e. by knowing the skills which are already defined as concepts in the skill ontolgy

978-1-5386-3368-7/17/$31.00 © 2017 IEEE

II. RELATED WORK

This section presents some of the existing approaches in the field of extracting relevant information from text.

In [1] the authors propose a tool, Screener, that is able to extract information from a CV summary of the job seekers. The information to extract from a CV summary is manifold: skills, experience, education etc. The method integrated into the Screener tool analyzes a large corpus of data. By using a set of pattern rules, it identifies four sections of the candidate profile – i.e. personal information, education, skills, experience or projects – from which the relevant information is further extracted.

The approach in [2] uses named entity recognition (NER) and named entity normalization (NEN) for the extraction of relevant information from CVs. The authors consider some aspects to take into account when detecting and extracting information from CV abstracts. For instance, the same entity could be expressed with different tokens (e.g. C# or C sharp). Another feature that could generate ambiguity is the same token defining multiple notions (e.g. Java in Java coffee and Java programming language). NER concerns with identifying the phrase which expresses a skill, whereas NEN goes further and links that phrase to an actual individual or entity. The authors define an application called SKILL, which detects, extracts, and links a skill to a qualified entity. The system is composed of two main parts: skill taxonomy generation and skill tagging. The taxonomy generation consists of several steps. In the first step, the text containing the skills is gathered and divided (i.e. the phrases are split by headers and punctuations). Then the second step eliminates the noise by using a dictionary containing stop words, adverbs, adjectives, names, and other phrases. Then the taxonomy is built with the help of the category tags obtained by using the Wikipedia API. The skill tagging matches the skills found in CVs with the concepts of the above generated skill taxonomy. The automation of this task is done by using MediaWiki and rules established on category tags.

In [3] the authors present a hybrid approach for extracting relevant information from a curriculum vitae corpus. The proposed approach combines knowledge-based methods with machine learning in order to improve the performance of the classical methods. The novelty of this approach consists in the concept of multi-dimensional space that guides the users in selecting the appropriate hybrid concepts.

The paper [4] presents an approach for extracting special skills from resumes. In this approach the authors consider that the skill information in a resume is organized in skill types and skill values. They apply a method that consists of three main steps for extracting the skills. The first step consists in identifying the features from the skills section. The second step calculates the value of the degree of specialness for the skill type features and then organizes the special skill type features, while the third step calculates the value of the degree of specialness for the skill value features and then organizes the special skill value features. The first step actually identifies the skill type features as well as the skill value features. The second step actually begins by computing the degree of specialness value for the skill type features, and then it organizes the skill type features by using a three-level organization approach. The third step begins by computing the degree of specialness value for the skill value features, and then it organizes the skill value features by using a three-level organization approach.

In [5] the authors propose an approach for extracting relevant information from resumes that is based on a cascaded hybrid model. The approach consists of two main steps. In the first step, the resume is segmented into consecutive blocks according to some specific labels by using a Hidden Markov Model (HMM). In the second step, the educational and personal information is extracted by using a Hidden Markov model and Support Vector Machines. Based on the experimental results, the authors demonstrate that this hybrid model provides better results than the flat models.

III. ARCHITECTURE OF THE SYSTEM

This section presents the architecture of our system, which consists of the following modules: the crawler module, the specific pattern induction module, the skill detection module, and the ontology update module (see Figure 1).

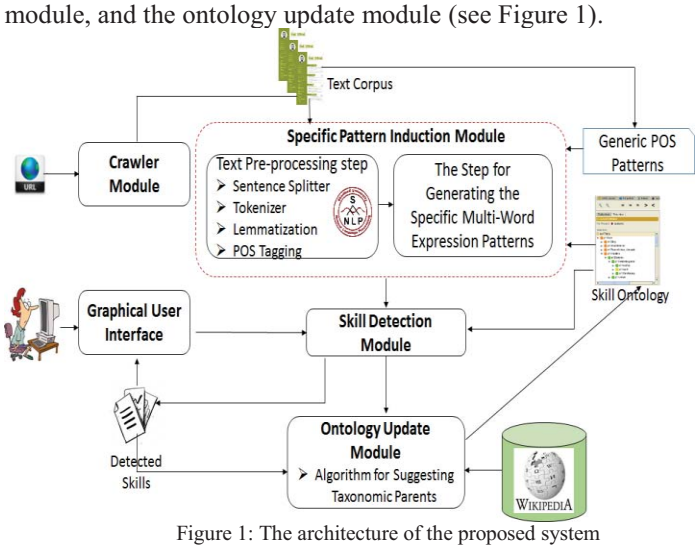


Figure 1: The architecture of the proposed system

A. The Crawler Module

This module visits specific Web sites – sites that expose resumes to recruiters –, reads the pages, and extracts a large number of CV resumes in order to create a data corpus. This corpus will be used as training and testing data by our approach for detecting professional skills from text.

We have initially a URL to a specific website, in which we search for resumes of candidates and we get a result list. This list represents the starting point. The crawler connects to the URL by using JSoup, a Java framework used for parsing HTML pages, which is capable of extracting and managing html data. In our approach, we have collected pages from two websites [7, 8]. For the pages collected, we have applied two rules for extracting the text corresponding to resumes of users: one rule for parsing the resumes found in the currently processed page and another rule for identifying the buttons that contain hyperlinks to other resumes. This hierarchy allows us to download multiple pages from a search result page, and consequently, to crawl through the website. It is important to say that both the start URL and the two rules are specific to each site. So, for each resume link that we found, we download the html code and then we apply the two specific

rules for the website of that link in order to extract the description of the resumes.

The resumes written in html format are then pre-processed in order to transform them into plain text. This involves the removal from the resumes of all the html tags or flags formatting the text. This step is very important as we want the text to be as smooth as possible. Each of the extracted resumes will be part of our corpus that will be used for automatically training, testing, and validating our system. More specifically, the corpus of resumes will help us test and improve (by training) the quality of our system. The improvement consists in adding more expression patterns that would be capable of detecting professional skills from text.

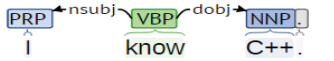
B. The Specific Pattern Induction Module

This module takes as input the corpus of resumes obtained by the previous module in the pipeline (the crawler module, see Section III-A and Figure 1), the skill ontology, and a set of standard, generic part-of-speech patterns (found by hand). The module returns a set of lexicalized (specific) multi word expression patterns, as induced by training on the corpus of resumes. These specific linguistic patterns will be used by the next module in the pipeline (the skill detection module, see Section III-C and Figure 1) in order to detect new skills. There are two steps in the process of inducing the specific expression patterns: (*i*) text preprocessing and (*ii*) generating the specific multi-word expression patterns.

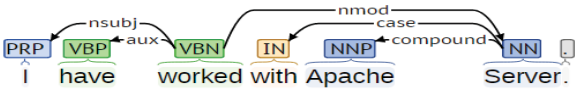
The Text Preprocessing Step. This step consists in performing sentence splitting, tokenization, lemmatization, and part of speech tagging on our corpus of resumes. The sentence splitting process identifies the sentence boundaries in the resumes and separates all the sentences by displaying them one per line. The tokenization means dividing each sentence into linguistic units (i.e. tokens). The lemmatization (by using morphological analysis) is the process of reducing each inflectional word form to its stem in canonical form. The part of speech tagging consists in annotating every word in the text corpus of resumes with information about the part of speech (POS). Since the skills to be extracted from the resumes are usually noun phrases, the words should indeed be annotated with part of speech. For all the preprocessing steps enumerated here we have used the Natural Language Processing software provided by the Stanford NLP Group [9].

The Step for Generating the Specific Multi-Word Expression Patterns. Our domain ontology of skills contains a large number of concepts. But there are cases in which a word or a phrase should be considered as a competence and it is not defined as a concept in our skill ontology. In this step, we generate the specific multi word expression patterns (i.e. the specific, lexicalized contexts) in which new, unknown skills, that are not concepts in our skill ontology, could be encountered in the text. We induce the specific multi word expression patterns as groups of words that could be situated in the text near any of the new skills.

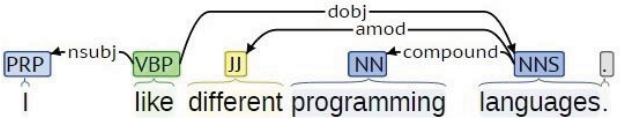
By analyzing our corpus of resumes, we noticed that CVs use a methodical, standard language, and moreover, usually the skills are preceded by certain patterns easy to detect by hand. In order to actually generate the lexicalized (specific) expression patterns able to extract new skills, we analyze each sequence of words positioned in the text of the corpus before a known skill, i.e. before an instance of a concept in the skill ontology, like in a bootstrapping approach. We noticed that, in a sentence, a skill is usually preceded by one of the following ten generic part-of-speech (POS) patterns: verb; verb and preposition; verb and adjective; verb and article; noun and preposition; adjective and preposition; noun and ':'; noun and '('; adjective, preposition, and article; noun, preposition, and article (see Figure 2). As a consequence of this analysis, we assume a set of ten standard POS patterns as enumerated above – as found by hand, as predefined – to be the basis for the induction[2] of the corpus-specific, lexicalized, multi word expression patterns. These lexicalized patterns to be induced/ generated will be similar to (matched on) the given standard POS patterns, though more specialized on resumes (i.e. they will be resume corpus specific, lexicalized from the resume corpus). Consequently, we expect the specific multi-word expression patterns generated/ induced here will be able to locate new, unknown skills in resumes. These expectations are confirmed by the experimental results in Section IV.



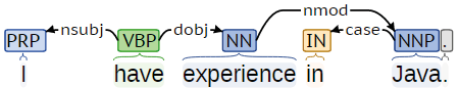
The skill C++ is preceded by a verb.



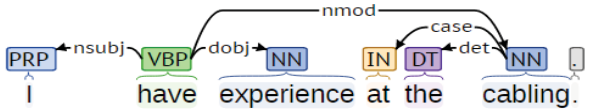
Apache server is preceded by a verb plus preposition.



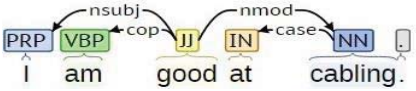
The construction formed by a verb plus adjective comes before the skill.



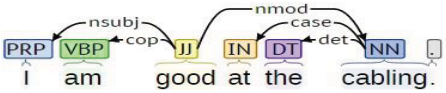
Java follows after a noun and a preposition.



The skill comes after a noun, preposition, and article.



The aptitude follows after an adjective and a preposition.



The aptitude comes after adjective, preposition, and article.

Figure 2: Examples of standard POS patterns with corresponding linguistic realizations

[2] induction from the corpus (by training). Each specific multi-word expression pattern is induced/ generated when it is first recognized in the text of the corpus by being matched on one of the ten standard part-of-speech patterns and at the same time when it precedes a known skill in the text of the corpus (see Algorithm 1).

Moreover, it is obvious that the set of ten standard, generic POS patterns have to be specialized towards the domain of resumes (by training/ induction on the domain specific corpus of resumes), since not any combination of any possible words merely conforming to one of the given standard POS pattern will indeed help retrieve an unknown skill. Only some domain specific words that occur in a lexicalized multi-word expression pattern – and which define a specific context – will actually help locate a new skill and no other semantic entities. For instance, the sentences "I have experience with the C++ language" and "I have doubts about the future salary growth" both match the standard POS pattern "noun + preposition + article". Yet the generic POS pattern will actually detect "the C++ language" as an unknown skill, (i.e. as a true positive) in the first sentence, whereas the same generic pattern will identify "the future salary growth" wrongly as a unknown skill (i.e. as a false positive). Based on the predefined set of ten generic POS patterns that could possibly precede any skill in the text of the resumes, we have developed a training algorithm for inducing the specific, lexicalized, patterns (i.e. the specific contexts) that would be able to locate new, unknown skills in resumes (see Algorithm 1).

Algorithm 1: Induction Algorithm for Generating the Specific, Lexicalized Contexts

---

Inputs: *resume* - a resume in the corpus; *skillSet* - set of skills defined in the skill ontology; set of ten standard POS patterns
Ouput: *ExPattern* – Lexicalized multi-word pattern preceding a known *skill* (i.e. preceding a *skill* **in** *skillSet*)

```
begin
  foreach sentence in resume
   foreach skill in skillSet
     ExPattern = ExtractFragmentBeforeSkill(sentence, skill)
     if(ExPattern ≠ {})
       if(ExPattern.noWords > 3)   // The threshold value is 3.
         ExPattern = ExtractLastThreeWords(ExPattern)
       endif
       aExPattern = AnnotateWithPartOfSpeech(ExPattern)
       switch(aExPattern)
         case 'any-part-of-speech any-part-of-speech verb' :
         case 'any-part-of-speech verb preposition' :
         case 'any-part-of-speech verb adjective' :
         case 'any-part-of-speech verb article' :
         case 'any-part-of-speech noun preposition' :
         case 'any-part-of-speech adjective preposition' :
         case 'any-part-of-speech noun :' :
         case 'any-part-of-speech noun (' :
         case 'adjective preposition article' :
         case 'noun preposition article' :
           return ExPattern
         default:
           return null
       endswitch
     endif
   endfor
  endfor
end
```

The algorithm consists of the following steps:

1) Each skill in the ontology is checked as whether it occurs in the different sentences of the resume. (One current resume in the corpus is an input for the algorithm.)
2) The text fragment positioned before the competence (i.e. preceding the skill in the text) is extracted.
3) At most the last 3 words in the text fragment are selected and the rest is ignored. This comes as a solution to the fact that the antecedent fragment could be very large. As a consequence, we use a threshold value for selecting the tokens preceding the skill. In this case the threshold value is 3.
4) This step consists in a part of speech tagging of the words extracted. We check what part of speech is every word in the extracted sequence. If the expression representing the extracted sequence matches one of the patterns in the predefined set of standard POS patterns then it will be retrieved to become one of the lexicalized (specific) multi-word expression patterns induced/ generated.

Several examples on how the algorithm works are shown in Table 1. By using all these specific sequences of words, or just a specific word in some cases – in short, by using the lexicalized patterns –, we are able to detect possible skills that are not in the skill ontology. Table 1 illustrates examples of lexicalized patterns that are induced from our corpus.

Table 1: Examples of lexicalized patterns as induced from our corpus

Sentence	**Skill**	**Standard POS pattern matched**	**Lexicalized pattern induced**
I am good at the cabling	Cabling	Adjective + preposition + article	Good at the
I have experience at the cabling	Cabling	Noun + preposition + article	Experience at the
I have experience in C++	C++	Noun + preposition	Experience in
I have experience with SAP	SAP	Noun + preposition	Experience with
I have implemented Java applications	Java	Verb	Implemented
I write in Java	Java	Verb + preposition	Write in
I have worked with Apache Server	Apache Server	Verb + preposition	Worked with
I like different programming languages	Programming languages	Verb + adjective	Like different
I have good PHP knowledge.	PHP	Verb + adjective	Have good
Managing the database	Database	Verb + article	Managing the
The skills: Java, C++	Java	Noun + ':'	Skills:
I know the following foreign languages: Spanish, French.	Spanish	Noun + ':'	Languages:
My skills (cabling, programming)	Cabling	Noun + '('	Skills (

C. The Skill Detection Module

The main idea of this module is to identify and extract skills from a resume that are concepts in our skill ontology and to also detect possible skills that are not defined as concepts in the skill ontology. The skill detection process consists of the following steps that are performed for each sentence in the resume:

- We search the sentence for skills extant in the ontology.
- After identifying all the instances of skill ontology concepts in the sentence, we search for lexicalized patterns. We consider each of the specific context defining lexicalized patterns generated by the previous module in the pipeline (i.e. the specific pattern induction module, see Figure 1 and Section III-B with Algorithm 1). So, the sentence is searched by matching any such specific pattern. The process of finding the multi word sequences in the sentence that match any of the specific, lexicalized patterns is driven by actually matching word sequences against the lexicalized pattern represented as a regular expression.
- For any match of a lexicalized pattern in the sentence, the current step extracts the noun phrases that are placed in the text after the sequence of words found (matched). This process of noun phrase retrieval involves using the Stanford natural language processing framework [9].
- Every noun phrase extracted is then verified to see whether it is contained, as a concept, in the ontology. If it is not, this means the noun phrase is a possible new skill.

D. The Ontology Update Module

The goal of this module is to enrich the domain ontology of skills with newly added skills. When inserting a new skill into the ontology, the system suggests a set of possible taxonomic parents for the skill, i.e. the possible appropriate places in the ontology where to insert the skill. For this task of suggesting taxonomic parents, there are several steps to be performed, as illustrated by Algorithm 2. The main idea of Algorithm 2 is to have as input a possible new skill and to return a set of possible candidate parents. First of all, – before applying Algorithm 2 – for every possible new skill, which is a noun phrase, we need a definition or a synonym. To fulfill this idea, we perform a search on **Wikipedia** and we keep the first definition of the new skill as returned by the search. Suppose the system detected Microsoft Word as a new skill. The definition returned by searching ”Microsoft Word” on **Wikipedia** is ”Microsoft Word is a word processor developed by Microsoft”. The definition thus retrieved is then processed in the first phase of Algorithm 2 in order to extract the relevant parts. We are mainly interested to extract the subject and the predicate of the definition, where the definition is seen as a sentence. More concretely, the first phase of Algorithm 2 extracts the subject ”Microsoft Word” and (the direct object of) the predicate ”word processor” from the definition retrieved by Wikipedia. This extraction is performed by using the Reverb natural language processing tool [10].

The second phase of Algorithm 2 consists in applying a matching process that is able to find the appropriate direct superclass (i.e. parent) for the new skill. This process considers the words which occur in the subject and the predicate that have been extracted in the first phase of Algorithm 2. The two phrases – i.e. the noun phrase and the verb phrase representing the subject and the predicate – are split into tokens, which are then added into a set of subject tokens and a set of predicate tokens. The linking words such as {*and*, *or*} and the articles {*a*, *an*, and *the*} are removed. In the next step, we query the skill ontology on every word in the two sets of (subject and predicate) tokens. The results of this search are stored in a map, where the key represents the ontology concept found by the search, and the value is the number of times the concept was found as a result of the query. The keys in the map which have a value greater than 1 are then extracted and added into a set of possible taxonomic parents for the new skill. We also add into the set of possible taxonomic parents the class (i.e. ontology concept) that is semantically the most generic among the query results when querying the ontology on the predicate tokens. We do this only for the predicate tokens since (the direct object of) the predicate in any Wikipedia definition indicates the taxonomic superclass, parent (i.e. the value of the *is-a* relation) of the currently defined concept – and we indeed are searching for a parent for the new skill concept.

Algorithm 2: Algorithm for Suggesting Taxonomic Parents

Inputs: *def* - the definition of the possible new skill
Ouput: *pParents* - the set of possible taxonomic parents for the new skill

```
begin
 subject = ExtractSubject(def)
 predicate = ExtractPredicate(def)
 sTokens = SplitSubjectPhraseIntoTokens(subject)
 pTokens = SplitPredicatePhraseIntoTokens(predicate)
 sTokens = RemoveStopWords(sTokens)
 pTokens = RemoveStopWords(pTokens)
 Map = {}
 foreach token in sTokens
   TokenMatchers = QueryOntology(token)
   foreach tokenMatcher in TokenMatchers
    Map = Map ∪ Put(tokenMatcher, numberOfTimes)
   endfor
 endfor
 foreach token in pTokens
  TokenMatchers = QueryOntology(token)
  if(token.isLast)
    pMatchers = pMatchers ∪ TokenMatchers
  endif
  foreach tokenMatcher in TokenMatchers
    Map = Map ∪ Put(tokenMatcher, numberOfTimes)
  endfor
 endfor
 bestMatchers = GetKeysWithValueGreaterThanOne(Map)
 pParents = bestMatchers ∪ getMostGenClass(pMatchers)
 return pParents
end
```

Moreover, the use of the selector *isLast* in Algorithm 2 means selecting only the last word – i.e. the head noun – of the noun phrase representing (the direct object of) the predicate in the Wikipedia definition. The reason is that the head noun of

any noun phrase is always more generic semantically than the noun phrase as a whole, according to the "vertical relation" (or "head matching") heuristic [11]. The modifiers included in the noun phrase always particularize the semantics of the whole noun phrase as compared with the semantics of the head noun taken alone. Yet why do we need now the ontology class that is semantically the most generic among the query results on the predicate tokens? The reason is that sometimes the concepts added into the set of possible taxonomic parents for the new skill in the first step (via the map, as described in the previous paragraph) may give no results (actually no concept), since there was no key in the map with value greater than 1, as the ontology were still too sparse. This way, we offer at least one fall-through, catchall, most generic suggestion of possible taxonomic parents in case the first step gave no results when building the set of passible parents.

At the end, this set of possible taxonomic parents becomes the final result of Algorithm 2. The set consists of several concepts in the ontology, each of which representing a suggestion as a possible candidate parent for the new skill.

For a better understanding of the process of suggesting taxonomic parents (Algorithm 2), consider the following example. Let "Apache Tomcat" be a new skill, hence not yet defined as concept in the skill ontology. It will be inserted as a new concept into the ontology. The definition returned by Wikipedia is "Apache Tomcat, often referred to as Tomcat, is an open-source web server developed by the Apache Software Foundation (ASF)". All the processing steps are displayed in Table 2.

Table 2: Example of how a Wikipedia definition is processed by Algorithm 2

Step of the algorithm	**Subject**	**(Direct Object of) Predicate**
Extract the subject/ predicate (S/ P) phrase	Apache Tomcat	an open-source web server
Split the S/ P phrase into words and add the words into the set of S/ P tokens	{ Apache, Tomcat }	{ an, open-source, web, server }
Remove the linking words from the set of S/ P tokens	{ Apache, Tomcat }	{ open-source, web, server }
Query the skill ontology on every word in the set of S/ P tokens and add the query results into a map <concept, numberOfTimes>	< Apache Server, 2 > < Novell-Server, 1 > < Samba-Server, 1 > < WebSphereApplicationServer, 2 > < Client-Server, 1> < Citrix Presentation Server, 1 > < special operating systems and server software, 1 > < Microsoft SQL Server, 1 > < multimedia and web design, 1 > < graphic web design, 1 > < web promotion, 1 > < multimedia and web authoring software, 1 > < accessible web design, 1 >	
Extract the keys in the map that have value > 1	{ Apache Server, WebSphereApplicationServer }	
Add the ontology class that is the most general among the query results when querying the ontology on predicate (P) tokens	{ special operating systems and server software, Apache Server, WebSphere Application Server }	

IV. EXPERIMENTAL RESULTS

In order to validate the performance of our system, we have first trained, tested, and validated our system on a set of 500 CV resumes extracted from two web sites, [7] and [8]. Then we have tested our system on a set of resumes written in natural language that are entered by the users. An example of a resume used in the initial testing of our system, which was collected from the Web site [8], is shown in Figure 3.

The domain ontology of skills that is used by our system consists of 13,337 concepts, which are split into two main categories: *domain specific skills and competences* and *non-domain specific skills and competences*, as well as relationships between them. A fragment of the skill ontology is illustrated in Figure 4.

A Contract Web Developer with over 9 years experience of developing websites for various business sectors and size of enterprise.

This candidate has experience of managing the complete development cycle, working to tight deadlines and budgets. Developing using HTML, CSS, Javascript and LAMP technologies. Having recently completed a short contract customising and modernising ORSEE for a large Edinburgh client, adding functionality and improving use ability. Improved UI/ UX, using the CSS framework and PDO for security against SQL injections.

This candidate is currently immediately available and looking for the next exciting opportunity, I look forward to discussing how this person can fit in to your team.

Figure 3. Resume example from website [8]

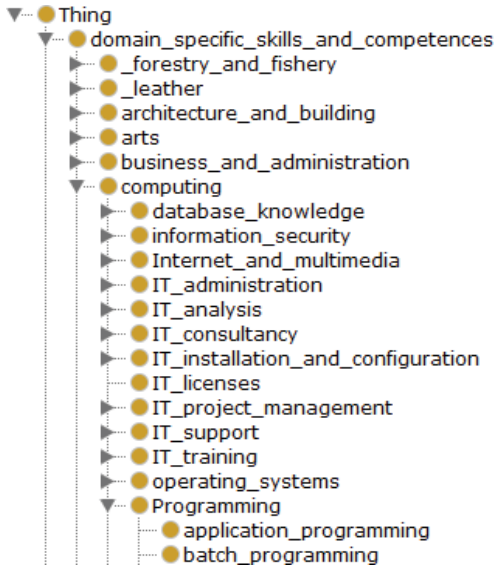


Figure 4. Fragment of the skill ontology

A. Validating the System on a Set of Resumes Collected from Different Web Sites

The performance of our system is measured by using the most known information retrieval metrics: *precision*, *recall*, and *Matthews correlation coefficient*. These measures are computed based on the values of the *true positives*, *true negatives*, *false positives*, and *false negatives*.

The *Matthews correlation coefficient* (*MCC*) [12] is a metric for classifying into two classes, which has proven to be balanced even if there is a significant difference between the sizes of the two classes. The metric correlates the predicted and observed categorization. It takes values in the interval [–1, 1], where –1 means a bad prediction, 0 classifies a random prediction, and 1 a perfect guess.

Table 3 presents a fragment of the experimental results obtained by our system when tested on the set of 500 resumes. We gathered this subset of results for a sample of 20 resumes. In Table 3, *noW* represents the total number of words, *noS* is the number of skills, while *noNonS* is the number of words that are not skills; the rows illustrate the results for the 20 resumes individually, while the last row gives the average precision, recall, and MCC values measured for these 20 resumes. From Table 3 we could notice that our system identifies more than half of the skills actually present in a resume.

Table 3: Sample of the experimental results obtained on a subset of 20 out of the total number of 500 resumes

TP	FN	FP	TN	noW	noS	noNonS	PREC	REC	MCC
27	9	13	1073	1122	36	1086	0,68	0,75	0,70
15	5	8	711	739	20	719	0,65	0,75	0,69
5	7	3	162	177	12	165	0,63	0,42	0,48
7	3	1	228	239	10	229	0,88	0,70	0,77
1	1	2	88	92	2	90	0,33	0,50	0,39
4	7	6	143	160	11	149	0,40	0,36	0,34
18	2	7	120	147	20	127	0,72	0,90	0,77
8	4	3	107	122	12	110	0,73	0,67	0,66
4	3	5	120	132	7	125	0,44	0,57	0,47
6	4	3	178	191	10	181	0,67	0,60	0,61
2	3	3	149	157	5	152	0,40	0,40	0,38
1	1	3	150	155	2	153	0,25	0,50	0,34
2	1	1	72	76	3	73	0,67	0,67	0,65
8	2	3	162	175	10	165	0,73	0,80	0,75
5	2	3	106	116	7	109	0,63	0,71	0,65
3	1	3	121	128	4	124	0,50	0,75	0,60
2	3	6	95	106	5	101	0,25	0,40	0,27
5	3	1	81	90	8	82	0,83	0,63	0,70
4	2	2	103	111	6	105	0,67	0,67	0,65
5	3	4	132	144	8	136	0,56	0,63	0,56
Average PREC, REC, and MCC							**0.58**	**0.62**	**0.57**

The charts in Figures 5, 6, and 7 illustrate the values of the metrics *precision*, *recall*, and *Matthews correlation coefficient* computed individually on each of the 500 resumes in the experimental data set. As seen on the charts in Figures 5 and 6, the precision and the recall of the system are good, but of course not perfect, since information extraction is not a trivial task. So, the system errors in terms of labeling a non-skill as skill and in terms of not detecting some skills are little.

The values obtained for the *Matthews correlation coefficient* (see Figure 7) show that the system offers a strong correlation between the expected output – in terms of correctly classifying a noun phrase as a skill versus as a non-skill – and the actual output. All the values obtained are between 0 and 1 (i.e. they are all positive), and this is a good result.

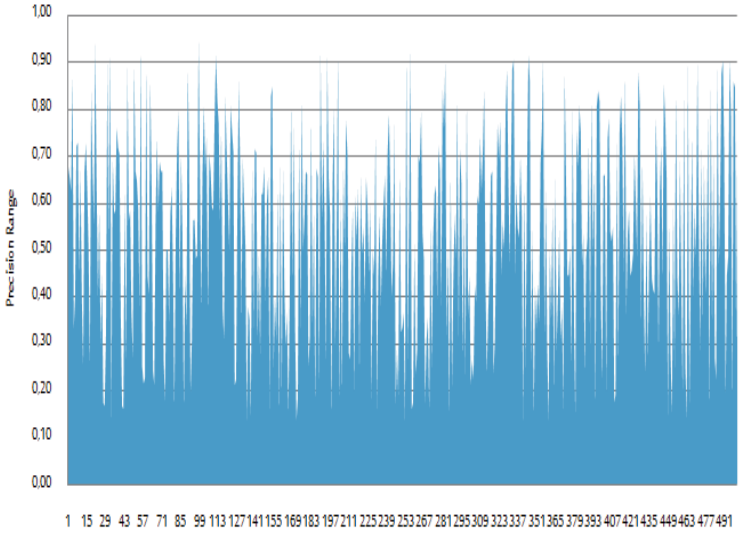


Figure 5. Range of the precision values throughout the set of 500 resumes

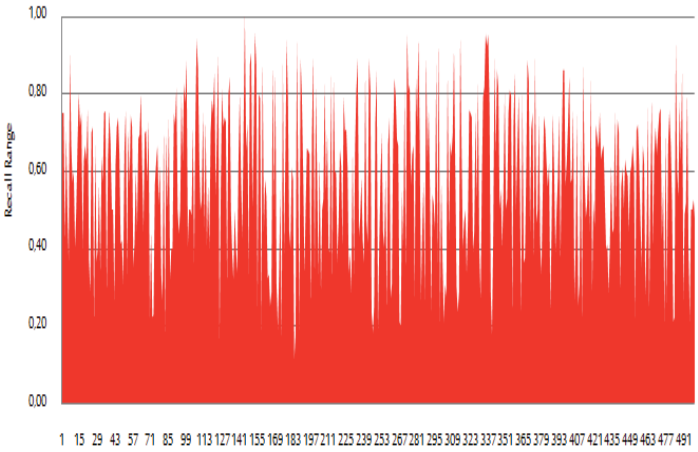


Figure 6. Range of the recall values throughout the set of 500 resumes

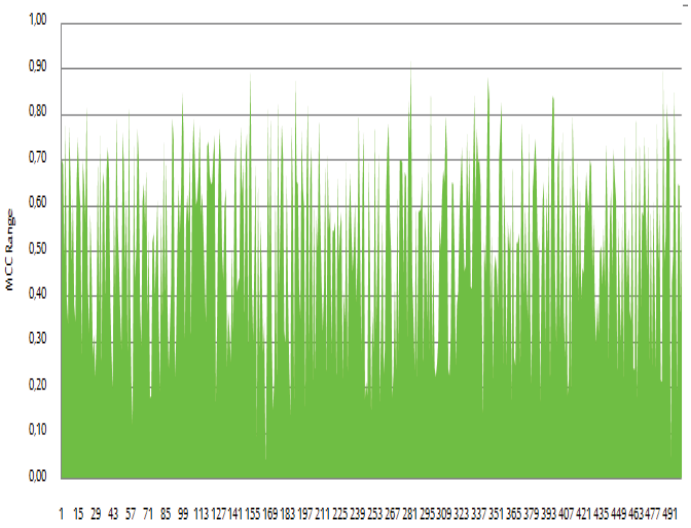


Figure 7. Range of the MCC values throughout the set of 500 resumes

B. *Testing the System on a Set of Free Form Resumes Entered by the Users*

We have also tested our system on a set of free form resumes – i.e. resumes written in natural language – that are entered by users through the graphical interface of our system. Below we present an example of how our system is able to identify "on the fly" the skills contained in a resume during the time the user enters the resume. The example also shows how the system updates the skill ontology with skills identified in the resume and though unknown semantically – i.e. not yet defined as concepts in the skill ontology. Figure 8 shows an example of a free form resume entered by a user.

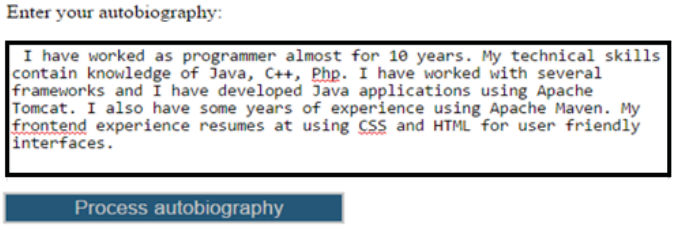


Figure 8. Example of a resume entered by a user

Based on the resume in Figure 8, the system is able to identify skills that are already in the skill ontology (see Figure 9), as well as new, unknown skills, that are not concepts in the ontology.

Figure 9. Skills recognized in the resume of Figure 8.

The system then updates the skill ontology by inserting the unknown skills as new concepts into the ontology (see Figure 10).

V. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a system for extracting professional skills from resumes. The system includes a training algorithm for inducing/ generating some specific, lexicalized patterns that would define the specific context in which a possible new skill could appear. Moreover, our approach proposes a matching algorithm for fitting a new skill into its corresponding class in the skill ontology. This means that any new skill is inserted as a new concept that becomes a child class of an existent class in the ontology. The paper also proposes a solution for overcoming the ambiguity problem caused by the polysemy of the phrases that express skills in the text of a resume. The solution is based on searching Wikipedia for every phrase that expresses a new skill. Our system has proved to be very efficient, since it performs very well on a large corpus of resumes. Also, there are 13,337 skills defined in the skill ontology, which are efficiently recognized/ found in the ontology. As further work, we will enlarge the set of standard, generic part-of-speech patterns, in order to be possible to parse not only English resumes, but also CV summaries written in Romanian or German.

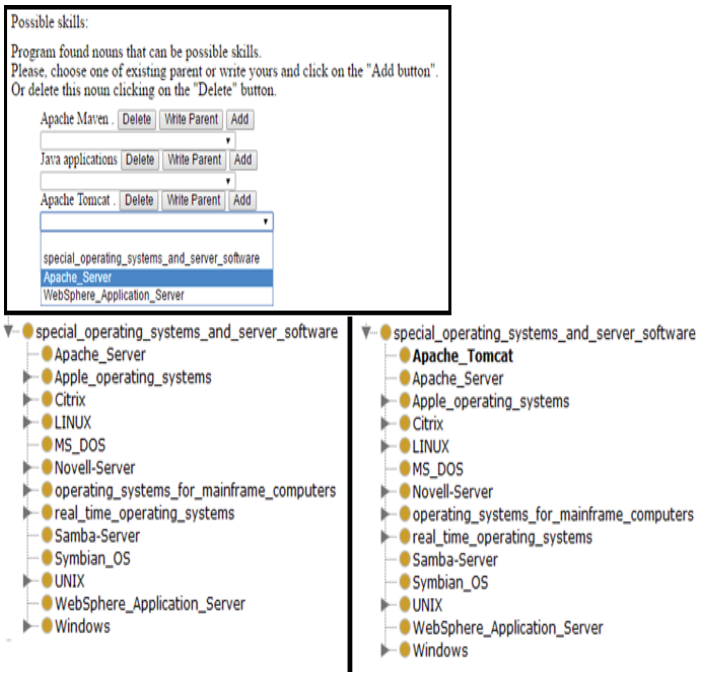


Figure 10. Detection of new, unknown skills in the resume of Figure 8.

REFERENCES

[1] A. Sen, A. Das, K.l Ghosh, and S. Ghosh, *Screener: a System for Extracting Education Related InformationFrom Resumes Using Text Based Information Extraction System*, 2012 International Conference on Computer and Software Modeling, vol. 54, pp. 31-35, 2012.

[2] M. Zhao, F. Javed, F. Jacob, and M. McNair, *SKILL: A System for Skill Identification and Normalization*, Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 4012-4017, 2015.

[3] C. Feilmayr, K. Vojinovic, and B. Pröll, *Designing a Multi-Dimensional Space for Hybrid Information Extraction*, 23rd International Workshop on Database and Expert Systems Applications (DEXA), pp. 121-125, 2012.

[4] S. Maheshwari, A. Sainani, and P.K. Reddy, *An Approach to Extract Special Skills to Improve the Performance of Resume Selection*, Databases in Networked Information Systems , vol. 5999 of the series Lecture Notes in Computer Science, pp. 256-273, 2010.

[5] K. Yu, G. Guan, and M. Zhou, *Resume Information Extraction with Cascaded Hybrid Model*, Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 499-506, 2005.

[6] S. Sonar and B. Bankar, Resume Parsing with Named Entity Clustering Algorithm, 2015. http://www.slideshare.net/swapnilmsonar/resume-parsing-with-named-entity-clustering-algorithm

[7] http://recruiters.ukstaffsearch.com

[8] https://www.hays.co.uk.

[9] Stanford Natural Proceesing Tool, http://nlp.stanford.edu/software/

[10] http://reverb.cs.washington.edu/README.html

[11] P. Cimiano, A. Pivk, L. Schmidt-Thieme, and S. Staab, *Learning taxonomic relations from heterogeneous sources of evidence*, In P. Buitelaar, P. Cimiano, and B. Magnini, (Eds.), Ontology Learning from Text: Methods, Evaluation and Applications, Frontiers in Artificial Intelligence and Applications Series. IOS Press, pp. 59-73, 2005.

[12] F-measure definition, https://en.wikipedia.org/wiki/Matthews_correlation_coefficient