Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 141 – 159

doi:10.3906/elk-1304-130

Towards a semantic-based information extraction system for matching résumés to job openings

Duygu ÇELİK*
Department of Computer Engineering, İstanbul Aydın University, İstanbul, Turkey

Received: 15.04.2013 • **Accepted/Published Online:** 18.10.2013 • **Final Version:** 01.01.2016

Abstract: A curriculum vitae or a résumé, in general, consists of personal details, education, work experience, qualifications, and references. The overall objective of this study was to extract such data as experience, features, and business and education information from résumés stored in human resources repositories. In this article, we propose an ontology-driven information extraction system that is planned to operate on several million free-format textual résumés to convert them to a structured and semantically enriched version for use in semantic data mining of data essential in human resources processes. The architecture and working mechanism of the system, similarity of the concept and matching techniques, and an inference mechanism are introduced, and a case study is presented.

Key words: Web semantics, web ontology language, résumé, semantic search agents, information extraction

1. Introduction

When a résumé is mentioned, a written document containing details of a person's education, work experience, skills, personal information, and so on comes to mind. Résumés are created and sent as individual free-format documents or as requested CV formats by a company for a job application via a human resources brokerage firm or directly to companies that call for the recruitment of staff. Although résumés are in different formats, they include common units of information. The purpose of this study was to extract the requested information from sections of a résumé into related sections using a semantic-based information extraction system and making this information presentable in relational data forms.

On the database of Kariyer.net, there are more than 6,000,000 unstructured and free-style résumés as MS Word documents written in both English and Turkish. The structure and information contents of résumés, their classification and collection under subtitles, and the presentation of information can be totally different. Gathering information from each of these résumés and storing it in the company database in a specific format will reduce possible losses in terms of human effort. There are many difficulties in servicing résumé databases faced by government agencies, commercial companies, and unions, which expend too much of their crucial resources such as time, capacity, human effort, money, and so on, on such servicing. Therefore, filtered résumés are likely to become the most important source of employment in human resources departments of government agencies and commercial companies.

The proposed ontology-driven information extraction system, named Ontology-Based Résumé Parser (ORP), will be executed on several million English and Turkish language résumés to convert them to ontological format (http://wwwksl.stanford.edu/kst/what-is-an-ontology.html). Moreover, the system will also assist in

*Correspondence: duygucelik@aydin.edu.tr

performing expert finding/detection and aggregation of skills information in a résumé pool through its involved semantic approach.

The development of English and Turkish semantic-based information parsing systems and the acquisition of data in a requested format can be a valuable qualification for many disciplines that are open to development. In the literature, a limited number of research studies on this topic considering only the English language can be found. Most of the studies did not use semantic web (SW) approaches (http://semanticweb.org/wiki/Main_Page).

The rest of this paper is organized as follows. Section 2 presents various recent studies about information extraction from résumés in the literature. Section 3 investigates the ontology knowledge bases used in the system. Section 4 explores the architecture and working mechanism of the system through a case study. Section 5 depicts the concept similarity matching techniques of the system. Section 6 presents an inference mechanism for the expert finding task through the system's rules, and Section 7 is dedicated to the conclusions.

2. Related works

Ontologies are considered pillars of the SW and can be developed through ontology languages that are types of knowledge representation languages, used for describing concepts, properties, and relationships among concepts of a domain. A SW-based term vocabulary can be considered as a special form of ontology, or also sometimes basically as a set of URIs with a (casually) described meaning. Recently, many ontology languages have been projected and standardized. such as the Resource Description Framework Schema RDF(S) (http://www.w3.org/RDF/), the Web Ontology Language (OWL) (http://www.w3.org/tr/owl-features/) and its new version OWL 2.0 (http://www.w3.org/TR/owl2-overview/), and so on. OWL expresses concepts in specific spaces, terms, and features in the form of ontology. In this way, it is possible to adapt the heterogeneous information from the distributed information systems. Until recently most of the research studies were designed for a résumé ontology/vocabulary in English: Bojārs and Breslin proposed the résumé RDF ontology in order to identify résumés semantically. This ontology was designed for human resources systems that reveal the structure of the 'authority finder'. With this ontology structure, it is possible to describe information about people, features, and skills semantically [1].

Another similar study is Description of a Career (DOAC). In this study, a vocabulary is suggested by Parada to describe résumés (http://wiki.foaf-project.org/w/DOAC). In DOAC, concepts related to information, features, capabilities, or skills of people are described. However, a limited number of concept descriptions were listed. In the résumé RDF, basic ontology topics like jobs, academic knowledge, experience, skills, publications, certificates, references, and other information were discussed. In the résumé RDF, a greater number of properties (73) are defined in information extraction semantically; more information can be obtained from queries. In contrast with the résumé RDF, in the DOAC, less semantic feedback can be achieved as a smaller number of properties (17) are defined. The comparison of the DOAC and the résumé RDF is depicted in Table 1.

Table 1. DOAC and résumé RDF comparison.

Ontology structure	DOAC	Résumé RDF
Classes/concepts	15	16
Properties/relationships	17	73
Year	2005	2002

Another system, proposed by Karamatlı and Akyokuş [2], is an IE-based system model doing finding and dismantling operations from résumés in four different sequential steps. These are text segmentation, scanning and identifying name property, classifying name property, and text normalization. In addition to the above-mentioned studies, some commercial products have been developed, such as: Sovren Résumé/CV Parser, ALEX Résumé Parsing, Résumé Grabber Suite, and Daxtra CVX. Usually they have used their own IE methods. For such systems, the related vocabulary was either developed by the commercial products or an existing vocabulary on the web such as WordNET2.x was used. In general, according to Karamatlı and Akyokuş, in terms of résumés, four different approaches to information extraction are mentioned for vocabulary-based systems: named entity-based, rule-based, statistical, and learning-based. According to the named entity-based method, words, phrases, or well-known patterns are matched either by using a vocabulary method such as in the Daxtra CVX project or with the regular expressions method [3]. With the rule-based method grammar rules are defined and information extraction is done through these rules [4]. In the statistical approach, solutions for a given document are produced through numerical modeling [5]. In the learning-based approach classification algorithms are used [6]. In the literature, some hybrid approaches combining several methods from these studies are also mentioned. For example, in the studies by Yu et al. [7] hidden Markov modeling is presented with a statistical approach, and Kowalkiewicz et al. presented the SVM modeling with the learning-based approach [4].

Within most of the above-mentioned systems, a set of résumés written in English are considered to be parsed through a syntactic-based matching algorithm instead of semantic matching. While matching, the system matches the extracted information from a document with the predefined résumé vocabulary. For example, the system can see that the abbreviation *'İAU'* is *'İstanbul Aydın University'* but it cannot understand the semantics of '*It is a university*', or, in other words *'İAU is a university'*. The overall objective of the ORP system proposed in this paper is based on a concept-matching task and ontological rules for English and Turkish résumés; it analyzes data semantically and parses required and related information such as personal details, experience, and business and education information from a résumé. The details are going to be stated in the next sections.

3. Ontology knowledgebase (OKB)

The ontology knowledgebase (OKB) of the ORP system contains many domain ontologies where each ontology has its domain-based concepts, properties, and relationships according to the segments of a personal résumé. These ontologies are education, location, abbreviations, occupations, organizations, concepts, and résumé ontologies. The ontologies and the purpose of the OKB are described below and Turkish translations of the ontologies are referenced in parentheses:

- The education ontology (eğitim ontolojisi–EO) keeps the concepts of comprehending words in terms of the education domain such as university (üniversite), high school (lise), and some properties among concepts such as 'has a degree', i.e. 'honor degree' (onur derecesi), that are related to the education domain. The ontology also keeps the individuals that are instances of entire education institutions such as 'İstanbul Aydın University'. İAU is an individual of the 'university' concept in the ontology.
- The location ontology (konum ontolojisi–LO) keeps the concepts of the location domain such as 'country' (ülke), 'city' (şehir), 'village' (köy), and so on. Besides, some properties such as '*hasPostalCode*' are declared as a property of a city.
- The abbreviations ontology (dil kısaltmaları ontolojisi–AO), for abbreviations of certain word groups such as 'İstanbul Aydın University', is a single concept and also keeps its properties such as '*hasAbbreviation*' having a property value of IAU.

- The occupations ontology (meslekler ontolojisi–OCCO) contains entire concepts of the used terms for the types of occupations such as ‘doctor’ (doktor), ‘chief assistant’ (şef yardımcısı), and ‘academician’ (akademisyen) and also includes their relations such as ‘*subClass*’, i.e. the ‘chief assistant’ concept is a subclass of the ‘chief’.
- The organizations ontology (organizasyonlar ontolojisi–OO) contains entire concepts for the types of companies/institutions such as ‘pastry-shop’ (pastane), ‘university’ (üniversite), or ‘hospital’ (hastane).
- The concepts ontology (kavramlar ontolojisi–CO) contains common concepts in general such as ‘date’ (tarih), ‘year’ (yıl), ‘month’ (ay), ‘day’ (gün), or ‘currency’ (para birimi) and their relationships.
- The résumé ontology (özgeçmiş ontolojisi–RO): there cannot be only one correct format or style for writing a résumé in English or Turkish, as well as other languages. Therefore, it can be summarized that the following types of information are generally used concepts by a résumé owner, and a typical organization can be seen in Section 3.1.

Moreover, all of the above mentioned ontologies in the OKB are generated in English but also contain < rdfs : label xml : lang = ‘tr’ > tags used for indicating the Turkish equivalent of each English concept as stated in the following form:

< Declaration >
< ObjectProperty IRI = ‘ # LanguageSkill’ / >
< rdfs : label xml : lang = ‘en’ > Language Skill < / rdfs : label >
< rdfs : label xml : lang = ‘tr’ > Dil Becerileri < / rdfs : label >
< / Declaration >

The above-mentioned domain ontologies are interconnected with each other through concepts, properties, and individuals. This interconnection property is needed for some special cases (i.e. a résumé ontology is a domain ontology in the OKB that is required for both the work experience and the education segments, since a person’s work place or company/institution information can be a university that they work in as well as a university that they study in (such as the individual of İstanbul Aydın University in the résumé ontology).

Effective detailing of the OKB will help in the understanding of the mechanism of the system. Therefore, in the following section only the resume ontology is explained in detail; additionally, the association among these ontologies is discussed.

3.1. Résumé (Özgeçmiş) ontology

The résumé (özgeçmiş) ontology (RO) is developed in order to express the semantic data contained in a résumé, such as personal information, work and academic experience, skills, publications, certifications, and so on. A personal résumé is described in an ontology form via a résumé upper ontology.

It is possible to annotate semantically the information of personal and work experiences, academic or educational life, skills, courses, certificates, publications, personal/professional references, and other information of a person in the résumé. Personal information uses many concepts such as *‘Name’*, *‘Address’*, *‘Email’*, *‘Mobile’*, *‘Home Phone’*, and *‘military service status’* of a person. The ‘*Education*’ concepts contain subconcepts such as ‘*dissertation*’, *‘certificates’*, *‘fellowships / awards’*, *‘areas of specialization’*, *‘areas of research’*, *‘teaching* *interests'*, *'teaching experience'*, *'research experience'*, *'publications/presentations'*, and *'related professional experience'* and their properties. The education segment in the RO is mostly associated with the EO.

The '*Work Experience*' segment involves semantic annotations for the current job, previous jobs, and hunted jobs, namely information about personal job preferences for the future. Furthermore, the RO is designed with querying in mind and is able to extract better semantic information from résumés. For example, '*Company*' is a concept in the RO that may be related to a person's current job, previous jobs, or targeted jobs that are annotated via a '*ro : employedIn*' (*ro : gecmis_is*) property used for work history, *'ro : isCurrentWork' (ro : halen_is*) used for the company information of current work, and *'ro : isGoalWork'* (*ro : hedef_is*) used for information about a person's target job.

Additionally, personal skills are considered in the RO and are designed as in the résumé RDF study proposed by Bojārs and Breslin [1]. The skill data can be described semantically by the *'ro : skillName'* (*ro : yetenekAdı*) concept in RO. Skill levels are also semantically described through *owl : objectProperty* named as '*ro : skillLevel'* (*ro : yetenekSeviye*) from 'bad' (kötü) to 'excellent' (mükemmel). Moreover, the *'ro : skillLastUsed'* (*ro : yetenekSonKullanım*), *'ro : skillHasCertificate'* (*ro : yetenekSertifika*), and *'ro : skillYearsExperience'* (*ro : yetenekYılSayısı*) are used since they allow to quantify skill levels particularly for foreign languages or software tools used. Thus, the RO uses literals to describe skills in the form of *owl : datatypeProperty* in order to avoid the uncertainty of skill identification and to enable fair skill matching. The concept 'skill' has many subclasses for language skills, driving skills, software skills, and tool or machine skills. They allow for describing semantically if a person has foreign language ability, has a driver license, or has used software/tools/machines and their skill levels respectively. The semantic declaration by the RO will assist to perform expert finding/discovery and the aggregation of skills information in a résumé pool.

In Table 2, a portion of the upper ontology of the résumé ontology is depicted. In fact, a résumé is characterized by its skills. The *'hasSoftware_Skill'* and *'isSoftware_SkillOf'* relations join the two classes together through a bidirectional link. They are inverses of each other, so they have inverted domains and ranges. The *owl : inverseof* construction can be used to de?ne such an inverse connection between relations through the *owl : ObjectProperty* (lines 16–22, Table 2). Skill assertion should be considered since people mostly mention their software/language/driving skills information on their résumés. In Table 2, the left column shows a small portion of the type of information on a résumé and the right column shows its ontology form of the RO.

4. Working mechanism of ORP

The ORP system performs six major steps: converting an input résumé, partitioning the résumé to some specific segments, parsing meaningful data from the input, normalizing, applying the classification and clustering task to structure the on-focus segment of the résumé, and finally creating the ontology form for the résumé. The modules of the system are the converter, segmenter, parser engine, normalization, classification and clustering of concepts, and generating personal résumé ontologies for individuals.

4.1. Converter

It will carry out the process of converting the given .doc, .docx, .pdf, .ps, and so on forms of résumés into plain text. In this case study, as a first step, a free formatted sample résumé (i.e. Mr Ali Budak's résumé on the right in Figure 1) is presented to the system as an input (.doc, .pdf, .txt, and so on.) that will be transformed to a plain text format through a converter (for example: .txt, Figure 1, steps 1 to 2).

Table 2. A small portion of the résumé (özgeçmiş) ontology–RO.

– Personal Information	< ! — A portion of the Résumé (Özgeçmiş) Ontology
– Education	in English Language -- >
• Dissertation	< owl : Class rdf : ID = "Résumé" / >
• Fellowships / Awards	< owl : Class rdf : ID = "Employee" / >
• Areas of Specialization	< owl : Class rdf : ID = "Company" / >
• Research and Teaching Interests	< owl : Class rdf : ID = "Skill" / >
• Teaching Experience	< owl : Class rdf : ID = "Software_Skill" >
• Research Experience	< rdfs : subClassOf rdf : resource = " # Skill" / >
• Publications / Presentations	< / owl : Class >
• Certificates	< owl : Class rdf : ID = "Driving_Skill" >
• Related Professional Experience	< rdfs : subClassOf rdf : resource = " # Skill" / >
– Work Experience	< / owl : Class >
• Current Work	< owl : Class rdf : ID = "Language_Skill" >
• Previous Work	< rdfs : subClassOf rdf : resource = " # Skill" / >
– Skill	< / owl : Class >
• Language	< owl : ObjectProperty rdf : about = " # hasSoftware_Skill" >
• Used Computer Software	< rdfs : range rdf : resource = " # Résumé" / >
• Driving License	< rdfs : domain rdf : resource = " # Tool" / >
– Activities	< owl : inverseOf >
– References	< owl : ObjectProperty rdf : about = " # isSoftware_SkillOf" / >
– Other	< / owl : inverseOf >
.	< / owl : ObjectProperty >
.	< owl : DatatypeProperty rdf : ID = " # hasDriversLicense" >
.	< rdfs : domain rdf : resource = " # Résumé" / >
	< rdfs : range rdf : resource = " & xsd : boolean" / >
	< / owl : DatatypeProperty >
	< owl : ObjectProperty rdf : ID = "WorksInCompany" >
	< rdfs : domain rdf : resource = " # Employee" / >
	< rdfs : range rdf : resource = " # Company" / >
	< owl : inverseOf rdf : resource = " # CompanyMembers" / >
	< / owl : ObjectProperty >

4.2. Segmenter

The outcome plain text will be segmented into parts such as personal information, education, work experience, personal experiences, and so on, with the help of a segmenter. Then the necessary parts will be pulled out and sent to the ORP parser engine. The case study's semantic-based segmentation process of the work experience segment is shown in Figure 1 (steps 3 to 4). Figure 1 depicts Mr Ali Budak's résumé transferred as an input résumé to the segmenter formatted as plain text (step 3). The system separates the segments (such as personal information, work experience, etc.) by using its OKB (steps 4 and 5) as shown above in yellow boxes (step 6). During segmentation, the ORP segmenter takes a number of sample terms from the résumé to differentiate particular segments of the résumé. During the sampling task, the terms from the résumé are matched with the concepts of the OKB through a semantic matching step (SMS) [8–12]. The concept-based SMS between two terms is discussed in Section 5.1.

4.3. Parser engine

At this step, the system parses proper names/concepts, abbreviations, suffixes, and prefixes of well-known patterns in sentences. During this decomposition process, as shown above, the OKB will be used. The OKB of the proposed system consists of education, place/space, abbreviations, personal information, concepts, and companies' ontology. In the example above, only the 'work experience' section is transferred to a parser engine as an input (Figure 1, step 7, or Figure 2, step 8). One by one, each section will be separated and transformed

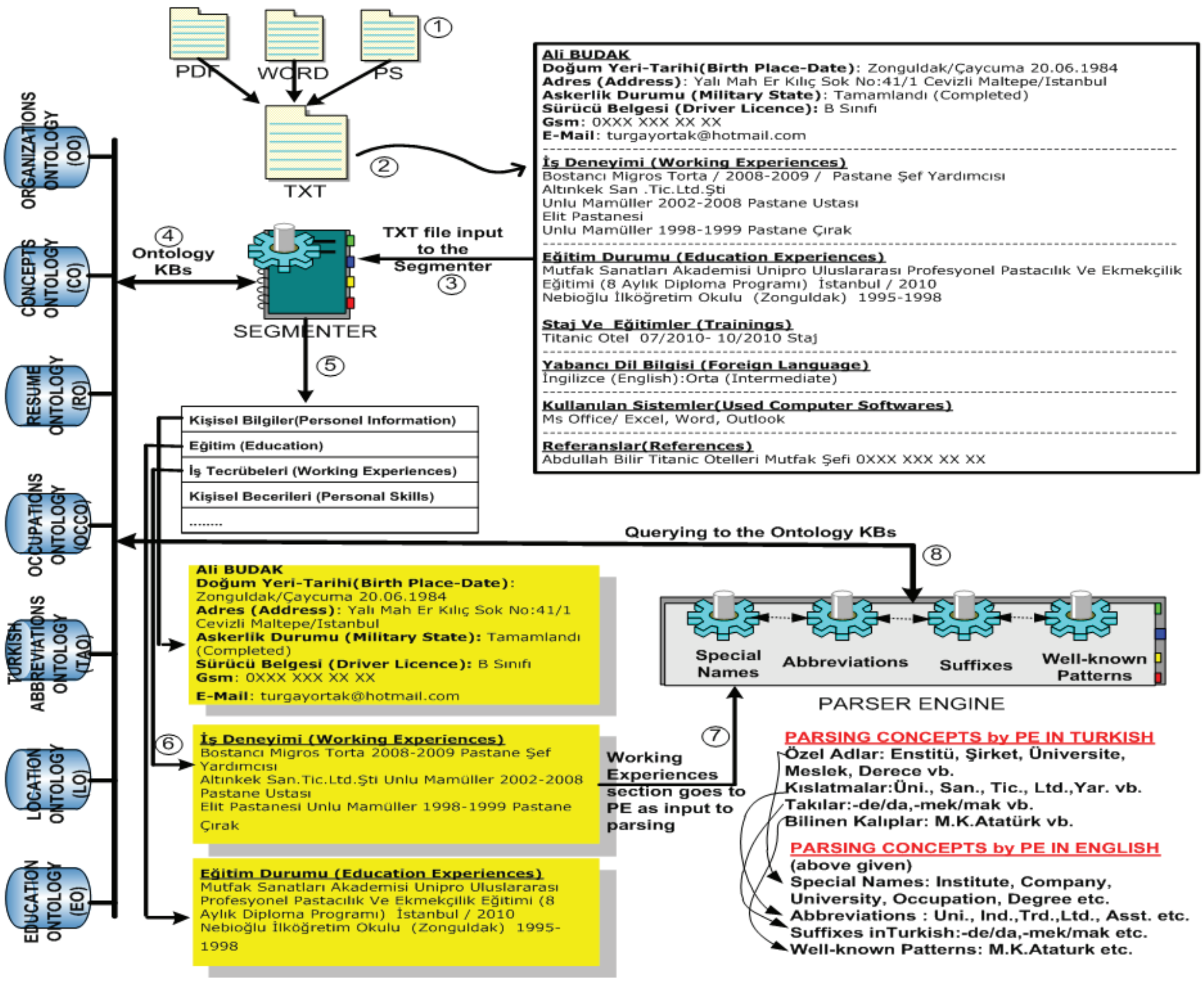

Figure 1. A sample résumé in Turkish and the system's segmenter engine working on it.

into the parser engine. The obtained output from the parser engine is a table (as shown in step 11 in Figure 2). The system can infer the concept of the part scanned in a sentence from ontologies, and also can keep the information of the start and finish lines for the next sentence.

In Figure 2, the yellow box contains the work experience segment of the sample résumé, which will be analyzed by the parser engine. Table 3 depicts English meanings of the used Turkish terms in the yellow box for a given sample work experience segment for a better understanding of this whole section.

The segment contains the sentence 'Bostancı Migros Torta / 2008-2009 / Pastane Şef Yardımcısı' (each term in this sentence is explained in Table 3). The system starts to detect each term of the segment in the concept base, tries to find each concept's URI information (in the OKB), and then determines the appropriate location (or field) to assign the correct storage in the system's database (Figure 2, steps 8 to 11).

4.4. Normalization

The system scans the abbreviations in the latest result table and converts them into standard usage (Figure 3, step 12). For example, in the second row of the work experience segment in Figure 2, the work experience information includes many abbreviations such as Ind. (San.) or Trd. (Tic.). The system converts these abbreviations into a normal form (Figure 3, steps 11 to 13).

Table 3. Eng/Tr translation of the given résumé above (for Figure 1).

Segments Turkish / English	**Ontology knowledge bases Turkish / English**	**Sample résumé given above Turkish / English**
Kişisel Bilgiler / Personal Information	Eğitim Ont. / Education Ont.	'Bostancı' is a location in İstanbul
Eğitim / Education	Yer Ont. / Location Ont.	'Migros' is a groceries company in Turkey
İş Tecrübeleri / Working Experience	Kısaltmalar Ont. / Abbreviation Ont.	'Torta' is an irrelevant unnecessary datum
Kişisel Becerileri / Personal Skills	Meslekler Ont. / Occupations Ont.	'Pastane' is a pastry-shop
	Özgeçmiş Ont. / Résumé Ont.	'Şef Yardımcısı' is a title of occupation that is called Assistant of the Chief
	Kavramlar Ont. / Concepts Ont.	'Altınkek' is a name of a company
	Organizasyonlar Ont. / Organizations Ont.	'San.' is an abbreviation for industry that is the same as 'Ind.' in English, and the same for others: 'Tic.' is Trade 'Trd.', 'Ltd.' is Limited 'Ltd.', and 'Şti.' is company 'Co.'

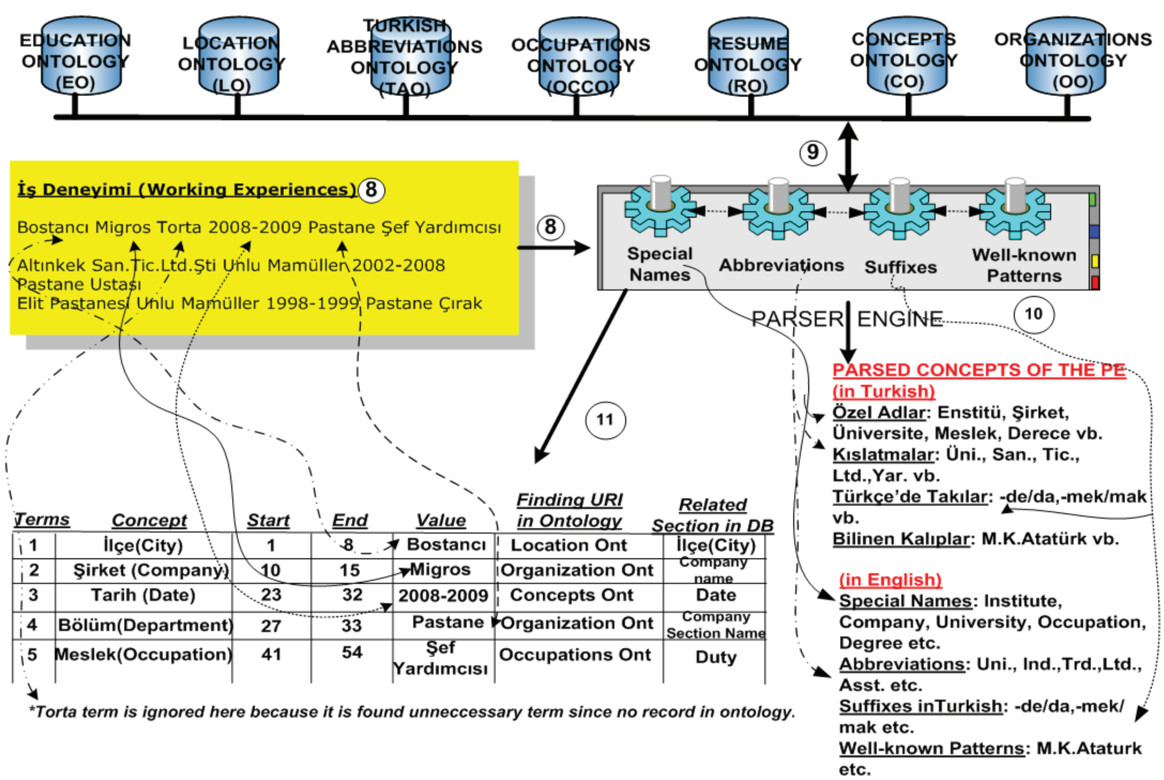

Figure 2. The parser continues to analyze the work experience segment of the same résumé according to special names, abbreviations, suffixes, and well-known patterns.

The transformation module detects the abbreviated concepts, retransforms them to a normal form, and assigns them into another table that does not involve any abbreviation concepts. For instance, the table may include the 'Ltd.' abbreviation (in the AO of the OKB), and the transformation module will convert it into the normal form 'limited', with its indicated concept in the CO of the OKB. In the final table, the transformation module always keeps the URIs of the fitting full concepts of abbreviations.

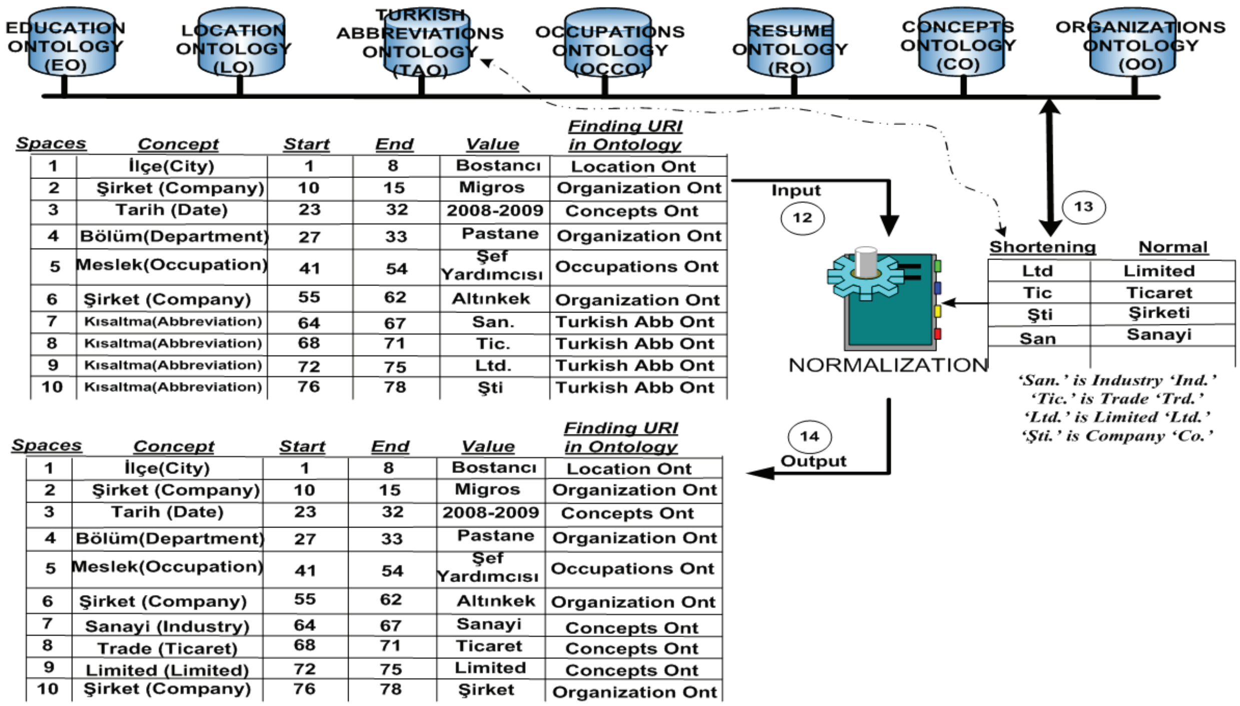

Figure 3. The system's normalization module converts involved abbreviations in the work experience segment.

4.5. Classification and clustering of concepts

In this step, the blending operation of the obtained concepts from the concept stack returned from the parser engine (as shown in the table in Figure 4) will be analyzed. As shown in Figure 4, there are three different phrases in the work experience segment of the sample résumé (shown in the business section).

The system uses suitable currently found concepts of the OKB to produce meaningful sentences during its classifications processes (Figure 4, steps 15 to 17). The module distinguishes each individual sentence in the work experience segment of the résumé according to the generated table of the transformation module in the previous step. The module starts to detect the starting and stopping points of each sentence according to the generated table. Each typical work experience sentence may contain some possible concepts such as a 'city', a 'date', a 'company', a 'department name', an 'occupation', and some other abbreviations. Therefore, the module will be able to define the start and stop points in a typical work experience sentence. Then it is able to guess the appropriate sentences of the work experience segment that were written by the résumé owner while developing the résumé. In Figure 4, the system finds three different work experience sentences, and then the system starts to perform structuring of its OWL ontology file for the currently focused résumé to keep reusing it during the expert finding task. The generation of the OWL form for a given résumé is investigated in the next section.

4.6. Generating personal résumé ontologies for individuals

At the end of the classification and clustering step, the system is able to generate the OWL form of an input résumé. For instance, the personal information section in a résumé may involve *hasBirthCity*, *hasBirthTown*, *hasBirthDate*, *hasCurrentAddress*, *hasCurrentTown*, *hasEMail*, and so on. Similarly, the past work experience section is converted through some crucial properties such as *WorkingExperience*, *hasWorkCity*, *hasWorkCom-*

pany, *hasWorkDate*, *hasStartWorkDate*, *hasEndWorkDate*, *hasWorkDuty*, and so on. The values of these properties are kept in the OWL form according to the RO upper ontology form for the person (Mr Ali Budak) shown in Table 4.

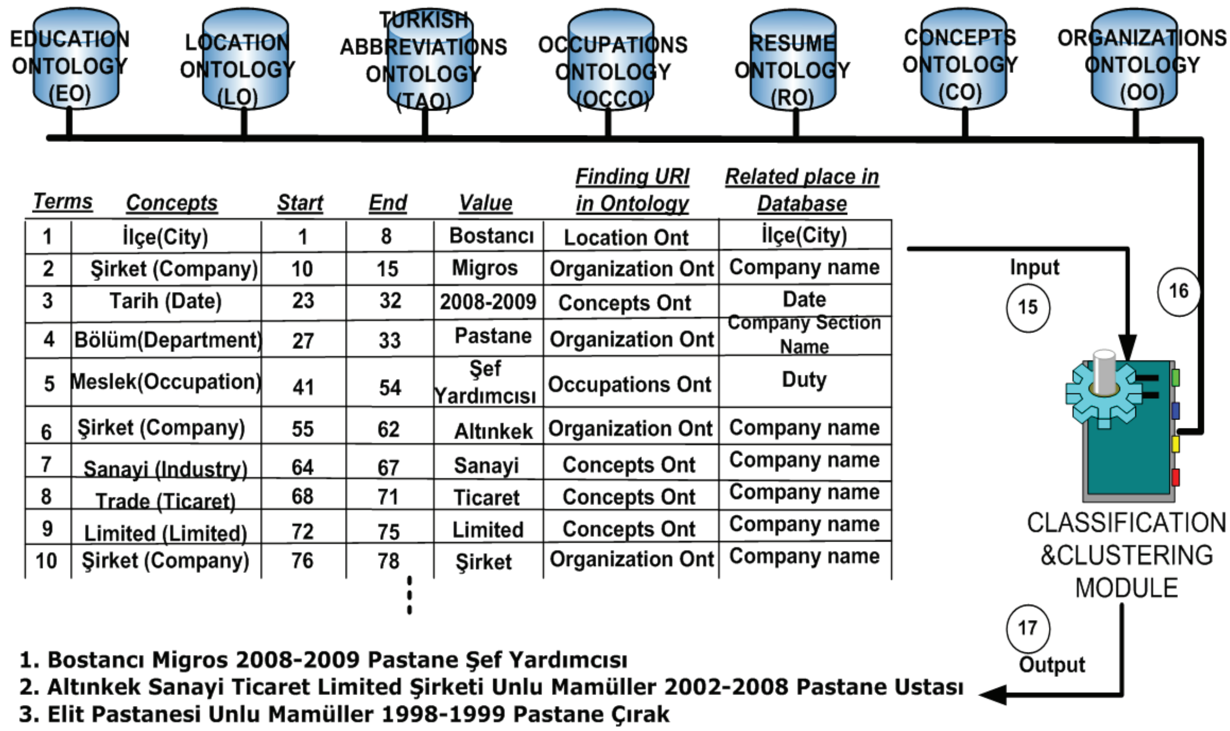

Figure 4. The system's classification and clustering module detects involved meaningful sentences in the work experience segment of the résumé.

The first result sentence after the classification and clustering module is depicted in an OWL form through the *WorkExperience* property (WorkExperience_1, lines 17–25 in Table 4). For this example, the first sentence of the work experience segment of the input résumé is 'Bostancı Migros Torta / 2008-2009 / Pastane Şef Yardımcısı'.Whole concepts of the sentence are determined and also the sentence is separated from other sentences of the sample segment at the end of the classification and clustering of concepts step of the ORP system. The property contains the *hasWorkCity* property that indicates a city concept. The city concept keeps a value that is 'Bostancı' under the *' & Location'* ontology (WorkExperience_1, line 18 in Table 4). The property contains the *hasWorkCompany* property that indicates a supermarket concept in the organization ontology. The supermarket concept has a value that is *'Migros'* (lines 19–20 in Table 4). All other OWL statements for the example will be similarly processed.

The generated OWL file of the input résumé will be kept by the Kariyer.net company to serve the human resources departments of their customer companies as an expert finder service. In Section 5, the expert finding mechanism having its specific, various, expert-finding, logical rules that are declared by using the semantic web rule language is investigated through a case study.

Table 4. A sample of a person's résumé (özgeçmiş).

	< ! — A portion of the Résumé (Özgeçmiş) Ontology in English Language -- >
Résumé No : R-0000001	< Résumé rdf : ID = " R-0000001" >
Name : Ali Budak	< hasBirthCity rdf : resource = " & Location ; Zonguldak" / >
Doğum Yeri-Tarihi (Birth Place-Date) :	< hasBirthTown rdf : resource = " & Location ; Çaycuma" / >
Zonguldak / Çaycuma 20.06.1984	< hasBirthDate rdf : datatype = " & Concept ; Date" > 20.06.1984
Adres (Address) : Yalı Mah Er Kılıç Sok No :	< / hasBirthDate >
41 / 1 Cevizli	< hasCurrentAddress rdf : resource = " & Location ; Street" / > Yalı Mah Er Kılıç Sok No : 41 / 1
Maltepe / Istanbul	Cevizli < / hasCurrentAddress >
Askerlik Durumu (Military State) :	< hasCurrentTown rdf : resource = " & Location ; Maltepe" / >
Tamamlandı (Completed)	< hasCurrentCity rdf : resource = " & Location ; İstanbul" / >
Sürücü Belgesi (Driver Licence) : B Sınıfı (B	< hasMilitaryState rdf : resource = " # Completed" / >
Class)	< hasDirverLicence rdf : resource = " # B Class" / >
Gsm : 0XXX XXX XX XX	< hasCurrentGSM rdf : datatype = " & Concept ; GSMNumber" > 0XXX XXX XX
E-Mail : turgayortak@hotmail.com	< / hasCurrentGSM >
İş Deneyimi (Work Experience)	< hasEMail rdf : datatype = " & Concept ; EMail" > turgayortak@hotmail.com
-Bostancı Migros Torta / 2008-2009 /	< / hasEMail >
Pastane Şef Yardımcısı	< WorkExperience rdf : about = "WorkingExperience_1" >
-Altınkek San.Tic.Ltd.Şti Unlu Mamuller	< hasWorkCity rdf : resource = " & Location ; Bostancı" / >
2002-2008 Pastane Ustası	< hasWorkCompany rdf : datatype = " & Organization ; SuperMarket" > Migros
-Elit Pastanesi Unlu Mamuller 1998-1999	< / hasWorkCompany >
Pastane Çırak	< hasWorkDate rdf : resource = " & Concepts ; Date" / > 2008-2009
……..	< / hasWorkDate >
Some terms from above résumé (in English) :	< hasWorkDepartment rdf : resource = " & Organization ; Pastane" / >
'Bostancı' is a location in İstanbul.	< hasWorkDuty rdf : resource = " & Occupation ; Şef_Yardımcısı" / >
'Migros' is a groceries company in Turkey.	< / WorkExperience >
'Torta' is an ignored unnecessary datum.	< WorkingExperience rdf : about = "WorkingExperience_2" >
'Pastane' is an organization that is pastry-	< hasWorkCompany rdf : datatype = " & Organization ; Pastane" > Altınkek San.
shop.	Tic. Ltd. Şti Unlu Mamuller < / hasWorkCompany >
'Şef Yardımcısı' is a title of an occupation	< hasWorkDate rdf : resource = " & Concepts ; Date" / > 2002-2008
type that is called Assistant of Chef.	< / hasWorkDate >
'Pastane Ustası' is a type of occupation that is	< hasWorkDepartment rdf : resource = " & Organization ; Pastane" / >
called Pastry-shop Chef.	< hasWorkDuty rdf : resource = " & Occupation ; Pastane Ustası" / >
'Çırak' is a title of an occupation type that is	< / WorkingExperience >
called apprentice.	< WorkingExperience rdf : about = "WorkingExperience_3" >
'Altınkek' is a name of a company.	< hasWorkCompany rdf : datatype = " & Organization ; Pastane" > Elit Pastanesi
'San.' is abbreviation of the industry that	Unlu Mamuller < / hasWorkCompany >
same as 'Ind.' in English and same for others :	< hasWorkDate rdf : resource = " & Concepts ; Date" / > 1998-1999
'Tic.' is Trade 'Trd.', 'Ltd.' is	< / hasWorkDate >
Limited 'Ltd.' and 'Şti.' is company 'Co.' so	< hasWorkDepartment rdf : resource = " & Organization ; Pastane" / >
on.	< hasWorkDuty rdf : resource = " & Occupation ; Çırak" / >
'Elit Pastanesi Unlu Mamuller' is a name of a	< / WorkingExperience >
company.	< / Résumé >
	< / rdf : RDF >

5. Matching concepts in the ORP system

Some modules of the ORP system use the semantic matching step (SMS) to find the matching degree between two terms. For instance, the segmenter of the ORP uses the SMS task to find separate segments of an input résumé through extracting some sample terms from an input résumé during the converter module. The parsing module also uses the SMS step to determine equivalency or similarity of two terms during the parsing process from a particular segment of an input résumé (such as the work experience, personal info, or education segments). Consequently, the SMS step is needed for some essential tasks such as specifying résumé segments of an input résumé, finding out falsely written vocabulary and the correcting step (i.e. the concept of *'University'* in the companies/organizations ontology that might be associated with a misspelled concept of '*Universty*' in an individual's résumé), finding synonyms, finding certain word groups and operating accordingly (for instance, the 'Grand National Assembly of Turkey' or 'GNAT'), finding the stem and suffixes of words and their usages while giving meaning, and so on. Besides, the system is able to determine that a concept of any ontology in the OKB is in line with the same predefined concept of another ontology during the concept matching step. For instance, the concept of *'University'* in the companies/organizations ontology that is associated with the concept of *'University'* in the education ontology. Consequently, the ontological associations are necessary in order to recognize a possible relativity/similarity relationship between a concept (in ontology) and an extracted term of a sample résumé document, especially on text spelling mistakes.

5.1. The semantic matching step (SMS)

The ORP takes the relationship/similarity among concepts into account (such as a synonym) for the domain of résumés before initiating the SMS execution. The purpose of this step is to detect if a parsed sentence from an input résumé would belong to a work experience sentence, personal information, or an education background sentence of the person. For instance, the work experience and education segments are the most complicated segments in a résumé, since, say, "*university*" may be a person's work place or school: for example 'I am working at İstanbul Aydın University as an instructor in the Computer Engineering Department since 2009' or 'I am studying at İstanbul Aydın University in the Computer Engineering Department since 2009'. To understand each term in the sentence, the system needs to specify its equivalent concept via domain ontologies.

At the beginning of the SMS, it parses a list of terms from a sample résumé, ***R***, and then finds all related concepts, ***A***, from a domain specific ontology in the OKB that are assigned to the SMS step. ***A*** is a list of the classes, super-/subclasses, individuals, and synonym concepts in the domain ontology (e.g., *RésuméOntology.owl*). Shortly, the inputs to the SMS are the requested parsed term list, ***R***, from a résumé/a résumé segment and also the list of all concepts/individuals, ***A***, of a domain ontology in the OKB. That can be functionalized as ***SMS (R, A)***. The output is a set of matched concepts sorted according to a similarity score. The SMS focuses on two concepts for each run, which are symbolized as R_i and A_i above. The SMS uses the matching scores of these two concepts, where dissimilar = 0, subsume = 0.5, plugin = 0.75, and exact = 1. The four degrees of similarity are related as follows: dissimilar < subsume < plugin < exact. The SMS is explained below, through Algorithm 1, where the following terms are used as defined.

Algorithm 1. degreeOfProcessMatching (concept R_i, Concept A_i)

1. if ((Ri ≡ Ai) or (hasSyn (Ri) ≡ Ai) or (Ri ≡ hasSyn (Ai))) then return rel = EXACT ;
2. if ((Ri ⊂ Ai) or (hasSyn (Ri) ⊂ Ai) or (Ri ⊂ hasSyn (Ai)) or (hasIs_a (Ri) ≡ Ai)) then return rel = PLUGIN ;
3. if ((Ri ⊃ Ai) or (hasSyn (Ri) ⊃ Ai) or (Ri ⊃ hasSyn (Ai)) or (Ri ≡ hasIs_a (Ai))) then return rel = SUBSUME ;
4. if ((Ri ≠ Ai) or (hasSyn (Ri) ≠ Ai) or (Ri ≠ hasSyn (Ai))) then return rel = DISSIMILAR;

D : The set of all concepts/classes and individuals defined in the selected domain ontology in the OKB.

R_I : A set of parsed terms (concepts) from an input résumé segment (e.g., education, work experience, personal information) from the currently focused résumé R.

R_i : A concept in set R_I.

A_i : A concept in the set of all concepts in the D.

Exact relation: A 1–1 mapping $R_i \rightarrow A_i$ such that $Concept(R_i) \equiv Concepts(A_i)$: if any two focused concepts (R_i and A_i) with the same or equivalent concepts or having hasSyn () relation. For instance, if (R_i= 'Driving_Licence' and A_i= 'Driving_Licence') or (R_i= 'Driving_Certificate' and A_i= 'Driving_Licence') or (R_i= ' Driving_Licence' and A_i= 'Driving_Certificate') then an exact relation exists, giving a score of 1.

Plugin relation: A 1–1 mapping $R_i \rightarrow A_i$ such that $Concept(R_i) \subset Concepts(A_i)$: if any focused input concept (R_i) is a subset or subclass of the concept of A_i , or has *hasSyn* () or *hasIs_*a () property, then a plugin relationship exists. For instance, if (R_i= 'Driving_Skill' and A_i = 'Skill') or (R_i = 'Driving_licence' and A_i = 'Skill') or (R_i= 'Driving_:Skill' and A_i = 'Driving_Licence'), and so on, then a plugin relation exists, giving a score of 0.75.

Subsume relation: A 1–1 mapping $R_i \rightarrow A_i$ such that $Concept(R_i) \supset Concepts(A_i)$: if any focused input concept (R_i) is a superclass of A_i , or has a h*asSyn* () or *hasIs_*a () property, then a subsume relationship exists. For instance, if (R_i= 'Skill' and A_i= 'Driving_Skill') or (R_i= 'Driving_Licence' and A_i= 'Driving_Skill') or (R_i= 'Skill' and A_i = 'Driving_Licence'), and so on, then a subsume relation exists, with a score of 0.5.

Dissimilar relation: A 1–1 mapping $R_i \rightarrow A_i$ such that $Concept(R_i) \neq Concepts(A_i)$: if there is no relation between the concepts R_i and A_i then a dissimilar relationship exists.

5.2. Jaro–Winkler distance algorithm

If a term is not meaningful or correctly written, then the system executes the Jaro–Winkler distance algorithm [13,14] to find the correct concept for the term. While applying SMS, the ORP also considers the Jaro–Winkler distance algorithm, designed and best suited for matching strings to measure the syntactic similarity between two terms. We consider detecting incorrectly spelled words in a résumé that are easily noticed and corrected through this algorithm.

The Jaro distance, d_j, is calculated between two given strings s1 and s2 as:

$$d_j = \begin{cases} 0 \quad \text{if} \quad m = 0 \\ \frac{1}{3}\left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m}\right) \quad \text{otherwise} \end{cases} \tag{1}$$

- $\boldsymbol{m}$ is a value based on the number of matching characters
- $\boldsymbol{t}$ is half of the number of transpositions
- $\boldsymbol{p}$ is the constant scaling factor denoting how much the score has been adjusted upwards for common characters; the standard value for $\boldsymbol{p}$ in Winkler's work is 0,1
- $\boldsymbol{l}$ is the prefix length (number of starting characters in both strings that matched)

The Jaro–Winkler distance, d_w, is:

$$d_w = d_j + (l.p.(1 - d_j)) \tag{2}$$

where d_j is the Jaro distance for the strings s1 and s2. In this equation, the weighted metric (l . p) will not exceed 1, and the final result will always be within the 0–1 range of the Jaro metric. Additionally, it guarantees that the result of d_w will never be lower than the result of d_j alone. It effectively lets d_w enrich d_j by filling in the remaining gap.

If the system executes these two terms of JaroWinklerDistance (‘universty’, university’), then it will get the d_w as 0.9900001 (Eq. (3)), which means it is a good match. Because 1 is an exact match, 0. 9900001 is very close to it. Typically, the system will perform these calculations:

$$ifs1 = Universtys2 = University \Leftrightarrow s1 = 9, s2 = 10, m = 9, t = 0 then d_j is;$$

$$\begin{aligned} d_j &= \frac{1}{3}\left(\frac{9}{9} + \frac{9}{10} + \frac{9-0}{9}\right) = 0,966667 \\ d_j &= 0,966667, l = 7, p = 0.1 then d_w is; \\ d_w &= 0,966667 + (7.0, 1.(1 - 0,966667)) = 0,9900001 \end{aligned} \tag{3}$$

Finally, note that the overall distance via the Jaro–Winkler distance d $_w$ in Eq. (3) is an exact match between the parsed misspelled ***‘Universty’*** term from the input résumé and the *‘University’* concept in the OKB.

6. Inference in expert finding through SWRL

The semantic web rule language (SWRL) [15] is an expressive OWL-based rule language that provides more powerful deductive reasoning capabilities than OWL alone. A SWRL rule contains an antecedent part (body) and a consequent one (head). Both the body and the head consist of positive conjunctions of atoms:

$$BODY\{Atom \wedge Atom \ldots\} \rightarrow HEAD\{Atom \wedge Atom \ldots\}$$

An atom is an expression of the form that contains a predicate symbol such as P and also some parameters such as$par_1, par_2 ... par_n$. The predicate symbol P can be OWL classes, object properties, or data type properties. P may contain some parameters that can be OWL individuals or data values, or variables referring to them in the expression $P(par_1, par_2 ... par_n)$ such as $hasExperienceYR(?p, ?yr) or greaterThan(?yr, 3)$ where ‘ ? p ’ is a variable parameter used instead of an individual of the ‘person’ class. The ‘ ? yr ’ is a variable parameter that is used to keep the information about years of experience of a specialist for any field. The greaterThan predicate also takes two parameters: the first parameter ‘? yr’ keeps a value that is ‘3’ as the second parameter in the given example above. The system has its own SWRL rules that are based on particular expert specification queries answering the requirements of human resources department staff. For instance, a human resources staff is looking for a ‘Java Supervisor’ expert as a candidate for a company’s job position and searches for a person who has a university degree, Java knowledge, and more than 3 years of experience.The concepts of a person and a university can be captured from the OWL classes called ‘person’ and ‘university’. The length of experience, degree, and computer skill conditions can be expressed as *hasExperienceYR*, *hasdegreefrom*, and *hasComputerSkill* object properties form of this OWL formed résumé. This rule could be written in SWRL form as below and depicted as an ontology form in the protégé tool of the rule in Figure 5.

Person(? p) ^ University(? u) ^ hasComputerSkill (? p ,? JAVA)
^ hasdegreefrom(? p ,? u) ^ hasExperienceYR(? p ,? yr) ^
greaterThan(? yr ,3) → hasTitle(? p , JavaSupervisor)

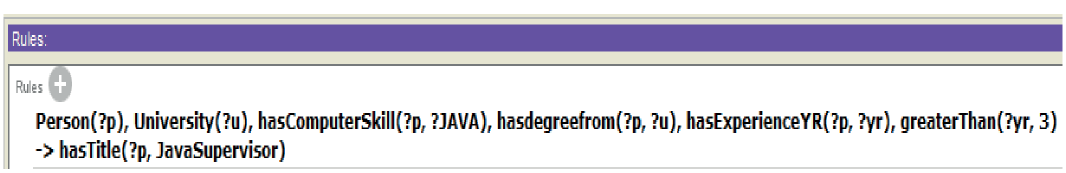

Figure 5. Rule syntax form SWRL of the protégé editor.

This rule implies that the *hasTitle* object property will be assigned as a *'Java supervisor'* for all OWL individuals that are members of the OWL Class *'person'* and have a degree from a university, which is a member of the OWL class 'university' and has appropriate classifications inferred from *hasExperienceYR*, *hasdegreefrom*, and *hasComputerSkill* object properties. According to the SWRL written rule (depicted in Figure 5), for all members of a person class that meet all of the conditions, the *'hasTitle'* property is inferred as '*Java supervisor*' by the Pellet reasoner [16] and a new assignment can be obtained even if it is not stated in the OWL form of that résumé.

The Pellet reasoner is used to obtain the associations among the classes/property assertions of a sample résumé ontology file (i.e. Murat Kalın in Figure 6) and the predefined SWRL rules for expert finding purposes to perform the inference task through the ontologies in the OKB. Figure 6 depicts three predefined facts that were already asserted before: *hasExperienceYR*, *hasdegreefrom*, and *hasComputerSkill* (highlighted in blue). After executing the Pellet reasoner, the system retrieves a new fact that is the *hasTitle* object property (4th row in Figure 6) as a *JavaSupervisor* for the person.

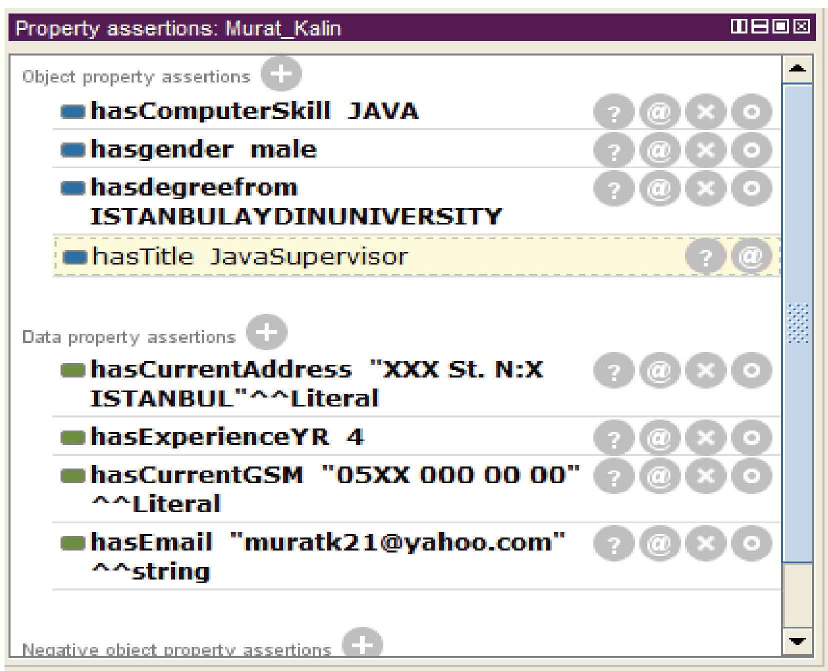

Figure 6. Object and data property assertion of a sample person's résumé ontology file.

Algorithm 2 is designed to bring out the inferred data for the sample resume from the above given sample SWRL rule according to the OKB. Once the algorithm is implemented, it brings the individuals of the OWL class 'Person' to a list (*OWLNamedIndividual individual*, line 12). Also, the *hasTitle* object property (*OWLObjectProperty op*, line 13) is assigned to each individual of the entire person class.

As seen from the code above, it starts by getting information about the OWL class person and assigns all of its members to a list in a loop between lines 15 and 19 in Algorithm 2. The extra control statement for the OWL classes is added to specify whether they are asserted or inferred with the help of a reasoner (the second box on the right side of Figure 7).

Algorithm 2. Inferring *hasTitle* object property (for a Person)

```
public List getAnObjectProperty () {
List < String > list = new ArrayList < String > () ;
public static final IRI localLocation_IRI = IRI.create (”http : / ..... / ResumeOntology.owl”) ;
public static final IRI Ont_Base_IRI = IRI.create (”http : / / Aydin.edu.tr / 2013 / ResumeOntology.owl”) ;
OWLOntologyManager m = OWLManager.createOWLOntologyManager () ; / / create an ontology manager
OWLDataFactory f = OWLManager.getOWLDataFactory () ; / / create an ontology factory
OWLOntology o = null ; / / create an OWL formed ontology file
private OWLReasoner reasoner ; / / create an OWL Reasoner instance
try {
o = m.loadOntologyFromOntologyDocument (localLocation_IRI) ; / / ontology URL info
PelletReasoner r = PelletReasonerFactory.getInstance ().createReasoner (o) ; / / create a Pellet Reasoner
OWLNamedIndividual individual = f.getOWLNamedIndividual (IRI.create (Ont_Base_IRI + ” # Person”)) ;
OWLObjectProperty op = f.getOWLObjectProperty (IRI.create (Ont_Base_IRI + ” # hasTitle”)) ;
NodeSet < OWLNamedIndividual > value = r.getObjectPropertyValues (individual, op) ;
for (OWLNamedIndividual rangeVal : value.getFlattened ()) {
System.out.println (labelFor (individual, o) + ” – > ” + labelFor (op, o) + ” – > \n” + labelFor (rangeVal, o)) ;
/ / add to list
list.add (labelFor (individual, o) + ” – > ” + labelFor (op, o) + ” – > ” + labelFor (rangeVal, o).toString ()) ;
}
m.removeOntology (o) ;
} catch (Exception e) {
System.out.println (”Could not create ontology : ” + e.getMessage ()) ;
}
return list ;
}
```

When a SWRL rule concludes with a new information assertion for an individual, the reasoner will infer this new information and the inferred information will also be stored in the ontology file of the resume belonging to that person. In conclusion, the SWRL is used for reasoning through predefined SWRL rules in the OKB. Another approach of expressing rules is description logic (DL) [17], which is used in artificial intelligence for formal reasoning on the concepts of an application domain. Both SWRL and DL are very powerful languages, which are needed both for the same application and for different cases. Also, in some cases it is possible to use both languages, the SWRL and the DL, with specific inference engines, one for the structural part (e.g., OWL DL for the ontology) and another one for the rule component (e.g., SWRL or other rule or logic programming language). According to another similar study by Golbreich et al. [18], applications need rules to extend the expressiveness of OWL and they require reasoning with rules in conjunction with the ontology for problem solving. These “hybrid” systems or languages are more complex because of decidability and complexity issues. The combination of OWL DL and rules is undecidable [15,19].

Golbreich et al. mentioned a medical case study that required reasoning with an OWL ontology extended by rules. They also focused on OWL queries in their research. They mentioned that the SWRL or the DL rules

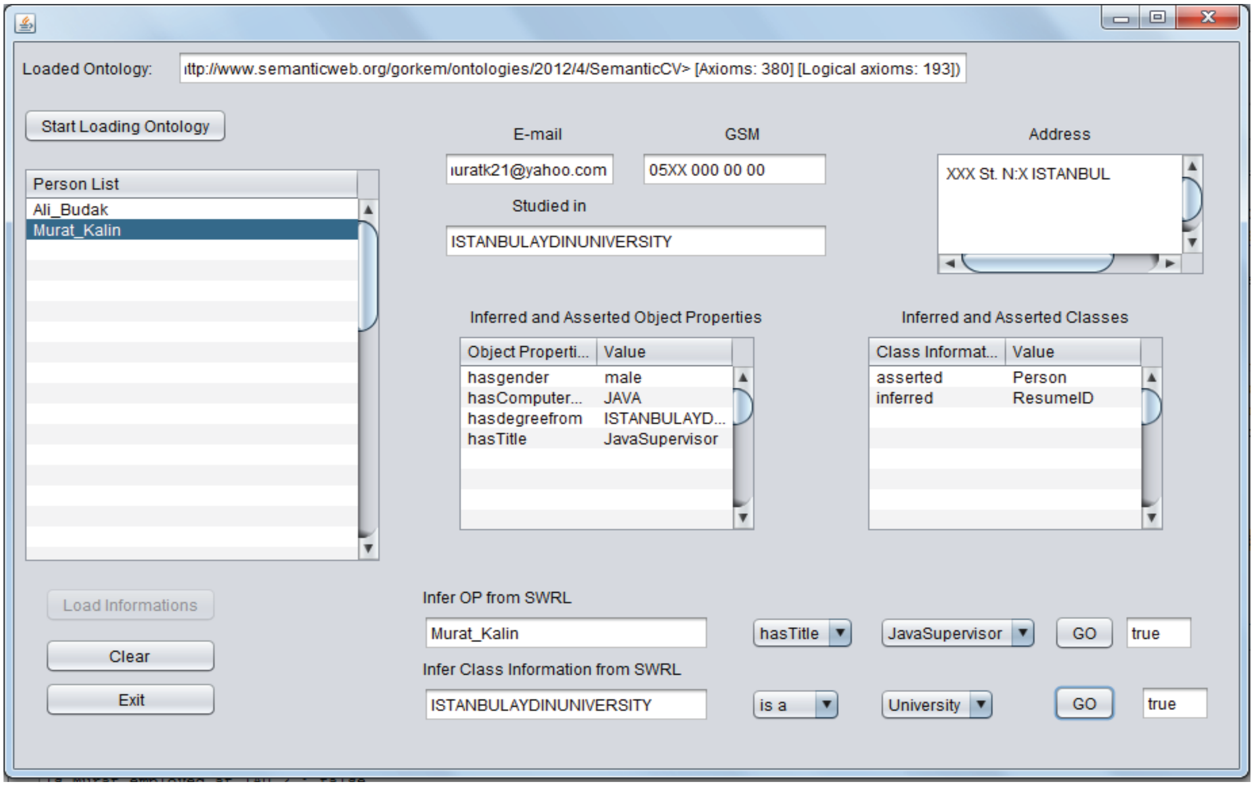

Figure 7. The system infers a new fact, that is the *hasTitle* property as a *JavaSupervisor*.

are similarly useful to express queries. They used both SWRL and DL rules in the same OKB for querying and reasoning purposes.

P1: property (?x,?y) and **P2: property (?x,?y,?z)**

As mentioned in [15,19], for the above given example properties, P1 can be expressed through the SWRL. However, the SWRL cannot express P2. P2 can be expressed by DL but it is still not suitable to describe such a "triangle", Rule 1 in Figure 5. The current facts are initial facts asserted by the user or facts issued from the ontology. Inferred facts are the facts derived at the current step. According to the above example, the system is looking for a Java expert, but the person should have some assets such as computer skills, should know Java or have his degree, should be from a computer engineering background, and so on (asserted facts from the user). The system tries to infer 'Is the person a Java Supervisor or not? The Pellet reasoner of the system scans all asserted facts and rules to get the new fact for him. Figure 6 presents 'He is a JavaSupervisor' after running the Pellet reasoner according to Rule 1 (Figure 5).

7. Conclusions

The proposed ontology-driven information extraction system, called ORP, will be operated on several million English and Turkish résumés for converting the résumés to an ontological format. The overall objective of the proposed ORP system is based on a concept-matching task and ontological rules for English and Turkish résumés that provide semantic analysis of data and parse related information such as experience, features, and business and education information from a résumé. The system contains various ontologies in its own OKB. Turkish and English clarifications are used for a better comprehension of the system mechanism for the case study section of the article. The system has its own SMS that is applied between two concepts, one from a résumé and the other from a related domain ontology in the OKB, to calculate a similar score. To conclude, the working system mechanism, the OKB, the matching steps, the transfer of plaintext résumé into ontology form, the case studies, and the inference mechanism though the SWRL rule base were discussed in this article. Further details of the above-mentioned system properties will be discussed separately in future studies of the project.

Acknowledgments

This project was supported by the Kariyer.net company in a project titled 'Document-Based Semantic Information Extraction System from Turkish Résumés through Ontology'. This article is an extended version of the project paper originally presented at the ICPCA / SWS 2012 7th International Pervasive Computing and the Networked World Conference, 26–28 October 2012, İstanbul, Turkey [20].

References

[1] Bojars U, Breslin JG. RésuméRDF: expressing skill information on the semantic web. In: 1st International Workshop on ExpertFinder; 2007; Berlin, Germany.

[2] Karamatlı E, Akyokuş S. Résumé information extraction with named entity clustering based on relationships. In: INISTA 2010 International Symposium on Innovations in Intelligent Systems and Applications; 2010; Kayseri, Turkey.

[3] Grishman R. Information extraction techniques and challenges. In: International Summer School on Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology; 1997. London, UK: LNCS. pp. 10–27.

[4] Kowalkiewicz M, Kaczmarek T, Piskorski, J. Information extraction from CV. In: BIS 2005 8th International Conference on Business Information Systems; 20–22 April 2005; Poznan, Poland. pp. 185–189.

[5] Sarawagi S. Information extraction. J Found Trends DBs 2008; 1: 261–377.

[6] Chieu HL, Ng HT, Lee YK. Closing the gap: learning-based information extraction rivaling. In: ACL 2003 Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics; 2003; Sapporo, Japan.

[7] Yu K, Guan G, Zhou M. Résumé information extraction with cascaded hybrid model. In: ACL 2005 Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics; 2005; Morristown, NJ, USA. pp. 499–506.

[8] Paolucci M. Semantic matching of Web service capabilities. In: ISWC2002 Proceedings of the 1st International Semantic Web Conference; 2002; London, UK. Berlin, Germany: Springer-Verlag. pp. 333–347.

[9] Bellur U, Vadodaria H, Gupta A. Semantic matchmaking algorithms. In: Bednorz W, editor. Advances in Greedy Algorithms. Vienna, Austria: IN-TECH Publications, 2008. pp. 481–502.

[10] Ilhan ES, Akkus GB, Bener AB. SAM: Semantic advanced matchmaker. In: SEKE 2007 19th International Conference on Software Engineering and Knowledge Engineering; 2007; Boston, MA, USA. pp. 698–703.

[11] Senvar M, Bener AB. Matchmaking of Semantic Web services using semantic-distance information. In: 4th Biennial International Conference on Advances in Information Systems; 2006. Berlin, Germany: Springer-Verlag, 2006. pp. 177–186.

[12] Celik D, Elci A. Ontology-based matchmaking and composition of business processes. In: Elci A, Mamadou TK, Jack A, Orgun M, editors. Semantic Agent Systems-Foundations and Applications (SASFA) Studies in Computational Intelligence. Heidelberg, Germany: Springer-Verlag, 2011. pp. 133–157.

[13] Winkler WE. The State of Record Linkage and Current Research Problems. Technical Report. Washington, DC, USA: U.S. Census Bureau, 1999.

[14] Winkler WE, Thibaudeau Y. An Application of the Fellegi-Sunter Model of Record Linkage to the 1990 U.S. Decennial Census. Technical Report. Washington, DC, USA: U.S. Census Bureau, 1991.

[15] Horrocks I, Patel-Schneider P, Harold B, Tabet S, Grosof B, Dean M. SWRL: a semantic web rule language combining OWL and rule ML. In: W3C; May 2004.

[16] Sirin E, Parsia B. Pellet: An OWL DL reasoner. In: ISWC2004 3rd International Semantic Web Conference; November 2004; Hiroshima, Japan.

[17] Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider P, editors. The Description Logic Handbook: Theory, Implementation, Applications. Cambridge, UK: Cambridge University Press, 2003.

[18] Golbreich C, Dameron O, Bierlaire O, Gibaud B. What reasoning support for ontology and rules? The brain anatomy case study. In: OWLED 2005 Workshop on OWL: Experiences and Directions; 11–12 November 2005; Ireland. pp. 3–11.

[19] Horrocks I, Patel-Schneider P, Bechhofer S, Tsarkov D. OWL rules: a proposal and prototype implementation. J Web Semantics 2005; 3: 23–40.

[20] Celik D, Elci A. An ontology-based information extraction approach for résumés. In: ICPCA / SWS 2012 7th International Pervasive Computing and the Networked World; 26–28 October 2012. İstanbul, Turkey: LNCS. pp. 4562–4568.