Ontology-Based Information Extraction from the Configuration Command Line of Network Routers

A. Martínez[1], M. Yannuzzi[1], R. Serral-Gracià[1], and W. Ramírez[2]

[1] Networking and Information Technology Lab (NetIT Lab)
[2] Advanced Network Architectures Lab (CRAAX)
Technical University of Catalonia (UPC), Barcelona, Spain
{annym, yannuzzi, rserral, wramirez}@ac.upc.edu

Keywords: Ontology-Based Information Extraction, Knowledge Discovery, Routers, Configuration Management, Data Mining, Semantic Relatedness

Abstract. Knowledge extraction is increasingly attracting the attention of researchers from different disciplines, as a means to automate complex tasks that rely on bulk textual resources. However, the configuration of many devices in the networking field continues to be a labor intensive task, based on the human interpretation and manual entry of commands through a text-based user interface. Typically, these Command-Line Interfaces (CLIs) are both device and vendor-specific, and thus, commands differ syntactically and semantically for each configuration space. Because of this heterogeneity, CLIs always provide a *"help"* feature—i.e., short command descriptions encoded in natural language—aimed to unveil the semantics of configuration commands for network administrators. In this paper, we exploit this feature with the aim of automating the abstraction of device configurations in heterogeneous settings. In particular, we introduce an Ontology-Based Information Extraction (OBIE) system from the Command-Line Interface of network routers. We also present ORCONF, a domain Ontology for the Router CONFiguration domain, and introduce a semantic relatedness measure that quantifies the degree of interrelation among candidate concepts. The results obtained over the configuration spaces of two widely used network routers demonstrate that this is a promising line of research, with overall percentages of precision and recall of 93%, and 91%, respectively.

1 Introduction

One of the most complex and essential tasks in network management is router configuration. Overall, router configuration encompasses a large number of functions, which include, setting routing protocols, security filters, interface parameters, forwarding rules, QoS policies, etc. Currently, network administrators rely on Command-Line Interfaces (CLI), as the means for configuring their

routers [1]. This is typically the case of administrators operating in Internet Service Providers (ISPs), data centers, corporate networks, public administrations, etc. Due to the lack of a better alternative, administrators must deal with the complexities associated with this practice, since CLIs are generally device and vendor-specific. This is further exacerbated by the heterogeneity of today's network infrastructures, as networks are typically composed of routers from different vendors. Accordingly, network administrators are forced to gain specialized expertise for a wide range of devices, and to continuously keep knowledge and skills up to date as new features, operating system upgrades, or technologies appear in the routing market. In light of all this, router misconfiguration is a common event, which often leads to serious service disruptions [2].

From reviewing the literature, it becomes evident that, over the last few years, academic and industrial communities have devoted considerable efforts to overcome the inherent complexities of the so-mentioned CLIs or the cumbersome and rarely used configuration means provided by the Simple Network Management Protocol (SNMP) [3]. Clearly, the need for standards has always been among the best interests of the Internet community, since they are essential to achieve interoperability at a global level. In light of this, the NETCONF protocol [4] emerged as an attempt for standardizing router configurations. However, the lack of a common data model, along with its slow and scarce adoption, have made it fail as the preferred mechanism for router configuration [5]. Some industrial initiatives have also attempted to provide uniform configuration means, by developing and maintaining dedicated software agents. Unfortunately, this approach demands serious development efforts, which are neither scalable nor easy to maintain under the dynamics of current networking environments.

In light of this, there is a need to explore other fields that can help pave the way toward seamless network device configuration. The use of ontologies and other semantic techniques has been considered in the past for achieving interoperability in the network management domain [6–11]. However, many of these works target interoperability at the network-level, but they do not approach the underlying interoperability issue at the element-level, i.e., device configuration management. The work in [9] is the most closely aligned to our research goals. Their main contribution is a semantic similarity function that enables mappings between ontologies of different router configuration domains. In that work, ontologies for each available device are assumed to be given in advance, but the task of building an ontology is quite challenging and sufficiently complex by itself. Unless vendors start providing the ontologies—which formally define the knowledge of their own configuration space—and the corresponding plugins, scalability is a major concern for a solution of this nature. Our approach is considerably different since it aims to be independent of device-specific knowledge models (i.e., ontologies) and instead, we automatically build these models from the information natively provided by network vendor's in their CLIs.

Despite numerous efforts, there is still a long research path to follow in order to significantly benefit from the field of ontologies and semantic technologies. It is worth highlighting that, the exploitation of knowledge from natural language

resources has been the focus of several research initiatives in numerous domains, but—to the best of our knowledge—researchers in the networking field have not explored this path yet. In this paper, we follow this path, by targeting a semantic-based approach that can automatically build specialized forms of knowledge from the information already available in routers' CLI.

The remainder of this paper is organized as follows. Section 2 introduces our semantic approach toward the abstraction of device configuration, and describes the architecture and fundamentals of our OBIE System. Section 3 presents the experimental results carried out over the configuration spaces of two different routers that are widely used in the networking community. Finally, Section 4 concludes the paper.

2 Approach

In this section, we describe an Ontology-Based Information Extraction system for the router configuration domain. Our central hypothesis is that configuration knowledge can be acquired from CLIs, by exploiting two distinctive features, namely, *(i)* the information provided in *"help"* descriptors, and *(ii)* the knowledge inherent to the hierarchical arrangement of commands. In light of this, we developed both a router domain ontology, and an ontology-based learning approach for information extraction (IE) from CLIs. Note that, herein, we focus on the router configuration domain, but the fundamentals of our approach can be extended to other domains wherein configurations rely on the utilization of CLIs (e.g., medical equipment, printer stations, etc).

The general architecture of our OBIE system is depicted in Fig. 1. Notice that, there are two inputs, namely, *(i)* the routers' CLI—as natively provided by

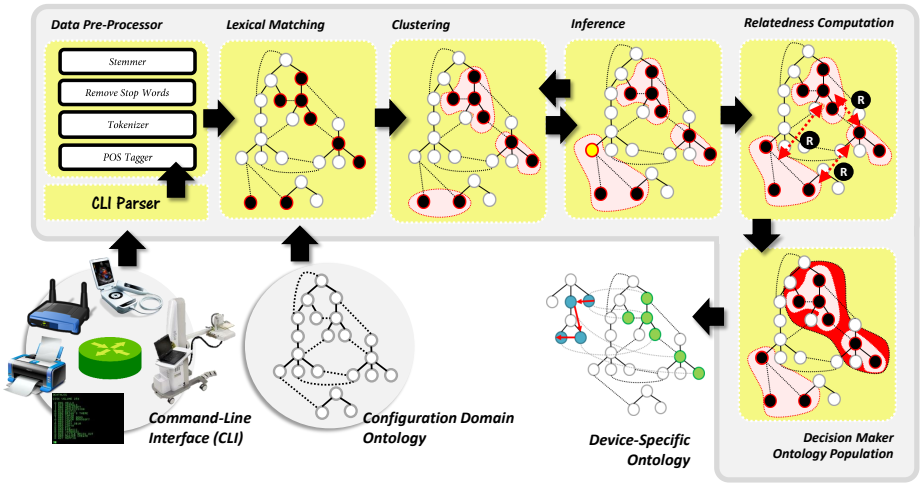

Fig. 1. General architecture of our OBIE system for device command instantiation from Command-Line Interfaces (CLIs).

device vendors—and *(ii)* the router configuration domain ontology. The output of our system is a device-specific ontology which is the result of populating the domain-ontology with instances of commands and variables for a particular device. In other words, a device-specific ontology provides the configuration knowledge (commands and variables) for a concrete network element. These device-specific ontologies are further stored in a database to enable future potential applications (e.g., outsource of configuration tasks to third-party systems). We shall first describe the ontology modeling phase, and then provide details on our OBIE process, which comprises a multi-stage algorithm for the semantic-based instantiation of commands.

2.1 ORCONF: the Ontology for the Router Configuration Domain

We developed and implemented ORCONF, the **O**ntology for the **R**outer **CONF**-iguration domain, motivated by the fact that the knowledge expressed in routers' CLIs—disregarding proprietary technologies—mostly corresponds to protocols and technologies that are broadly accepted by practitioners in the field. On this basis, it is not *"what"* CLIs provide what mostly concerns network administrators, but instead, *"how"* they provide it (i.e., the terminologies and corresponding semantics), a fact which, after all, determines the usability of the CLI.

The ontology modeling stage relies on the knowledge provided by networking experts in addition to that extracted from configuration manuals and textbooks. The design of an ontology is closely related to the ultimate use of the model. In our approach, ORCONF constitutes a valuable resource as we make use of this knowledge to guide the extraction of configuration information from the CLI. In light of this, we have defined two distinct layers (cf., Fig. 2), namely, the router *resource layer* and the router *operation layer*. The former defines all physical and virtual resources of the routing space (e.g., port, interface, domain-name, etc.), and integrates a domain lexicon that will aid in the IE. The latter defines the set of atomic operations (e.g., delete traffic filter, set bandwidth, etc.) that can be performed over resource(s) of the previous layer. With this in mind, concepts within the *operation layer* are semantically associated to concepts in the *resource layer* through the non-hierarchical property "to configure" (cf., Fig. 2). In short, a router *resource* represents a component that can be supplied or consumed in the *configuration*. Both layers of the domain ontology include numerous taxonomies, which are used to indicate all hierarchical relations. The intuition of building the ontology in two separate layers lies on the fundamentals of our learning approach, wherein we first, identify resources and verb phrases from the CLI help—i.e., we identify routing concepts from the *resource layer*—and second, we derive the semantics of the complete sequence of commands from the set of identified resources—i.e., we determine the corresponding configuration action from the *operation layer*. ORCONF was formally defined using the Ontology Web Language (OWL) and built with Protégé. Moreover, we used the Protégé API to access, create and manipulate ontology resources from the OBIE process.

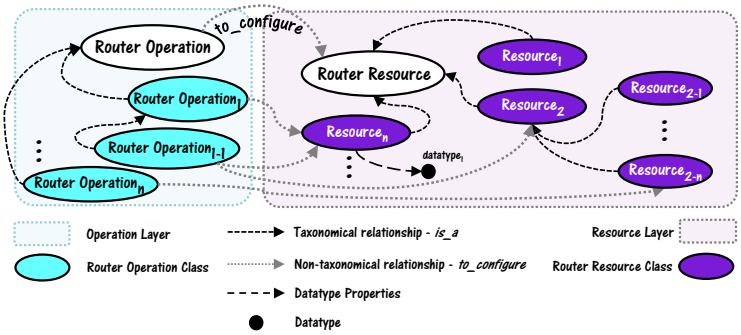

Fig. 2. Layered structure of the Domain-Ontology (ORCONF)—further details including full access to the ontology can be provided after the peer-review process.

2.2 OBIE Process from the Command-Line Interface

In this section, we describe our ontology-based approach for IE from CLIs. The final goal is to semantically instantiate CLI commands, and automatically generate device-specific versions of the domain ontology. Next, we will provide details on each stage of our semantic approach (cf., Fig. 1).

CLI Parser: The CLI Parser has a two-fold functionality, *(i)* to identify the structural components of the CLI—i.e., distinguish between commands, variables and help descriptors—and *(ii)* to scan the hierarchy from top to bottom in order to build all executable configuration statements—i.e, valid sequences of commands and variables that combined represent a configuration operation. For example, the sequence: $set \rightarrow system \rightarrow host\_name \rightarrow \langle name \rangle$, represents a valid statement for the configuration of a router's name. Due to the simplicity of CLIs structure, these tasks are somewhat straight-forward, but still key to our solution. Consider the following—in the context of our approach—the information included in "help" descriptors is not target of instantiation. However, they represent valuable sources of information for determining the semantics of *commands* or *variables*—which are ultimately our goal of instantiation. In practice, the syntax and semantics of *commands* and *variables* are prone to customization, as vendors struggle to distinguish from others. For this reason, *help descriptors* are fundamental for setting common foundations, which guide network administrators on the interpretation and use of the CLI. Note that, a command's name per se is neither sufficient nor determinant for extracting the semantics. Even if the help descriptors do not explicitly provide a common reference for disambiguation, the context can aid in the inference of a term's sense—this feature will be further exploited in our learning approach.

Data Pre-Processing: This stage combines shallow Natural Language Processing (NLP) tools for basic data preprocessing. The first NLP resource applied to our input stream is the Part-Of-Speech (POS) Tagger, aimed to identify verbs. The motivation for this stems from the following: (i) information given in CLIs is concise, hence, there is no need for in-depth POS analysis; (ii) CLIs lack of verbosity and proper grammar, i.e., descriptions are barely restricted to verbs and resources; (iii) the number of actions (verbs) are finite and well-known in the do-

main, e.g., “delete”, “add”, “set”, “show”, etc. For this reason, verb identification in an early stage can limit the scope of the semantic search. In our implementation, we used the Stanford Log-linear POS Tagger. Next, we separate data into tokens, remove stop words, and reduce inflectional forms of a word to its common base form for further processing. Notice that, in our approach—as done by many search engines—words with the same stem are treated as synonyms to increase the probability of hits when performing lexical matching.

Lexical Matching: We rely on lexical matching for initial identification of routing entities from the CLI. To this end, we automatically build a gazetteer (an entity dictionary) from the knowledge provided in ORCONF. From now on we will consider the ontology as a semantic graph—with nodes representing concepts and edges relations between concepts—the outcome of this stage is a set of “activated” (i.e., highlighted) nodes as for every identified routing entity in the CLI. The notion of “activated” nodes is depicted in Fig. 1. Our strategy admits both partial and exact matching. In the case for which the matching keywords are the same for several gazetteer entries, our approach keeps the most general concepts, i.e., we generalize in the absence of information. Moreover, if a concept is not identified in the stage of lexical matching (not even a generalized form of the same), mainly because of the terminology used by vendors, further stages can infer this knowledge based on context information. For example, if we identify in contiguous levels the concepts $\langle ip-address \rangle$ and $\langle bandwidth \rangle$, we can decide upon activating the $\langle interface \rangle$ concept, based on the notion that, subsequent levels specify properties that derive from this general concept.

Clustering and Inference: In this stage directly related concepts for adjacent levels of the CLI (i.e., contiguous “activated” nodes in the semantic graph) are grouped into clusters (cf., Fig. 3). Notice that, the notion of concepts being part of semantic clusters is based on the premise that, commands are arranged in the hierarchy by association—i.e., commands become specific down in the tree structure. Accordingly, concepts in adjacent levels are expected to be semantically related to a certain extent. Notice that, the degree to which concepts are related depends on the granularity of the CLI—which differs for every configuration space. For this reason, clustering is not sufficient and we require a means to measure the degree of semantic relatedness.

Furthermore, we perform basic semantic inference with a two-fold purpose. First, to reason over equivalent axioms of the ontology, i.e., activate an entity if a set of conditions is fulfilled for a *defined* class and further allocate it within a cluster. For example, if the entity $\langle route \rangle$ was not activated by lexical matching, but still we identify $\langle hop \rangle$, $\langle source \rangle$ and $\langle destination \rangle$ entities, we can infer from the equivalent axioms of our ontology that we are referring to the $\langle route \rangle$ concept. Second, to generalize or specify already “activated” concepts based on context information. For example, if the domain entity $\langle interface \rangle$ was activated for a certain level, and further IE identifies “exclusive” properties of a child concept, we infer that, the most specific concept is most likely to be the asserted entity (e.g., a specific interface type).

Relatedness Computation: The semantic relatedness of a set of clusters $\{\mathcal{C}\}$ is as a means to determine the degree to which candidate entities are associated by meaning. As mentioned earlier, due to the hierarchical nature of CLIs, we assume that maximum interrelated concepts are most likely representative of an executable sequence of commands. To exemplify this, let us consider the example shown in Fig. 3, where the entities *"OSPF_Area"* $\in \mathcal{C}_B$ and *"Local_Area_Network"* $\in \mathcal{C}_C$ are two possible candidates for the CLI term "area". Observe that both entities and the clusters to which they belong to are disjoint, as only one ontological class can represent the semantics of the term. From a lexical perspective, the succinctness of the CLI is what generates ambiguity between both concepts. However, based on the contextual background, the entity *"OSPF_Area"* seems to be a better candidate, as it is semantically related to concepts identified in adjacent levels (i.e., concepts $\in \mathcal{C}_A$). Thus, our relatedness measure must contribute to the problem of word sense disambiguation, i.e., picking the most suitable sense of the word and constraining the potential interpretation of terms in our system. To this end, the relatedness measure $\mathcal{R}$ that we introduce next quantifies the degree to which a set of clusters $\{\mathcal{C}\}$ are semantically related.

Let $G(\mathcal{C}, R)$ be a directed graph, where the vertex $\mathcal{C}$ represents a cluster $\in G$, and the edge R represents a relationship among two adjacent clusters. Let $G_k \subseteq G$ represent a connected subgraph of G, and $\mathcal{C}_k^i$ be the i^{th} cluster $\in G_k$. As depicted in Fig. 4, the ontological class l within $\mathcal{C}_k^i$ shall be denoted as c_k^{il}. Equation (1) shows the relatedness measure $\mathcal{R}$ used in our model, which consists of two components: *(i)* the connection density $d(G_k)$, and *(ii)* the maximum information content coverage $\mathcal{I}(G_k)$. The density component $d(G_k)$ is shown in (2), and it is basically a measure of the semantic connectivity of graph G_k. It is computed as the relation between the number of "activated" entities along the shortest path between any pair of clusters in graph G_k, and the total number of connections (i.e., the number of relations between entities) in those shortest paths. More specifically, let $\mathcal{C}_k^i$ and $\mathcal{C}_k^j$ be a pair of clusters in G_k, and let $\mathcal{P}(c_k^{il}, c_k^{jp})$ denote a path between a pair of entities $c_k^{il} \in \mathcal{C}_k^i$, and $c_k^{jp} \in \mathcal{C}_k^j$. The shortest path between two clusters is defined as $\mathcal{SP}(\mathcal{C}_k^i, \mathcal{C}_k^j) = \min \mathcal{P}(c_k^{il}, c_k^{jp})$, $\forall$ c_k^{il}, c_k^{jp} in clusters $\mathcal{C}_k^i$, and $\mathcal{C}_k^j$, respectively. To illustrate this, consider the paths between the clusters $\mathcal{C}_k^1$ and $\mathcal{C}_k^2$ as shown in Fig. 4. In this case, the shortest path between any pair of entities (c_k^{1l}, c_k^{2p}), i.e., paths with source in cluster $\mathcal{C}_k^1$

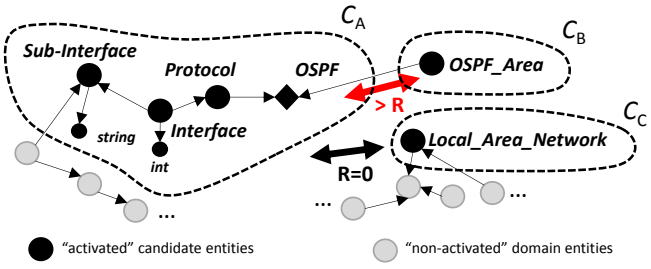

Fig. 3. The rationale behind the quantification of the Semantic Relatedness.

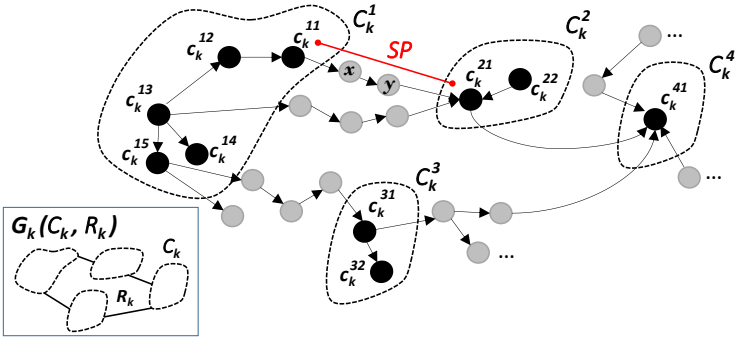

Fig. 4. An example of Semantic Relatedness.

and termination in $\mathcal{C}_k^2$, or vice-versa, is $\mathcal{SP}(\mathcal{C}_k^1, \mathcal{C}_k^2) = [(c_k^{11}, x), (x, y), (y, c_k^{21})]$. Now, let the function $\mathcal{A}(\mathcal{P})$ return the total number of "activated" entities (i.e., the ontological classes) in path $\mathcal{P}$. In our example, $\mathcal{A}(\mathcal{SP}(\mathcal{C}_k^1, \mathcal{C}_k^2)) = 2$, which are c_k^{11} and c_k^{21}. Observe that the source of a path $\mathcal{P}$ is always an "activated" entity—recall that the clusters are composed of activated entities only—hence the number of "active connections" along a path $\mathcal{P}$ is $(\mathcal{A}(\mathcal{P}) - 1)$ (cf. (2)). Similarly, the function $\mathcal{H}(\mathcal{P})$ in the denominator of (2) returns the total number of hops in path $\mathcal{P}$. For instance, in the example shown in Fig. 4, $\mathcal{H}(\mathcal{SP}(\mathcal{C}_k^1, \mathcal{C}_k^2)) = 3$. Observe that when the clusters $\mathcal{C}_k^i$ and $\mathcal{C}_k^j$ are not adjacent, the shortest path will traverse other clusters. Hence, in a connected graph, the number of activated entities always satisfies $\mathcal{A}(\mathcal{SP}(\mathcal{C}_k^i, \mathcal{C}_k^j)) \geq 2$.

The second term of the relatedness measure $\mathcal{R}$ is the Information Content $\mathcal{I}(G_k)$, which is shown in (3). This term represents a measure of the knowledge covered by the entities (i.e., the ontological classes) in their corresponding clusters. Let t_k^{il} denote the number of CLI terms that "triggered" the activation of an entity $c_k^{il} \in \mathcal{C}_k^i$ in the semantic graph G_k. Observe that in (3), the contribution of an entity c_k^{il} to the domain knowledge is weighted by two factors, m_k^{il}, and o_k^{il}. Let, m_k^{il} be the "matching" factor of the l^{th} entity, which is 1 for entities identified by perfect match; otherwise, its value is chosen as $\frac{1}{(e+1)}$, with e the total number of entities identified for the same CLI term. In other words, in case of partial match, the weighting factor m_k^{il} represents the probability of being any of the e entities identified for the same CLI term—including none of them (+1).

$$\max_k \quad \mathcal{R}(G_k) = d(G_k) \cdot \mathcal{I}(G_k) \tag{1}$$

$$d(G_k) = \frac{\sum_{i=1}^{|\mathcal{C}_k|-1} \sum_{j=i+1}^{|\mathcal{C}_k|} [\mathcal{A}(\mathcal{SP}(\mathcal{C}_k^i, \mathcal{C}_k^j)) - 1]}{\sum_{i=1}^{|\mathcal{C}_k|-1} \sum_{j=i+1}^{|\mathcal{C}_k|} \mathcal{H}(\mathcal{SP}(\mathcal{C}_k^i, \mathcal{C}_k^j))} \quad \leq 1 \tag{2}$$

$$\mathcal{I}(G_k) = \sum_{i=1}^{|\mathcal{C}_k|} \sum_{l=1}^{|c_k^{il}|} t_k^{il}.m_k^{il}.o_k^{il} \tag{3}$$

In the example shown in Fig. 3, e = 2 for the entities triggered by the term "area", with equal probability from the information content perspective of being *"OSPF_Area"*, *"Local_Area_Network"*, or none of them. Moreover, the other weighting factor, i.e., o_k^{il}, is a measure of the "occurrence" of a candidate entity in the CLI, over the total number of occurrences of its exclusive disjoint entities. This measure is computed during the IE process at the lexical matching stage.

Finally, observe that we compute the semantic relatedness $\forall\ G_k \subseteq G$, that is, over the total number of connected cluster subgraphs of G. As indicated in (1), the relatedness measure that we chose is the maximum obtained $\forall\ G_k$. It is worth mentioning that, even though at first sight our model might look a bit intricate, its computation is actually quite straightforward. The nature and hierarchical structure of CLIs typically yields a small number of interrelated clusters, and more importantly, the system outlined in Fig. 1 operates in offline mode, so the only and fundamental goal is the accuracy of the OBIE process. Indeed, the results that we present in the next Section confirm the strengths of our model and the approach proposed in this paper. Also observe that, although the ontology and some of the descriptions made in this Section are application-specific—i.e., they consider particular features of CLI environments for routers—the essence of our model can be generalized and applied to other contexts, especially, those that rely on hierarchical CLIs for device configuration.

3 Evaluation

In this section, we present the experimental results of our semantic approach, when carried out over the configuration spaces of two widely used routers. In order to ensure heterogeneity, we have selected both a commercial and an open-source router environment, namely, *Juniper* and *Quagga*. Observe that the number of commands available in a router can be significantly large, but the ones that are commonly used in practice represent a relatively small set. For this reason, our evaluations are centered on the set of features that are widely used in practice. In order to define a relevant set of configuration features, we thoroughly selected dissimilar branches of the CLI hierarchy, in an effort to encompass a broad set of functionalities. We used a set of approximately 150 commands (over 70 configuration statements) including not only protocol or technology-dependent settings but also router-related functions—e.g., administrative configurations. We consider that this a significant set as typical routers' configurations contain around 100 configuration statements [12], taking into account that many settings are recurrent, e.g., the configuration of interfaces. Moreover, notice that there is no semantic approach following our same line of research, so without comparison possibilities, we can only evaluate the overall performance of our instantiation process.

Table 1. Performance Results of our OBIE Process.

	Bag						*Per-Level*					
	Traditional			*Augmented*			*Traditional*			*Augmented*		
	P	***R***	***F1***	***AP***	***AR***	***AF1***	***P***	***R***	***F1***	***AP***	***AR***	***AF1***
Juniper	89%	88%	88%	92%	92%	**92%**	78%	87%	82%	81%	91%	**86%**
Quagga	91%	88%	89%	94%	91%	**92%**	81%	81%	81%	85%	85%	**85%**
Overall	90%	88%	89%	**93%**	**91%**	**92%**	80%	84%	82%	83%	87%	**85%**

Typical measures to assess the performance of IE systems are inadequate when using ontologies [13]. For this reason, we rely on the Balanced Distance Metric (BDM) [14]—a cost-based component that measures the degree of correctness according to the ontological structure. Notice that, the BDM per se does not provide a means for evaluation. To this end, we measure the Augmented Precision (AP), Augmented Recall (AR), and Augmented F_1-measure (AF_1), as introduced in [14]. The evaluation results are depicted in Table 1. We report percentage values of traditional and augmented Precision, Recall and F_1-Measure for both configuration spaces separately. Observe that, we have considered two criteria for reporting our results, namely, per-level and "Bag". These criteria are just for evaluation purposes. The former, strictly considers an entity valid if it was identified in the level that it actually corresponds to in the configuration statement, while the latter handles all identified entities regardless of the level to which the entity is attributed. Overall, the results for the "Bag" criteria are better than those reported per-level. This is mainly because the information within a level is not always sufficient to accurately determine its semantics. Instead, the inference stage further derives knowledge from the information obtained from subsequent levels. Moreover, evaluation results are also affected by assertions made by vendor's, which are not strictly aligned to the domain knowledge. The apparent miss-classification of entities due to inconsistencies of this nature are actually a degree of correctness of our learning approach.

In a nutshell, our system achieved an overall augmented F_1-Measure of **92**%. Notice that, augmented measures are in all cases greater than traditional ones, basically because our ontology-based approach has the ability to generalize or specify a concept according to contextual knowledge, and the BDM metric adds a cost-based component to this matched entity.

4 Conclusions

Overall, our experimental results show that the semantic approach proposed in this paper opens a promising line of research in the router configuration domain, and that the knowledge provided by CLIs is a valuable source for Information Extraction (IE). The results indicate that we achieve remarkable precision and recall for both case studies, but still, there is significant room for improvement.

Acknowledgment

This work was supported in part by the Spanish Ministry of Science and Innovation under contract TEC2012-34682.

References

1. J. Schonwalder, M. Bjorklund, and P. Shafer. Network configuration management using NETCONF and YANG. *Communications Magazine, IEEE*, 48(9):166–173, Sept 2010.
2. F. Le, S. Lee, T. Wong, H.S. Kim, and D. Newcomb. Detecting Network-Wide and Router-Specific Misconfigurations Through Data Mining. *Networking, IEEE/ACM Transactions on*, 17(1):66–79, Feb 2009.
3. A. Pras, J. Schonwalder, M. Burgess, O. Festor, G.M. Perez, R. Stadler, and B. Stiller. Key Research Challenges in Network Management. *Communications Magazine, IEEE*, 45(10):104–110, 2007.
4. R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. Network Configuration Protocol (NETCONF). RFC 6241, IETF, June 2011.
5. A Martinez, M. Yannuzzi, V. López, D. López, W. Ramírez, R. Serral-Gracia, X. Masip-Bruin, M. Maciejewski, and J. Altmann. Network Management Challenges and Trends in Multi-Layer and Multi-Vendor Settings for Carrier-Grade Networks. *Communications Surveys Tutorials, IEEE*, (99):1, 2014.
6. J. López, V. Villagrá, and J. Berrocal. Applying the web ontology language to management information definitions. *Communications Magazine, IEEE*, 42(7):68–74, 2004.
7. H. Xu and D. Xiao. A Common Ontology-Based Intelligent Configuration Management Model for IP Network Devices. In *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on*, volume 1, pages 385–388, Aug 2006.
8. H. Xu and D. Xiao. Applying semantic web services to automate network management. In *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pages 461–466, May 2007.
9. A. Wong, P. Ray, N. Parameswaran, and J. Strassner. Ontology mapping for the interoperability problem in network management. *Selected Areas in Communications, IEEE Journal on*, 23(10):2058–2068, 2005.
10. J. López de Vergara, A. Guerrero, V. Villagrá, and J. Berrocal. Ontology-Based Network Management: Study Cases and Lessons Learned. *Journal of Network and Systems Management*, 17(3):234–254, 2009.
11. F. Colace and M. De Santo. A Network Management System Based on Ontology and Slow Intelligence System. *International Journal of Smart Home*, 5(3):25, July 2011.
12. CISCO Sample Configuration. `http://www.cisco.com/c/en/us/td/docs/routers/access/1800/1801/software/configuration/guide/scg/sampconf.html` Online; accessed Oct-2014.
13. D. Maynard, W. Peters, and Y. Li. Metrics for Evaluation of Ontology-based Information. In *In WWW 2006 Workshop on Evaluation of Ontologies for the Web*, May 2006.
14. D. Maynard, W. Peters, and Y. Li. Evaluating evaluation metrics for ontology-based applications: Infinite reflection. In *LREC*. European Language Resources Association, 2008.