International Journal of Intelligence Science, 2012, 2, 190-202
http://dx.doi.org/10.4236/ijis.2012.224025 Published Online October 2012 (http://www.SciRP.org/journal/ijis)

Sequence Validation Based Extraction of Named High Cardinality Entities

Khamisi Kalegele[1], Hideyuki Takahashi[1,2], Kazuto Sasai[1,2], Gen Kitagata[1,2], Tetsuo Kinoshita[1,2]

[1]Graduate School of Information Sciences, Tohoku University, Sendai, Japan
[2]Research Institute of Electrical Communication, Tohoku University, Sendai, Japan
Email: kalegs@k.riec.tohoku.ac.jp, kazuto@fir.riec.tohoku.ac.jp, minatsu@fir.riec.tohoku.ac.jp, hideyuki@riec.tohoku.ac.jp, kino@riec.tohoku.ac.jp

Received May 21, 2012; revised July 26, 2012; accepted August 6, 2012

ABSTRACT

One of the most useful Information Extraction (IE) solutions to Web information harnessing is Named Entity Recognition (NER). Hand-coded rule methods are still the best performers. These methods and statistical methods exploit Natural Language Processing (NLP) features and characteristics (e.g. Capitalization) to extract Named Entities (NE) like personal and company names. For entities with multiple sub-entities of higher cardinality (e.g. linux command, citation) and which are non-speech, these systems fail to deliver efficiently. Promising Machine Learning (ML) methods would require large amounts of training examples which are impossible to manually produce. We call these entities Named High Cardinality Entities (NHCEs). We propose a sequence validation based approach for the extraction and validation of NHCEs. In the approach, sub-entities of NHCE candidates are statistically and structurally characterized during top-down annotation process and guided to transformation into either value types (v-type) or user-defined types (u-type) using a ML model. Treated as sequences of sub-entities, NHCE candidates with transformed sub-entities are then validated (and subsequently labeled) using a series of validation operators. We present a case study to demonstrate the approach and show how it helps to bridge the gap between IE and Intelligent Systems (IS) through the use of transformed sub-entities in supervised learning.

Keywords: Entity Recognition; Supervised Learning; Sequence Validation; Intelligent Systems; Text Mining

1. Introduction

Web aggregated content has become so popular and useful that it is considered indispensable. While utilizing the information that Web content provides, users are also so busy populating new data into the Web. It is a cycle of sharing, searching, browsing, to mention some, that is undertaken by so many people in vast domains. Like many other stake holders in their respective disciplines of this cycle, scientists and IT professionals are playing a vital role in facilitating easy access to this information through many a field like Information Extraction (IE) and text mining. IE, particularly, involves extraction of specific and exact pieces of information *extracts* in addition to searching for documents which contain them. Examples of *extracts* for which IE systems have been proposed include citations [1,2], course descriptions [3], news items [4], protein data [5] etc. Within IE systems, one of the most vital solutions is Named Entity Recognition and Classification (NERC). A field which has been extensively researched over the past 20 years.

As described in [6,7], Named Entity (NE) is a term widely used in NPL and was designed such that the expression "Named" aims at restricting the task to only those entities for which one or many rigid designators stand for the referent. For instance, a Japanese national university located in Sendai city, Miyagi prefecture in the Tohoku Region, is referred to as Tohoku University. Over the years, many NE have been studied, from proper names [2,8], email addresses to bioinformatics domain [5].

Successfully approaches to NERC have mostly employed hand-coded pattern rules and statistical methods with varying degrees of automation using Machine Learning (ML) techniques. The basis to these approaches is NLP related features like orthographic features (e.g. capitalization) and parts-of-speech (e.g. nouns). Ultimate reliance to NLP features have seen NERC limited to only those entities which are speech compliant and appear in well formed sentences. As a result, following our observations, most NERC systems have dealt with entities of up to few sub-entities of lower cardinality levels. For instances, with the exception of open sub-entities like

personal names, there are limited instances of a title (e.g. Mr., Mrs., Dr. etc.). This fact, together with mostly fewer number of sub-entities, enables writing of handful hand-coded pattern rules in many successful NERC approaches.

In the ongoing Web-facilitated data deluge (as referred to in [9]), we are seeing diverse efforts of harnessing Web information for which NERC does not suffice as a vital building block to IE. So, many entities are now in need to be recognized, classified and organized, in a way that NE have been, if the full potential of harnessing Web information for use in various domains is to be realized. Only that these many entities are neither speech compliant nor do they appear in well formed sentences. An example in network administration domain would include system commands like *ssh* and *netstat* commands etc. In ML, as another example, *instances* can also be considered an entity because for a particular learning task and environment the structure of an *instance* is fixed, e.g. ARFF format in WEKA [10].

In this paper we refer to the entities which are neither directly speech compliant nor do they appear in well formed sentences as Named High Cardinality Entities (NHCEs) and propose an approach for their extraction. Unlike NE, since NHCE has a fixed format and structure, it is necessary to conduct a validation test after extraction because a NHCE can be structurally correct but invalid. Therefore, our proposed approach does include not only extraction but validation as well. The employed validation method in our proposed approach is sequence based whereby NHCE candidates are regarded as sequences. Validation is conducted in two phases: during top-down extraction (using generic pattern rules) and in post-processing phase (statistical). The approach enables NHCE sub-entities to transform as more observations are made. This transformation is achieved using a ML-based transformation model.

In the past, an increase in online publications gave rise to research interests in classifying and organizing citation *extracts*. As a result, we have seen success in products like citation databases (Citeseer [11]) whose power in facilitating information sharing among researchers have now been unleashed. We believe that progress in recognition and classification of NHCEs will facilitate better organization of many other information entities in the Web like technical forums (technical forum databases), ML training data (training data databases) etc. This will also help in bridging the gap between IE systems and Intelligent Systems (IS) through areas like supervised learning. In this paper, we also present a case study and demonstrate how recognition and classification of NHCEs can help in reducing this gap.

The rest of this paper is organized as follows. We provided an overview to IE systems, and define and discuss NHCE in the next Section 2. Our proposed approach is presented in Section 3 and a case study in Section 4. In Section 5, we discuss related works. Section 6 presents our future research direction and concludes this paper.

2. Information Extraction

2.1. Overview and Our Contributions

IE systems deal with extraction of pieces of information from text based on some predefined concepts. Four factors provide the dimensions for their presentation and discussion. These are source-document type, *extracts* type, involved techniques and automation degree.

1) Source-Document Type: IE System can be designed to deal with either unstructured [1,5] or semi-structured [3,8] or structured [12] or both sources of data. Extraction difficulty easies as document structuredness increases.

2) *Extracts* Type: In most cases, *extracts* are very specific to a system. Types of *extracts* for which systems have been designed in the past include single-entity *extracts* or NE like names (of places, of persons etc.), a pattern (multi-entity *extracts* e.g. publication citation [1, 2]), entities relationship (substance-protein metabolization [5]), and concepts (e.g. document or sentence classification).

3) Techniques: The basic IE systems involve sentence splitting, tokenization, named entity recognition, syntactic parsing and pattern recognition. Variations are with the last three whereby some systems use lookup lists and dictionary/thesaurus [8], some use pattern and logical rules (e.g. using JAPE) and others use ML techniques and ontologies [4]. Non rule-based approaches are also referred to as statistical methods. They are characterized by their approach of using models to label entity tokens. Various models have been used including ordered classification, Hidden Markov Models, HMM [13,14], Conditional Markov Model, CMM [15], Conditional Random Fields, CRF [16] etc. CRF-based are currently regarded as state-of-the-art methods in assigning labels to either tokens or their sequences.

4) Automation: The degree of automation is inline with the used technique. With rule-based techniques, automation is in rule generation. With ML techniques, automation is in model creation for either classification or clustering or for rule generation. The central point in automation has been in how much of labeled documents is needed to initiate an extraction process. In ML-based apporaches, progress has been made from supervised learning [17] to semi-supervised learning [7] to unsupervised learning [18]. Despite tremendous achievements in automation, the most reliable and efficient of systems are still hand-coded ones. Although ML-based approaches have huge potentials, they still face a critical challenge

when it comes to labeling of examples for their consumption.

Commonly, IE systems are designed under NL-dependence context. Few have ventured into NL-independence. In [15], a probabilistic tagging approach for tagging sequences of words constituting NEs, based on HMM and NL-independence, is proposed. This approach, however, considered only NE with few sub-entities of low cardinality like personal and organization names. The task of classifying *extracts* which conform to NHCE has been approached by few researchers [1]. *Extracts* which are commonly dealt with and used for demonstrations are citation *extracts*. To the best of our knowledge, most IE systems which have dealt with *extracts* which conform to NHCE have only focused on a parsing task using already annotated *extracts* [1].

IE field is still facing a good number of challenges. We have already highlighted two of the biggest challenges (labeling of examples and NHCE recognition and classification). Other challenges, as described in [19], are those beyond automation and scalability of systems. These includes domain adaptation and an intriguing problem to demonstrate *extracts*' utility to IS, to mention few.

Our focus in this paper is on recognition and classification of NHCEs. We believe that being able to recognize and classify NHCEs brings in another milestone towards improved utility of *extracts*. Our contributions which are put forward can be expressed from different points of view. One is in leveraging the Web when solving ML problems like classifying or associating items etc. Typically, these tasks require examples which are always painfully produced. For long ML community have longed for data processing tools and approaches which will enable them to exploit the abundant Web content. We believe that the proposal in this paper brings ML community closer to Web content leveraging. Another point of view is in relation to limitation to speech compliant entities. Our work is part of the long term effort of organizing non-speech entities (e.g. source codes, ML training data, network administration forums) in order to improve their sharing and utility. In a conceptual diagram shown in **Figure 1**, the left side shows the current common practice and focus whereby ML and other methods are applied on Web content to further improve efficiency of IE systems in extraction process. The right side gives a picture of our contribution whereby the immediate beneficiary of the various processing, in addition to IE systems, is ML.

2.2. Named High Cardinality Entity

We use the term NHCE to refer to those atomic entities in text which contains sub-entities of high cardinality. Our expression "Named High Cardinality Entity" loosens the expression "Named Entity" by including not so rigid designators which the later require. NHCE is not limited to speech compliant entity atomicity. For instance, "proper names" as NEs can not be worked out outside speech context. NHCEs can be regarded as further loosening of temporal expressions type of NEs like money in order to include other entities like computer systems related atomic entities and ML related entities which are increasingly becoming popular. On one side, NHCE overlaps NE to include such entities like citations and *ssh* commands (computer systems related entities). On the other side, NHCE includes such temporal expressions like training data instances (ML related) which are not covered by NE.

Three aspects in which NHCE differs from NE are sub-entity cardinality, speech compliance and rigidness of designators. These are summarized in **Table 1**. Although not formally stated, researches in NERC have

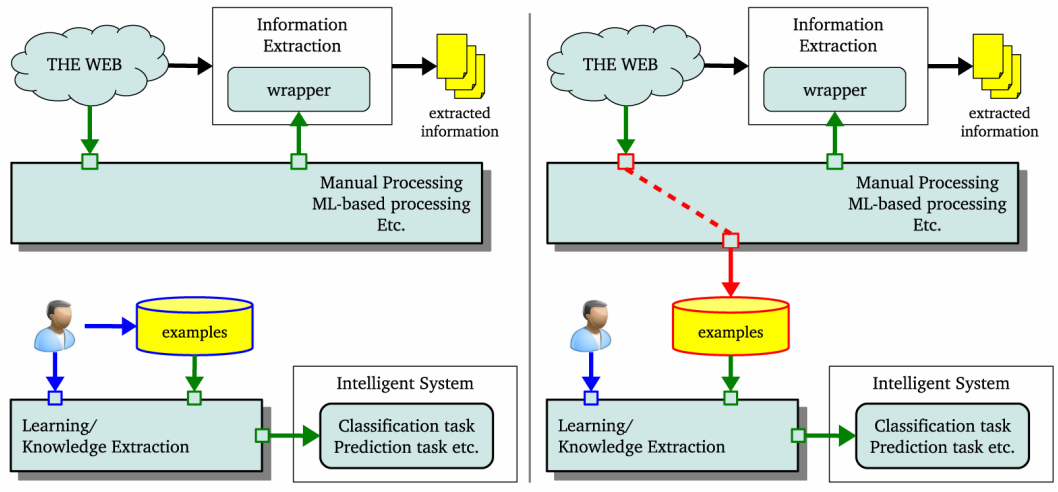


Figure 1. Conceptual diagram: Bridging the gap between IE and IS.

Table 1. How NHCE differs from NE.

Aspect	NE	NHCE
Sub-entity cardinality	Mostly low	Any level
Designator rigidness	Rigid	Any rigidness
Speech compliance	Compliant	Non-compliant

mostly dealt with relatively lower cardinality entities, e.g. the "enamex" (names of persons, locations etc.) and "timex" (date, time, money etc.). All these are speech compliant and constitute very rigid designators.

3. The Proposed Approach

3.1. Problem Formulation

Suppose an $n-$entity NHCE whereby each sub-entity is either a token or a NE referred to as a type. An entity field $f_i : 0 < i \le n$ within that NHCE can either be unoccupied (missing sub-entity) or occupied by any of the $\|T^i\|$ types ($t^i \in T^i$) which are valid for that field. There are, therefore, $\prod_{i=1}^{n}\left(T^i + \phi_i\right)$ possible sequences of this $n-$entity NHCE. ϕ_i is either one or zero based on whether entity n_i can be missing ($\phi_i = 1$) or not ($\phi_i = 0$).

For instance, a linux command can be considered as a 6-entity NHCE taking the format shown in **Table 2**. In an *ssh* command, valid type for ID field is only "ssh", *i.e.* 1-cardinality. For OPTION field, there are 25 valid types (1, 2, "A", ···), *i.e.* 25-cardinality. INPUT field is 15-cardinality (ip, path, ···). ARGUMENT field is 2-cardinality (host and user). For COMMAND field, there are almost as many as all linux commands as valid types (more than 400). DESTINATION field is 0-cardinality. For simplicity, without considering COMMAND field, there are therefore at least 750 possible sequences of *ssh* command some of which are invalid. For example, in *ssh* command sequences above, although the second sequence is structurally correct, it is not a valid *ssh* command.

This paper addresses extraction and validation of *n*-entity NHCE so that its *extracts* can be used in solving ML problem by providing labeled training examples.

3.2. Overview

We propose a sequence validation based approach for the extraction and validation of NHCEs. Our approach is a top-down one in which generic pattern rules are used to extract text fragments that constitute the desired NHCE *extracts*. These text fragments are parsed into sub-entities and matched with pre-defined template sequences which are statistically scored using sequence frequency-inverse page frequency, *sf-ipf* (explained in Section 3.5). The sub-entities and the matched template sequences are then passed through a series of three categories of validation operators in which sub-entity sequences (*extract* sequences) are also labeled (valid or invalid) as described in Section 3.5. By characterizing sequence fields, the approach enables sub-entity occurrences to be either extended into value types (v-types) or transformed into either basic structural types (b-types) or user-defined types (u-types) using a ML-based transformation model. Finally, sub-entity transformed and labeled *extract* sequences of the desired NHCE are used to generate output for use in solving ML problems like validation or classification of the NHCE under consideration (can also be for use in further NHCE extraction). A schematic which summarizes the involved processes (template matching, entity transformation and validation) is shown in **Figure 2**. We explain the motivations and requirements which led to this approach designed in Section 3.3. It is inline with these requirements that the rest of the approach is presented.

Table 2. Example of *n*-entity format.

Format ID OPTION INPUT ARGUMENT COMMAND DESTINATION
Examples: ssh ? -R 99:lhost:22 82:lhost:22 ? ? ssh -x -L 34:lhost:66 server.com sleep 10 ? mv ? ? file1 ? file2
Key: ? represents a missing entity

3.3. Requirements and Motivations

1) Annotation and Parsing: The objective of leveraging Web content necessitates an annotation process for the identification of text fragments which constitute the desired NHCE and a wrapper for parsing those fragments into sub-entities. Instead of manual supply of documents (as text source or for training), an iterative and interactive extraction can allow us to leverage Web content and save human capital.

2) Validation of Sub-entity Sequences: Since knowledge about the *extracts* is initially not exact enough, a natural mechanism is to consider all sequences of sub-entities which constitute the desired NHCE *extracts*. To validate or invalidate an extracted text fragment and subsequently label it, a variety of means (e.g. statistical, expert input) is necessary if one is to expect comprehensive results.

3) Sub-entity Transformation: To cope with the evolving and transforming nature of Web information, sub-entities must be allowed and guided to transformation in order to accommodate new occurrences as more NHCE *extracts* are seen.

3.4. Annotation and Parsing

We adopt a top-down extraction process for annotating

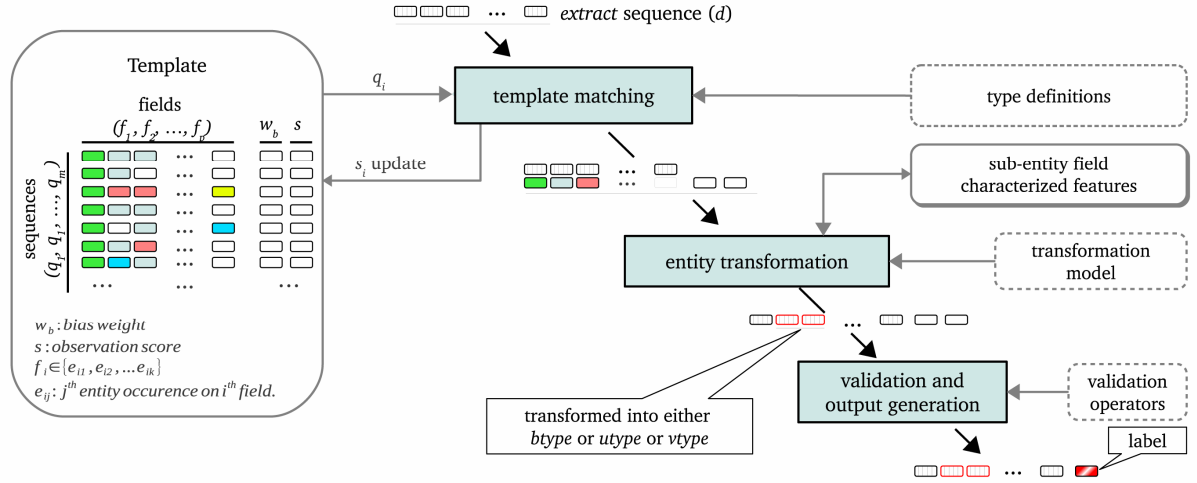


Figure 2. NHCE extraction process: A schematic diagram showing template matching, entity transformation and sequence validation.

and parsing *extracts*. Generic rules are used to extract text fragments which constitute the desired *extract*. At this stage, semantical and structural relationships are ignored. For instance, when *ssh* commands are desired, *ssh -R* 5555: localhost: 3389 and *ssh -l* 5555: localhost: 3389 will all be extracted because the relationships between *-R*, *-l* and 5555: localhost: 3389 are ignored. Shown in **Figure 3** is the employed generic rules format in JAPE [20].

In the generic rule format, the terms ent_i are logical expressions for matching sub-entities. Under JAPE, these can be organized into JAPE macros. A combination of sub-entities forming up a sequence is bound to a label $l_i : 0 < i \le n$ which is used as a reference to the matched text fragment during post-processing after the rule has fired. For each label l_i, there are intermediate labels $l_{ij} : 0 < j < i$ which are used for parsing the matched text fragment. This annotation approach, therefore, also serves as a parser for the desired NHCE *extract*.

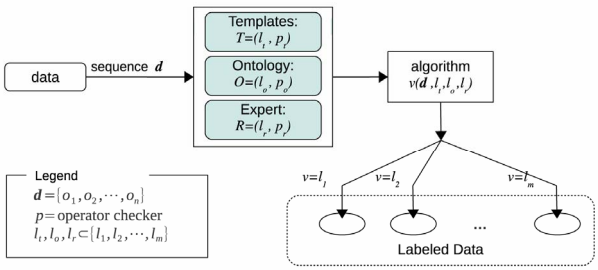
We use rules in this initial stage and conduct statistical processing in later semi-automated stages because rules are easier to interpret, develop and deploy while statistical processing brings in robustness [21] to noise for unstructured data.

3.5. Sequence Validation

Extract sequences are then validated using template sequences and a set of two other categories of operators (ontological and direct expert input) as shown in **Figure 4(a)**. Operators $T = (l_t, p_t)$, $O = (l_o, p_o)$ and $R = (l_r, p_r)$ are such that a sequence d_i (of objects $o_k : 0 < k \le n$) for which either $p_t(d_i) = \text{true}$ or $p_o(d_i) = \text{true}$ or $p_r(d_i) = \text{true}$ is labeled either l_t or l_o or l_r respectively. Operators serve to bring flexibility into validation process and enables hands-on by an expert. Ontological and expert operators are case and domain dependent and therefore are described further with a case study in Section 4. The different validation results (l_t, l_o and l_r) are then combined by using **Algorithm 1**.

```
Rule: r1
 ent1: l1 |
(ent1: l21  ent2): l2 |
(ent1: l31  ent2: l32  ent3): l3 |
 ...
(ent1: ln1  ent2: ln2  ... ... entn): ln
  → {post rule fire processing}
```

Figure 3. Pattern rule format.



(a) Validators

Specifications	Examples
field; type; repetition; missing	**field; type; repetition; missing**
f_1;f_1-type;int;binary	ID;cmdid;1;false
f_2;f_2-type;int;binary	OPTION;option;1;true
f_3;f_3-type;int;binary	INPUT;input;2;true
f_4;f_4-type;int;binary	ARGUMENT;argument;2;true
f_4;f_5-type;int;binary	COMMAND;command;1;true
f_6;f_6-type;int;binary	DESTINATION;destination;1;true

(b) Template Specification

Figure 4. Sequence validation.

Algorithm 1. Aggregation of validation results.

$l \leftarrow l_t$
if $l_o \neq null$ **then**
  $l \leftarrow l_o$
end if
if $l_r \neq null$ **then**
  $l \leftarrow l_r$
end if

Templates, as shown in **Figure 3**, are made up of all possible sequences of the desired NHCE and are in RDF format. Sequences in a template can be manually written or auto-generated using specifications shown in **Figure 4(b)**. They are defined in terms of fields and their types with either missing occupants or entities, as defined in a definition files to which the types act as pointers, with a specified level of allowed repetition. For instance, with *ssh* commands as desired NHCE *extracts*, the first field (ID) can only have *cmdid* as its occupant with allowed repetition 1. The third field (INPUT), however, can be occupied by either of the following; *null*, *input*, *input input* etc. *i.e*. It is possible to have multiple inputs occupying an INPUT field (allowed repetition 2).

Each sequence in a template is assigned an observation score called sequence frequency-inverse page frequency, *sf-ipf* (Equation (1)) which shows how significant it is. It is denoted as *s* in **Figure 3**. In Equation (1), $\|P\|$ is the total number of pages seen. For a document-based source like the Web, $\|P\|$ is the total number of documents. $sf(q,t)$ is the sequence count in a template *t*, (has value 1 in this proposed approach). The definition of *sf-ipf* is given by Definition 6. *sf-ipf* is analogous to *tf-idf* ([22]) which is a well known numeric statistic used in Information Retrieval and Text Mining for showing how important a word is in a document. In templates, there is also a provisioning for defining a bias weight based on the source of the *extracts* because for some sources (e.g. the Web), an *extract* can have a statistical advantage over others due to factors other than its information richness (e.g. social, commercial factors). Bias weights are specified using an approximated *extracts* distribution.

$$\begin{aligned} &sf-ipf(q,t,P) \\ &= sf(q,t) \times \log \|P\| \|1\| + \{p \in P : t \in p\} \end{aligned} \quad (1)$$

sf-ipf is a numerical statistical weight on a sequence (*q*) of types that reflects the significance of that sequence across all sequences (in a template *t*) which defines an *extract*.

3.5.1. Approximated Distribution

When parsing and matching an *n*-entity NHCE *extract*, we use *n*-bit binary flags to represent its matches whereby there are $2^{n-1}-1$ different flags. In *ssh* command example, where *extracts* are 6-entity as described in Section 2, there are $2^5-1=31$ (100000 to 111111) different flags. Since it is a complex problem to model how sequences are distributed across a heterogeneous information source (e.g. the Web) [16], our approach involves approximating this distribution using a minimum of information. We use the flag which corresponds to the commonly used *extract* occurrence. We approximate a distribution from this flag value (as the mean), using lognormal distribution (Equation (2)) under the following two assumptions.

Assumption 1: *Extract* occurrences are skewed distributed around a commonly used one. e.g. *ssh-X user@host*

Assumption 2: The frequency of occurrence decreases with matched sub-entities. For instance, it is more likely to encounter *ssh user* than *ssh-F file.txt user@host*.

$$\begin{aligned} D(x) = exp\left(-(\ln x)^2 \Big/ \left(2\times\sigma^2\right)\right)x \\ \times\sigma\times\sqrt{2\pi} : x \le 0; \sigma > 0 \end{aligned} \quad (2)$$

Standard deviation σ provides a means to tune the distribution skewness depending on the desired *extracts* and respective source.

3.5.2. Validation

Validation is in two stages. The preliminary one is during template matching (**Figure 3**) and the other stage is during validation and output generation. Sequences are validated using a series of validation operators, the first of which is statistical, based on *sf-ipf*. A sequence is considered valid and assigned a label l_t after its *sf-ipf* has passed a threshold frequency. Other case-specific operators are then enforced and labels l_o and l_r are assigned and then the various validation results are aggregated using a simple algorithm (Algorithm 1).

3.5.3. Role of Sequence Validation

In this approach, sequence validation plays two major roles. First is to refine the just extracted sequences of sub-entities by filtering out those which do not comply with the template through a matching process. And second is to validate template sequences using statistical *sf-ipf* and other user operators (ontological and direct expert input).

3.6. Entity Transformation

To guide transformation of an entity, we first characterize its field and then use support of these characteristics to produce a transformation model. We statistically characterize entities using their observed structures (length and count) and types (b-type and u-type). The used structural properties and types are shown in **Table 3**.

Table 3. Entity characteristics.

Category	Characteristics
Structural	Length, count
Basic types (b-types)	Text, number, alphanumeric
User-defined (u-types)	Regular expression

During extraction and parsing, characterized features are formed for each entity field, out of its occurrences. These are in triplet form $\langle \text{name,value,support} \rangle$. Occurrence of "-L 127.0.0.1" in INPUT field of the *ssh* command could produce characteristics such as $\langle$ "count", 2, $S'(c_i)\rangle$, $\langle$ "btype", "alphanumeric", $S'(c_i)\rangle$, $\langle$ "utype", "ip", $S'(c_i)\rangle$ etc. Support $S'(c_i)$ of a characteristic c_i is a fraction of characteristic's count $\text{count}(c_i)$ to all occurrences $\sum \text{count}(c_j)$ (Equation (3)).

$$S'(c_i) = \text{count}(c_i) \sum \text{count}(c_j) \qquad (3)$$

Using support values of the four categories of characteristics in **Table 3** (length, count, b-type, and u-type), a set of fuzzy rules are produced to guide entity transformation. Transformation states and fuzzy conditions are shown in **Figure 5** and **Table 4** respectively. Conditions for the transitions in **Figure 5** which are indicated by numbers are shown in **Table 4**.

4. Case Study

In this section we demonstrate the proposed approach and discuss its performance. We use linux commands (*ssh*, *find* and *shutdown*) as desired *extracts* and demonstrate extraction and subsequent use in solving a command validation problem. In our case study, we treat command validation problem as a classification problem. We use our approach to extract commands from the Web, parse them, validate them and ultimately label them to produce training data for use in building a classification model. In discussing and presenting this case study, we focus on one command only (*ssh*) and present the results for others in discussion (Section 4.5).

4.1. Implementation

We implemented the approach under Java environment using tools which are shown in **Table 5**. **Figure 6** shows the design schematic of the implemented system. The details of the implementation are as follows.

1) Template is implemented using RDF container sequences as shown in **Figure 7**. In a template, field types (e.g. option and destination) also act as a pointers to files which define sub-entities (e.g. ip, email address etc.). A template contained 12 *ssh* command sequences (shown in **Table 7**) which were auto generated using specifications shown in **Figure 4(b)**. The shown bias weights and *sf-ipf*

Table 4. Entity transformation: Conditions.

Transition	Condition(s)
1	S(b-type) > THR_b
2	S(count) > THR_c or S(length) > THR_l S(u-type) > THR_u
3	Savg(length,count,type) > THR_a S(b-type) > THR_b

Table 5. Used tools and environments.

Tool	Usage
Gate's ANNIE [23]	Extraction of text fragments which constitute commands
RDF	Template implementation.
Weka [10]	Entity transformation model, supervised learning.
ManPages	Command reference, Benchmark list.
The web	Data source

Table 6. Sub-entity types of *ssh* command.

Type	Regular expression
ip	([01]?\d\d?|2[0-4]\d|25[0 · · ·
host	([a-z0-9\-\.]+\.(com|org| · · ·
hostport	(([01]?\d\d?|2[0-4]\d|25[0 · · ·
porthostport	([1-9]\d{0,3})\:(([01]?\d · · ·
hostporthostport	(([01]?\d\d?|2[0-4]\d · · ·
user	[\w\-](([\.\w])+[\w]+@(([0 · · ·
time	([1-9]|(1[0-9])|(2[0-4])) · · ·
directory	((\)?[^\s]+\([^\s]+\)?([^ · · ·
file	(((\)?[^\s]+\([^\s]+\)?([^ · · ·
tun:tun	([1-9]\d{0,2})\:([1-9]\d{ · · ·

Table 7. Comma separated *ssh* template sequences.

no.	Sequence	w_b	*sf-ipf*
1	cmdid,null,null,null,null,null	0.68	0.002
2	cmdid,null,null,argument,null,null	0.5	0.002
3	cmdid,null,input-input,null,null,null	0.36	0.004
4	cmdid,null,input,null,null,null	0.36	0.004
5	cmdid,null,input-input,argument,null,null	0.25	0.146
6	cmdid,null,input,argument,null,null	0.25	0.058
7	cmdid,option,null,null,null,null	0.18	0.005
8	cmdid,option,null,argument,null,null	0.13	0.010
9	cmdid,option,input-input,null,null,null	0.09	0.291
10	cmdid,option,input,null,null,null	0.09	0.024
11	cmdid,option,input-input,argument,null,null	0.07	0.291
12	cmdid,option,input,argument,null,null	0.07	0.058

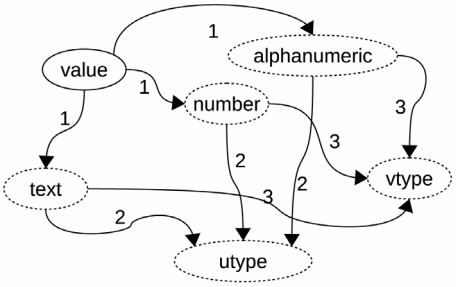


Figure 5. Entity transformation: States.

scores are after extraction and validation and are therefore described in Section 4.3.

2) Entity Type Definitions are shown in **Table 6**. We defined 10 utypes used in *ssh* commands.

3) Generic Rules, although seem complicated as shown in **Figure 8**, are easy to write because the details of sub-entity relationships are ignored at this stage. The rules do not cater for COMMAND field of *ssh* command.

4) Lookup Lists provide references to various NHCE keywords like cmdid (e.g. *ssh*), options (e.g. *-l*) etc.

5) Characterized Features give a statistical snapshot of observed sub-entities in the fields. **Table 8** shows characteristic triplets in tabular form. The values in the table are discussed in Section 4.3.

6) Transformation Model is implemented using C 4.5

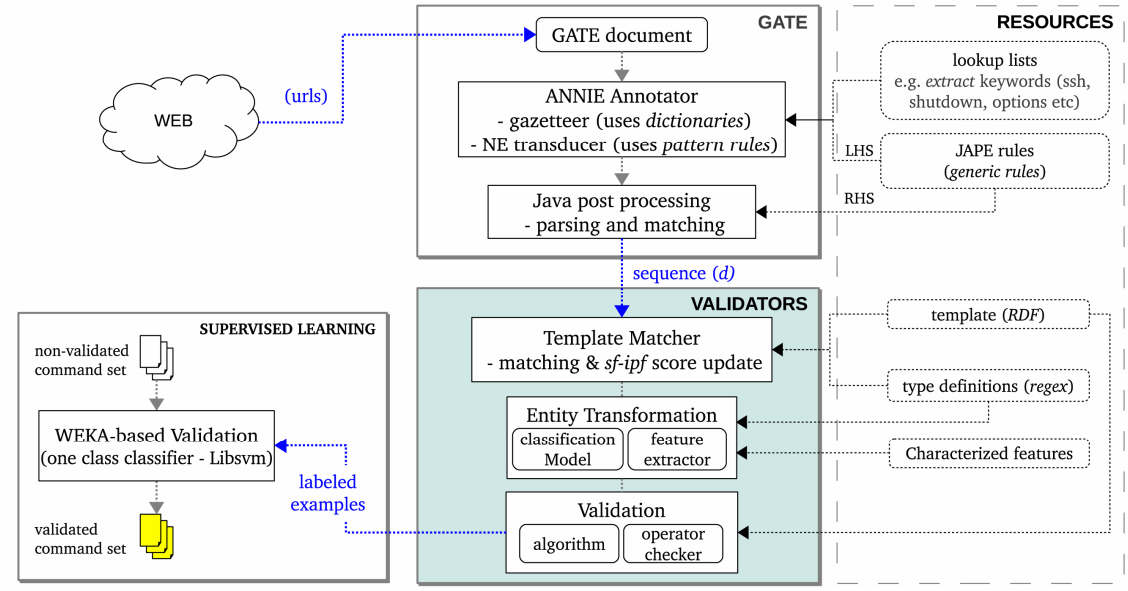


Figure 6. Schematic of the implemented design.

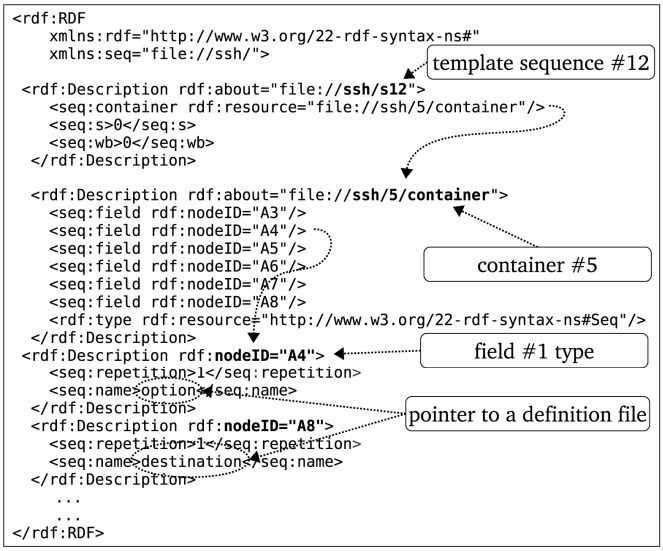


Figure 7. *Ssh* template.

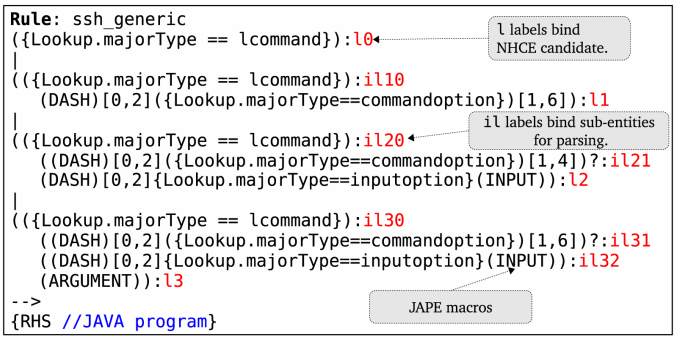


Figure 8. JAPE pattern rule for *ssh* commands.

Table 8. Field characteristic triplets in tabular form.

Characteristic	Value	Field support			
		S1	S2	S3	S4
count	·	·	·	·	·
count	2	0.00	0	0.1	0.00
count	1	1	1	0.9	0.98
length	·	·	·	·	·
length	2	0.00	0.91	0.33	0.27
length	1	0.00	0.09	0.03	0.00
b-type	alphanumeric	0.00	0	0.09	0.05
b-type	text	1	0.85	0.81	0.95
b-type	number	0.00	0.15	0.1	0.00
u-type	host	0.00	0	0.02	0.09
u-type	port	0.00	0.12	0.09	0.00
u-type	port-host-port	0.00	0	0.06	0.00
u-type	user	0.00	0	0.00	0.55
u-type	directory	0.02	0.00	0.07	0.00

algorithm which is built out of manually crafted fuzzy rules based on transitions shown in **Figure 5**.

7) Only template operator is used because we feel that the rationale behind any ontology and direct expert input is ambiguous for evaluational purposes.

4.2. Experimentation Data

We retrieved from the Web 120 html pages which contain information (usage and example) about *ssh* command, 50 html pages for *shutdown* command and 70 for *find* command. We used queries like "how to use...", "linux command reference", "how to connect using...", "how to shutdown...", "how to locate...", "how to find..." etc. against search engines and retrieve relevant pages.

4.3. Extraction, Validation and Results

This section and Section 4.4 are discussed using *ssh* command. Documents were annotated using GATE's ANNIE annotator in which we used our generic rules. Annotation results depicted that the documents contain about 959 *ssh* command candidates (*extracts*) with which we experimented. Our objectives were to investigate:

- Effectiveness of our post-rule processing in extracting and parsing NHCE.
- Performance of *sf-ipf* based validation method.
- Effects of bias weights.
- How rich the current Web content is in terms of diversity and utility of its *extracts*.

To realize these objectives, we manually prepared a benchmark list of 1250 valid *ssh* commands. We do not claim that this is a comprehensive list of *ssh* commands. The list only covers ipv4, does not include neither COMMAND field nor redirections and pipelines (for simplicity and easiness in experimentation). Threshold values were set proportionally, *i.e.* by assuming even distribution across characteristic values with the exception to structural characteristics (count and length) which were set to match btype characteristics. For instance, since there are 3 b-types, then $THR_b = 30\%$. Other thresholds are as follows, $THR_c = THR_l = 30\%$, $THR_u = 10\%$ and $THR_a = 60\%$. Threshold THR_{sf-ipf} was set to the template average value of *sf-ipf*. The values of bias weights used were changing depending on the observed most common sequence. In the end, the bias weights were as shown in **Table 7** with sequence No. 2 as the most common sequence, and therefore used a mean in the distribution Equation (2).

4.3.1. Post-Rule Processing Results

Post-rule processing serves as both a parser and initial validator. Out of the 959 candidate *extracts*, 275 were validated at this stage as to contain *ssh* commands. These were contained in 56 of the 120 used documents, meaning that the positive predictive value (PPV) of Web retrieval was 47%. The distribution of *extracts* among documents is such that a document contains 4.9 *extracts* on average, document with the highest number of *extracts* has 35 *extracts* and the majority of documents contain only 1 *extract*. The validated candidates were

also parsed into NHCE sub-entities for each of which its occurrences were used in characterizing that field. **Table 8** summarizes characterized feature triplets and their observed support values.

4.3.2. Validation Results

In the template, 9 of the 12 template sequences were validated using *sf-ipf*. These resulted into 171 *extract* of which 70 successfully transformed. For example, *ssh-R* 5555: localhost: 3389 to *ssh-R porthostport* is a successful transformation while *ssh-R* 5555: localhost: 3389 to *ssh-R alphanumeric* is a failed transformation. A field transformation into a non-existing type (neither utype nor vtype) is considered a failure and renders the whole *extract* sequence transformation unsuccessful. Like indicated in **Figure 5**, b-type is therefore not a final transformation state. **Table 9** summarized the results of transformation for each field.

When checked against the benchmark list, the precision of the successfully validated and transformed *extracts* was 83% while recall was 27.46%.

4.3.3. Effects of Bias Weights

Bias weights are meant to enable user to reduce the effect of Web-inherent statistical advantage. By changing the value of σ (Equation (2)), the significance of a template sequence can be tuned. Without bias weights, 9 out of 12 template sequences were validated as pointed out earlier. With $\sigma = 0.5$ bias however, sequence No. 6 is invalidated (**Figure 9**).

Table 9. Entity transformation summary.

Field	f_1	f_2	f_3	f_4
Occurrences	2	18	42	55
u-type	1	0	10	46
v-type	1	14	3	7
Rate (%)	100.0	77.80	31.0	96.4
Precision (%)	100.0	100.0	76.9	86.8
Recall (%)	100.0	58.3	45.5	100

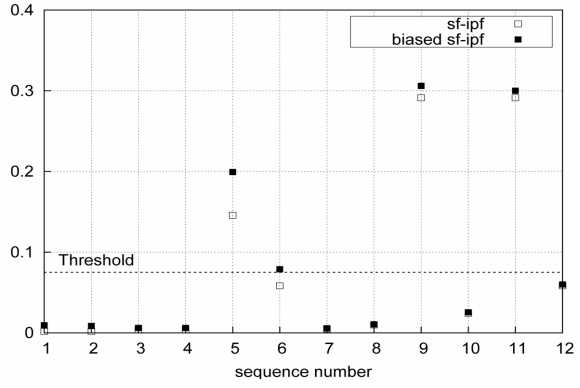


Figure 9. Biasing *sf-ipf*.

4.4. Supervised Learning

The final validated and transformed *extracts* comprise of sequences of vtypes and utypes (e.g. *ssh-v-l user host*). These can be used for further validation of *ssh* commands through either direct use as templates or as labeled examples in supervised learning. We experimented the later by using v-types and utypes as attributes in training datasets and build one class svm classifiers for validation of *ssh* commands. By using a one class classifier, invalid commands are considered outliers. 10-fold cross validation results, using the benchmark list as testing dataset, depicted a classifier with F-Measure 0.336 and accuracy of 39.33%.

4.5. Discussion

The presented case study demonstrates our proposed approach in terms of how NHCE candidates are annotated from free text, parsed, validated, sub-entity transformed and finally used in validating similar NHCEs in a supervised learning setup. **Table 10** summarizes results for all three linux commands which were experimented.

Our rule-based annotation approach is rather of a brutal force nature because relationships between sub-entities were not taken into account when writting the general rules. The initial *extracts* were therefore too noisy. For instance a text fragment *ssh -X -L* 172.0.0.*1 hostmachine* led to two *extract* candidates with both *-L* 172.0.0.1 and *hostmachine* considered as f_3 occurrences in one and *-L* 172.0.0.1 as f_3 occurrence while *hostmachine* as f_4 occurrence in another. Nonetheless the approach is preferred for a number of reasons in addition to existing facts like rule-based methods being methods which offer sound performances as pointed out in Section 2. First reason is that generic rules can be easily written and interpreted. Second reason is that, by considering all ambiguous text fragments in contention, fields in a template sequence are able to see more occurrences and so improve the richness of their entity

Table 10. Results summary.

Item	Metric	*ssh*	Shutdown	Find
Annotation	No. of extracts	959	77	276
Post processing	No. of extracts	275	59	215
Doc. retrieval	PPV	47.0	75.3	31.2
Transformation	Success rate	40.9	88.2	46.2
Template operator	No. of extracts	171	59	186
	Precision	83.0	93.2	63.2
	Recall	27.5	34.0	9.0
Supervised learning	No. examples	70	52	86
	Accuracy	39.3	45.9	17.4
	F-measure	0.3	0.4	0.1

characteristics. Improved entity characteristics lead to better entity transformation. In **Table 9**, for example, although only 10 different vtypes appear in the final validated *extracts*, 14 were transformed.

The use of hand-coded binding labels for parsing falls in the same line of reasoning as generic rules. These are equally important in order to maximize field observations in a top-down extraction approach.

The initial validation, during template matching, removes noisy *extracts*, a consequence of using generic rules. As presented, initial validation saw candidate *extracts* reduced to 275 from 959 for *ssh* command. This is a 71% noise. *shutdown* and *find* do not appear too noisy as shown in **Table 10** (23.4% for *shutdown* and 22.1% for *find*) because of the relatively few documents used and lesser popularity within the Web content.

sf-ipf based validation approach have shown both strength in filtering valid transformable *extracts*. This is supported by good precision values for all three commands (*ssh*-83.0%, *shutdown*-93.2% and *find*-63.2%). These results might demonstrate neither richness of Web content nor practical utility of the *extracts* because of very low recall values but demonstrate the effectiveness of *sf-ipf* in filtering out best NHCE candidates from the original *extracts*. From our experimentations, it is evident that *sf-ipf* alone does not produce comprehensive enough results for practical use. This is depicted by low accuracy values when the *extracts* were used in supervised learning. It is a result of being only statistical and the use of generic rules. For instance, since an occurrence *shutdown -c* is often then an *extract shutdown-c now* is likely to be validated even though it is not valid. As pointed earlier, it is for this reason that our approach incorporates expert input in either ontological operators or direct input operators forms. These operators are provided in terms of relationships between sub-entities. For example, an ontology or list which describes all valid inputs for a particular input flag in *ssh* commands. Surely, when used, these improve performance of *extracts*. When we experimented using direct input by specifying all valid inputs for the most observed input flag (*-L*) in *ssh* command, we noticed a 10% increase in supervised learning accuracy.

Another reason for low accuracy values of the end classification models is the small number of used documents relative to what is avaliable in the Web. Preparation of experimentation documents was done manually, therefore it was quite an expensive process.

5. Related Works

Although Bibpro [1] is a citation parser, it resembles our work in the use of sequence templates. In Bibpro, structural properties and local properties of citation fields are used to create template sequences. This means that a significant prior knowledge about these properties is needed in advance. In our approach, template sequences are created from user's specifications about sub-entity types which occupy NHCE field. Also Bibpro's use of canonicalization makes it highly reliant to NLP. In terms of cardinality of sub-entities, "Booktitle" and "Month" are the fields of highest cardinality (12) which are nevertheless made up of only NPL keywords. We could not compare our approach against Bibpro because of differences in *extract* types and difficulty in implementation of Bibpro as its not open source.

The idea of using relational databases [5] can also be applied in recognition and classification of NHCEs. According to this idea, *extracts* are considered as database queries which have well understood formats and relationships between sub-entities (columns) are not explicit. Although in [5], this idea is presented in NPL context, it provides food for throught in dealing with NHCE.

Transformation of sub-entities is closely related to the discovery of new attributes. Wong *et al*. [2] employ semantic analysis on text fragments and use Bayesian Learning to infer them. The approach enables discovery of such attributes like "online price", "publishing year" etc. using EM algorithm model.

Statistical methods using state-of-the art HMM [13] and CRF [16] are promising ones in dealing with NHCEs. The challenge, however, lies in making them easy to deploy and understand, and integrate with expert input.

To the best of our knowledge, there is no published IE work for the extraction and classification of NHCEs. Most studied *extracts* are made up of keyword sub-entities for which dictionaries [8] and lists are preferred.

6. Conclusions

This paper presented an approach to recognize and classify Named High cardinality Entities. The approach treats sub-entities of a NHCE as a *extract* sequence and uses a series of three categories of operators to validate these *extract* sequences. The foundational operator is a template based operator which scores template sequences with a statistical score called sequence frequency-inverse page frequency and use these sequences to validate *extract* sequences. The approach also enables sub-entities to transform into either a value type (v-type) or a user-defined type (u-type). This transformation brings flexibility on how extraction knowledge is shared, how *extract* features are captured and also how *extracts* are used.

We have demonstrated this approach using linux commands as desired NHCEs. Experimentation results reveal the effectiveness of sequence validation based extraction in capturing both structural and user-defined features and types, characterize them and subsequently use the characterized features to validate *extract* sequences. We have

also shown how an approximated distribution can be used to tune significances of template sequences as a way to regulate statistical acceptability of *extract* sequences.

There are some areas where improvements are still needed in order to make this approach easier to use and deploy. These includes automating the process of building a transformation model, improving the generic rules approach so that *extracts* are less noisy but in sufficient numbers and developing a framework by which ontological operators can be smoothly integrated. Other issues, which are to be further researched on, concern alternatives to generic rules approach in annotating and parsing NHCE *extract*, e.g. statistical CRF-based.

7. Acknowledgements

A part of this work is supported by third supplementary budget for 2011 of Ministry of Internal Affairs and Communications of Japan, Research and Development of Applicable Resource Unit Construction and Reconstitution Technology for Communication Network on Large-Scale Disasters.

REFERENCES

[1] C. C. Chen, K. H. Yang, C. L. Chen and J. M. Ho, "Bibpro: A Citation Parser Based on Sequence Alignment," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 2, 2012, pp. 236-250. doi:10.1109/TKDE.2010.231

[2] T. L. Wong and W. Lam, "Learning to Adapt Web Information Extraction Knowledge and Discovering New Attributes via a Bayesian Approach," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 4, 2010, pp. 523-536. doi:10.1109/TKDE.2009.111

[3] F. Ashraf, T. Ozyer and R. Alhajj, "Employing Clustering Techniques for Automatic Information Extraction from Html Documents," *IEEE Transactions on Systems, Man, and Cybernetics*, *Part C*: *Applications and Reviews*, Vol. 38, No. 5, 2008, pp. 660-673. doi:10.1109/TSMCC.2008.923882

[4] D. C. Wimalasuriya and D. Dou, "Components for Information Extraction: Ontology-Based Information Extractors and Generic Platforms," *Proceedings of the* 19*th ACM International Conference on Information and Knowledge Management*, New York, 2010, pp. 9-18. http://doi.acm.org/10.1145/1871437.1871444

[5] L. Tari, P. H. Tu, J. Hakenberg, Y. Chen, T. C. Son, G. Gonzalez and C. Baral, "Incremental Information Extraction Using Relational Databases," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 1, 2012, pp. 86-99. doi:10.1109/TKDE.2010.214

[6] S. A. Kripke, "Naming and Necessity," Harvard University Press, Cambridge, 1980.

[7] D. Nadeau and S. Sekine, "A Survey of Named Entity Recognition and Classification," *Linguisticae Investigationes*, Vol. 30, No. 1, 2007, pp. 3-26. doi:10.1075/li.30.1.03nad

[8] J. L. Hong, "Data Extraction for Deep Web Using Wordnet," *IEEE Transactions on Systems, Man, and Cybernetics*, *Part C*: *Applications and Reviews*, Vol. 41, No. 6, 2011, pp. 854-868. doi:10.1109/TSMCC.2010.2089678

[9] P. McFedries, "The Coming Data Deluge [Technically Speaking]," *IEEE Spectrum*, Vol. 48, No. 2, 2011, pp. 19. doi:10.1109/MSPEC.2011.5693066

[10] I. H. Witten, E. Frank and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques," 3rd Edition, Morgan Kaufmann, Burlington, 2011.

[11] S. Lawrence, C. L. Giles and K. Bollacker, "Digital Libraries and Autonomous Citation Indexing," *IEEE Computer*, Vol. 32, No. 6, 1999, pp. 67-71. doi:10.1109/2.769447

[12] J. Han, M. Kamber and J. Pei, "Data Mining: Concepts and Techniques," 2nd Edition, Morgan Kaufmann, Burlington, 2006. http://www.amazon.com/Data-Mining-Concepts-Techniques-Management/dp/1558609016

[13] E. Agichtein and V. Ganti, "Mining Reference Tables for Automatic Text Segmentation," *Proceedings of the* 10*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 20-29.

[14] V. Borkar, K. Deshmukh and S. Sarawagi, "Automatic Segmentation of Text into Structured Records," 2001.

[15] R. Malouf, "Markov Models for Language-Independent Named Entity Recognition," *Proceedings of the* 6*th Conference on Natural Language Learning*, Stroudsburg, 2002, pp. 1-4. doi:10.3115/1118853.1118872

[16] C. Sutton and A. McCallum, "An Introduction to Conditional Random Fields for Relational Learning," 2006.

[17] M. Asahara and Y. Matsumoto, "Japanese Named Entity Extraction with Redundant Morphological Analysis," *Proceedings of the* 2003 *Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, *NAACL*, Morristown, 2003, pp. 8-15. doi:10.3115/1073445.1073447

[18] O. Etzioni, M. Cafarella, D. Downey, A. M. Popescu, T. Shaked, S. Soderland, D. S. Weld and A. Yates, "Unsupervised Named-Entity Extraction from the Web: An Experimental Study," *Artificial Intelligence*, Vol. 165, 2005, pp. 91-134. doi:10.1016/j.artint.2005.03.001

[19] A. Yates, "Extracting World Knowledge from the Web," *IEEE Computer*, Vol. 42, No. 6, 2009, pp. 94-97. doi:10.1109/MC.2009.188

[20] H. Cunningham, D. Maynard and V. Tablan, "JAPE: A Java Annotation Patterns Engine (Second Edition)," Technical Report, University of Sheffield, Sheffield, 2000.

[21] S. Sarawagi, "Information extraction," *Found Trends Databases*, Vol. 1, No. 3, 2008, pp. 261-377. doi:10.1561/1900000003

[22] H. C. Wu, R. W. P. Luk, K. F. Wong and K. L. Kwok, "Interpreting tf-idf Term Weights as Making Relevance Decisions," *ACM Transactions on Information Systems*,

Vol. 26, No. 3, 2008, pp. 1-37.
doi:10.1145/1361684.1361686

[23] H. Cunningham, D. Maynard, K. Bontcheva and V. Tablan, "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications," *Proceedings of the* 40*th Anniversary Meeting of the Association for Computational Linguistics* (*ACL*), Philadelphia, 7 February 2003.

