Terminology Extraction from Log Files

Hassan Saneifar, Stéphane Bonniol, Anne Laurent, Pascal Poncelet, Mathieu Roche

Terminology Extraction from Log Files

Hassan Saneifar[1,2], Stéphane Bonniol[2], Anne Laurent[1], Pascal Poncelet[1], and Mathieu Roche[1]

[1] LIRMM - Université Montpellier 2 - CNRS
161 rue Ada, 34392 Montpellier Cedex 5, France
{saneifar,laurent,poncelet,mroche}@lirmm.fr
http://www.lirmm.fr/∼{saneifar,laurent,poncelet,mroche}

[2] Satin IP Technologies
Cap Omega, RP Benjamin Franklin, 34960 Montpellier Cedex 2, France
stephane.bonniol@satin-ip.com
http://www.satin-ip.com/

Abstract. The log files generated by digital systems can be used in management information systems as the source of important information on the condition of systems. However, log files are not exhaustively exploited in order to extract information. The classical methods of information extraction such as terminology extraction methods are irrelevant to this context because of the specific characteristics of log files like their heterogeneous structure, the special vocabulary and the fact that they do not respect a natural language grammar. In this paper, we introduce our approach Exterlog to extract the terminology from log files. We detail how it deals with the particularity of such textual data.

1 Introduction

In many applications, automatic generated reports, known as system logs, represent the major source of information on the status of systems, products, or even causes of problems that can occur. In some areas, such as Integrated Circuit (IC) design systems, the log files generated by IC design tools, contain essential information on the conditions of production and the final products. In order to extract information from textual data, there exists the classic method of Natural Language Processing (NLP) and Information Extraction (IE) techniques. But the particularity of such textual data (*i.e.* log files) raise the new challenges. In this paper, we aim particularly at exploring the lexical structure of these log files in order to extract the terms of domain which will be used in creation of domain ontology. We thus study the relevance of two main methods of terminology extraction within our approach Exterlog (EXtraction of TERminology from LOGs), both of which extract co-occurrences with and without the use of syntactic patterns.

In Sect. 2 we present the characteristics and difficulties of this context. Our approach Exterlog is developed in Sect. 3. Section 4 describes and compares the various experiments that we performed to extract terms from the logs. Finally, we propose a comparison of Exterlog and TermExtractor system.

2 Context

In the domain of log file analysis, some logs like network monitoring logs or web usage logs are widely exploited [1][2]. These kinds of logs are based on the management of events. That is, the computing systems record the system events based on their occurring times. The contents of these logs comply with norms according to the nature of events and their global usage (*e.g.,* web usage area). However, in some areas such as IC design systems, rather than being some recorded events, the generated log files are digital reports on configuration, conditions and states of systems. The aim of the exploitation of these log files is not to analyze the events but to extract information about system configuration and especially about the final product's conditions. Hence, information extraction in log files generated by IC design tools has an attractive interest for automatic management and monitoring of production line. However, several aspects of these log files have been less emphasized in existing methods of information extraction and NLP. These specific characteristics pose several challenges that require more research.

The design of IC consists of several levels each corresponds to some design rules. At every level, several tools can be used. Despite the fact that the logs of the same design level report the same information, their structures can significantly differ depending on the design tool used. Specifically, each design tool often uses its own vocabulary to report the same information. For example, at the so-called verification level, two log files (*e.g.,* log "A" and log "B") are produced by two different tools. The information about, for example, the "`Statement coverage`" will be expressed as follows in the log "A":

```
      TOTAL    COVERED PERCENT
   statements    20       21    22
```

But the same information in the log "B", will be disclosed from this single line:

```
   EC: 2.1%
```

As shown above, the same information in two log files produced by two different tools is represented by different structures and vocabulary. Moreover, the evolution of design tools changes the format of data in logs. The heterogeneity of data exists not only between the log files produced by different tools, but also within a given log file. For example, the symbols used to present an object, such as the header for tables, change in a given log. Similarly, there are several formats for punctuation, the separation lines and representation of missing data. To best generalize the extraction methods, we thus need to identify the terms used by each tool in order to create the domain ontology. This ontology allows us to better identify equivalent terms in the logs generated by different tools. The domain ontology can help to reduce the heterogeneity of terms existing in logs produced by different design tools. For instance, to check "`Absence of Attributes`" as a query on the logs, one must search for the following different sentences in the logs, depending on the version and type of design tool used:

```
"Do not use map_to_module attribute",
"Do not use one_cold or one_hot attributes",
"Do not use enum_encoding attribute",
```

Instead of using several patterns, each one adapted to a specific sentence, by associating the words "`map_to_module attribute`", "`one_hot attributes`" and "`enum_ encoding attribute`" to the concept "`Absence of Attributes`", we use a general pattern that expands automatically depending on the type of log. The ontology-driven expansion of query is studied in many work [3].
Moreover, the language used in these logs is a difficulty that affects the methods of information extraction. Although the language of log files are similar to English, the contents of these logs do not usually comply with "*classic*" grammar. Moreover, there exist words that are often constituted from alphanumeric and special characters.

Since the concepts used in domain ontology are the terms of log files, we aim at extracting the terminology of the log files. However, due to the particularity of log files described above, the methods of NLP, including the terminology extraction, developed for texts written in natural language, are not necessarily well suited to the log files. In this paper, we thus study these methods and their relevance in this specific context. Finally, we propose our approach ExterLog for extracting terminology from these log files.

3 Terminology Extraction from Log Files

The extraction of co-occurring words is an important step in identifying the terms. We explain at first some of approaches used to identify the co-occurrences and to extract the terminology of a corpus. Then, we introduce our approach of terminology extraction adapted to log files.

3.1 Related Work

Some approaches are based on syntactic techniques which rely initially on the grammatical tagging of words. The terminological candidates are then extracted using syntactic patterns (*e.g.,* adjective-noun, noun-noun). We develop the grammatical tagging of log files using our approach ExterLog in Sect. 3.2. Bigrams[1] are used in [4] as features to improve the performance of the text classification. Though, the series of three words (*i.e.* trigrams) or more is not always essential [5]. Exit, introduced by [6] is an iterative approach that finds the terms in an incremental way. Xtract is a terminology extraction system, which identifies lexical relations in the large corpus of English texts [7]. TermExtractor, submitted by [8], extracts terminology consensually referred in a specific application domain. To select the relevant terms of domain, some measures based on entropy are used in TermExtractor. The statistical methods are generally used for evaluating the adequacy of extracted terms [9]. In these methods, the occurrence frequency of candidates is a basic element. However, since the repetition of words is rare in log files, these statistical methods are not well suited. Indeed, statistical approaches can cope with high frequency terms but tend to miss low frequency ones [10].

[1] N-grams are defined as the series of any "n" words.

Most of these studies are experimented on textual data which are classical texts written in natural language. Most of the experimented corpus are structured in a consistent way. In particular, they comply with the grammar of NL. However, the characteristics of logs such as their non compliance with NL grammar, their heterogeneous, and evolving structures (cf. Sect. 2) impose an adaptation of these methods to log files.

3.2 EXTERLOG

Our approach, EXTERLOG, is developed to extract the terminology in the log files. This process involves normalisation of log files, grammatical tagging of words and co-occurrences extraction.

Normalization. Given the specificity of our data, the normalization method, adapted to the logs, makes the vocabulary and structure of logs more consistent. We replace the punctuations, separation lines and the headers of the tables by special characters to limit ambiguity. Then, we tokenize the texts of logs, considering that certain words or structures do not have to be tokenized. For example, the technical word "`Circuit4-LED3`" is a single word which should not be tokenized into two words "Circuit4" and "LED3". Besides, we make the normalization method to distinguish the lines representing the header of tables from the lines which separate the parts. This normalization makes the structure of logs produced by different tools more homogeneous.

Grammatical Tagging. Grammatical tagging (also called *part-of-speech tagging*) is a method of NLP used to annotate words based on their grammatical roles. In our context, due to the particularity of log files described in Sect. 2, there are some difficulties and limitations for applying a grammatical tagging. Indeed, the classic techniques of POS tagging are developed and trained according to the standard grammar of a natural language. To identify the role of words in the log files, we use BRILL rule-based part-of-speech tagging method [11]. As existing taggers like BRILL are trained on general language corpora, they give inconsistent results on the specialized texts. [12] propose a semi-automatic approach for tagging corpora of specialty. They build a new tagger which corrects the base of rules obtained by BRILL tagger and adapt it to a corpus of specialty. In the context of log files, we also adapted BRILL tagger to our context by introducing the new *contextual* and *lexical* rules. Indeed, the classic rules of BRILL, which are defined according to the NL grammar, are not relevant to log files. For example, a word beginning with a number is considered a "*cardinal*" by BRILL. However, in the log files, there are many words like `12.1vSo10` that must not be labeled as "*cardinal*". Therefore, we defined the special *lexical* and *contextual* rules in BRILL. Since the structures of log files can contribute important information for extracting the relevant patterns in future work, we preserve the structure of files during grammatical tagging. We introduce the new tags, called "*Document Structure Tags*", which present the different structures in log

files. For example, the tag "\TH" represents the header of tables or "\SPL" represents the lines separating the log parts. The special structures in log files are identified and normalized during preprocessing. Then, they are annotated during tagging according to the new specific contextual rules defined in Brill. We use these tagged logs in next level to extract the co-occurrences.

Extraction of Co-occurrences. We are looking for co-occurrences in the log files with two different approaches: (1) using defined *part-of-speech* syntactic patterns, (2) without using the syntactic patterns.

We call the co-occurrences extracted by the first solution "POS-candidates"[2]. This approach consists of filtering words by the syntactic patterns. The syntactic patterns determine the adjacent words with the defined grammatical roles. The syntactic patterns are used in [9] to extract terminology. For complex terms identification, [9] defines syntactic structures which are potentially lexicalisable. As argued in [9], the base structures of syntactic patterns are not frozen structures and accept variations. According to the terms found in our context, the syntactic patterns "\JJ - \NN" (Adjective-Noun) and "\NN - \NN" (Noun-Noun) are used to extract the "POS-candidates" from log files.

The co-occurrences extracted by the second approach are called "bigrams". A bigram is extracted as a series of any two adjacent relevant words[3]. Bigrams are used in NLP approaches as representative features of a text [4]. However, the extraction of bigrams does not depend on the grammatical role of words. To extract significant bigrams, we normalize and tokenize the logs to reduce the rate of noise. We also eliminate the stop words existing in the logs. In this method, we thus do not filter the words according to their grammatical roles.

4 Experiments

We experimented two different approaches for the extraction of terminology from these logs: (1) extraction of *POS-candidates* and (2) extraction of *bigrams*. Here, we analyze the terminological candidates obtained by each one. The log corpus is composed of the logs of all IC design levels and its size is about 950 KB.

4.1 POS-candidates vs. Bigrams

To analyze the performance of the two approaches chosen for the extraction of bigrams, we must evaluate the terms extracted. To automatically evaluate the relevance of the extracted terms, we compare the POS-candidates and bigrams with terms extracted from the reference documents. Indeed, for each level of design of integrated circuits, we use certain documents, which explain the principles and the details of design tools. We use these documents as "reference experts"

[2] POS: Part-Of-Speech

[3] The relevant words, in our context, are all words of the vocabulary of this domain excluding the stop words like "have" or "the".

in the context of an automatic validation. Indeed, if a term extracted from logs is used in the reference documents, it is a valid term of domain. However, there are several terms in the logs especially the technical terms that are not used in the references. Therefore, a validation by an expert, carried out in our future work is needed to complete the automatic validation. We note that, to extract the domain terminology, we have to use log files and not the reference documents because, as described above, there are some terms that do not appear in reference documents according to their nature. Hence, we could use the references as a validation tool but not as the base of domain terminology.

Moreover, in order to select the most relevant and meaningful terms, we filter the extracted terminological candidates based on their frequency of occurrences in the logs. Therefore, we choose terminological candidates having a frequency of at least 2 (*i.e.* pruning task). We calculate the precision and recall of extracted candidates as shown below:

$$\text{Precision} = \frac{|Candidates \cap Terms\ of\ ref|}{|Candidates|} \qquad \text{Recall} = \frac{|Candidates \cap Terms\ of\ ref|}{|Terms\ of\ ref|}$$

Table 1 shows the precision and recall of POS-candidates and bigrams before and after pruning. To evaluate the terms extracted from logs, the precision is

		Level 1		Level 2		Level 3		Level 4		Level 5	
		POS	Bigrams	POS	Bigrams	POS	Bigrams	POS	Bigrams	POS	Bigrams
Before	Precision	67.7	11.3	20.7	6.5	37.8	9.9	40.1	6.5	19.6	5.1
	Recall	0.7	0.4	7.6	7.5	1.3	1.0	9.5	8.8	0.3	0.5
After	Precision	81.1	10.1	18.0	5.0	37.2	5.9	27.3	7.1	37.1	5.5
	Recall	0.1	0.1	3.0	2.0	0.1	0.4	1.6	2.2	0.2	0.1

Table 1. Precision and recall of terminological candidates before and after pruning.

the most adapted measure to our context. Indeed, this measure gives a general tendency of the quality of terms extracted by our system. Note that to calculate a perfectly adapted precision, we should manually evaluate all the terms proposed by Exterlog. However, this task is difficult and costly to implement. The comparison of terminological candidates with the reference terms shows that the terminology extraction based on syntactic patterns is quite relevant to the context of log files. The precision of POS-candidates is indeed higher than the precision of bigrams. Our experiments show that an effort in normalization and tagging tasks is quite useful in order to extract quality terms. We note that the pruning of terms does not significantly improve results. As we have already explained, in our context, terms are not generally repeated in logs. Therefore, a *representative term* does *not* necessarily have a *high frequency*.

The low recall of terminological candidates is due to the large number of reference terms. The reference corpus is about five times larger than the logs corpus. In addition, we found that many extracted terminological candidates that

have not been validated by reference terms are technical words or abbreviations, which are only found in the logs and not in the reference documents of domain. That is why the recall results are not entirely representative for evaluating the quality of Exterlog.

4.2 Validation by Experts

In order to validate the "automatic validation protocol" that we experimented using the reference documents, we asked two domain experts to evaluate the validated terms by our protocol. We calculate the percentage of terms extracted by Exterlog and validated using reference documents which are also annotated as relevant by experts. The results show that 84% to 98.1% of the terms validated by our protocol are really relevant terms according to experts[4].

4.3 Exterlog vs. TermExtractor

Here, we compare the results of our approach Exterlog with those obtained by TermExtractor on the same corpus of logs. We chose TermExtractor because it is well configurable and is evaluated by many users in many domains [8]. To adapt TermExtractor to this context, we configured it according to characteristics of log files and especially the type of terms found in this context. Table 2 shows the results obtained by TermExtractor compared with those obtained by Exterlog (*using syntactic patterns*). By analyzing the terms ex-

	Level 1		Level 2		Level 3		Level 4		Level 5	
	Ext	Ter	Ext	Ter	Ext	Ter	Ext	Ter	Ext	Ter
Precision	67.7	56.1	20.7	14.0	37.8	38.1	40.1	35.2	19.6	26.3
Recall	0.7	0.3	7.6	0.3	1.3	0.4	9.5	2.5	0.3	0.1

Table 2. Precision and recall of terms extracted by Exterlog (Ext) and by TermExtractor (Ter)

tracted by TermExtractor, we find that the structure of logs has influenced the extraction of terms. That is, some terms extracted by TermExtractor must not be considered as a term because of the position of words (*used in the term*) in text of logs. Furthermore, the technical terms of domain, normally constituted of special or alphanumeric characters, like "`ks_comp engine`" or "`rule b9`" are rarely found by TermExtractor. According to Table 2, our approach Exterlog extracts more relevant terms than TermExtractor. That is due

[4] This interval is due to some terms which are annotated as no idea by experts. If we consider the no idea terms as irrelevant, 84% of terms validated by our protocol are really relevant according to experts. If these terms are not taken into account in the calculation, we obtain 98.1% of terms really relevant.

to the special normalisation of logs and particularly due to the special contextual and lexical rules that we have defined using Brill tagger.

5 Conclusion & Future Work

In this paper, we described a particular type of textual data: reporting log files. These textual data do not comply with the grammar of natural language, are highly heterogeneous and have evolving structures. To extract domain terminology from the log files, we extracted the co-occurrences with two different approaches: (1) using the syntactic patterns and (2) without syntactic patterns. The results show that terms obtained using the syntactic patterns are more relevant than those obtained without using syntactic patterns. Our experiments show that our approach extracts more relevant terms than other terminology extraction methods like TermExtractor. Our future work will especially focus on the study of the more advanced protocols of automatic term evaluation.

References

1. Yamanishi, K., Maruyama, Y.: Dynamic syslog mining for network failure monitoring. In: KDD '05, ACM (2005) 499–508
2. Facca, F.M., Lanzi, P.L.: Mining interesting knowledge from weblogs: a survey. Data Knowl. Eng. **53**(3) (2005) 225–241
3. Dey, L., Singh, S., Rai, R., Gupta, S.: Ontology aided query expansion for retrieving relevant texts. In: AWIC. (2005) 126–132
4. Tan, C.M., Wang, Y.F., Lee, C.D.: The use of bigrams to enhance text categorization. Inf. Process. Manage. **38**(4) (2002) 529–546
5. Grobelnik, M.: Word sequences as features in text-learning. In: In Proceedings of the 17th Electrotechnical and Computer Science Conference (ERK98. (1998) 145–148
6. Roche, M., Heitz, T., Matte-Tailliez, O., Kodratoff, Y.: Exit: Un système itératif pour l'extraction de la terminologie du domaine à partir de corpus spécialisés. In: Proceedings of JADT'04. Volume 2. (2004) 946–956
7. Smadja, F.: Retrieving collocations from text: Xtract. Comput. Linguist. **19**(1) (1993) 143–177
8. Sclano, F., Velardi, P.: Termextractor: a web application to learn the shared terminology of emergent web communities. In: I-ESA'07, Funchal, Portugal (2007)
9. Daille, B.: Conceptual structuring through term variations. In: Proceedings of the ACL 2003 workshop on Multiword expressions, Morristown, NJ, USA, Association for Computational Linguistics (2003) 9–16
10. Evans, D.A., Zhai, C.: Noun-phrase analysis in unrestricted text for information retrieval. In: Proceedings of the 34th annual meeting on Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1996) 17–24
11. Brill, E.: A simple rule-based part of speech tagger. In: In Proceedings of the Third Conference on Applied Natural Language Processing. (1992) 152–155
12. Amrani, A., Kodratoff, Y., Matte-Tailliez, O.: A semi-automatic system for tagging specialized corpora. In: PAKDD. (2004) 670–681