Journal of Universal Computer Science, vol. 21, no. 4 (2015), 604-635
submitted: 22/11/12, accepted: 26/3/15, appeared: 1/4/15 © J.UCS

From Terminology Extraction to Terminology Validation: An Approach Adapted to Log Files

Hassan Saneifar
(LIRMM - University Montpellier 2 - CNRS. Montpellier, France
Satin Technologies, Montpellier, France
hassan.saneifar@gmail.com)

Stéphane Bonniol
(Satin Technologies, Montpellier, France
bonniol@satin-tech.com)

Pascal Poncelet
(LIRMM - University Montpellier 2 - CNRS, Montpellier, France
poncelet@lirmm.fr)

Mathieu Roche
(TETIS - Cirad - Irstea - AgroParisTech, Montpellier, France
LIRMM - University Montpellier 2 - CNRS, Montpellier, France
mathieu.roche@cirad.fr)

Abstract: Log files generated by computational systems contain relevant and essential information. In some application areas like the design of integrated circuits, log files generated by design tools contain information which can be used in management information systems to evaluate the final products. However, the complexity of such textual data raises some challenges concerning the extraction of information from log files. Log files are usually multi-source, multi-format, and have a heterogeneous and evolving structure. Moreover, they usually do not respect natural language grammar and structures even though they are written in English. Classical methods of information extraction such as terminology extraction methods are particularly irrelevant to this context. In this paper, we introduce our approach Exterlog to extract terminology from log files. We detail how it deals with the specific features of such textual data. The performance is emphasized by favoring the most relevant terms of the domain based on a scoring function which uses a *Web* and *context* based measure. The experiments show that Exterlog is a well-adapted approach for terminology extraction from log files.

Key Words: Information Extraction; Natural Language Processing; Text Mining; Terminology Extraction; Terminology Ranking; Log Files
Category: I, I.2.7, I.7

1 Introduction

Nowadays, in many application areas, modern computing systems are instrumented to generate huge reports about occurring events in a format of textual data usually called log files. Log files are generated in every computing field to report the status of systems, products, or even causes of problems that can occur. Number of computational systems which output hundreds of log files, documenting what they are doing and how the tasks are performed, is dramatically increasing. In many application areas like as digital design or monitoring systems, it is not unusual that gigabytes of log files to be generated per day. Log files may also include data about critical parameters, sensor outputs, or a combination of those. Such files are also used during various stages of software development, mainly for debugging and profiling purposes. Log files became a standard part of large application and are essential in operating systems, computer networks, and distributed systems [Adedayo and Olivier, 2015].

1.1 Log Files Analysis

Some kinds of log files, called Web server logs, register data regarding user access to Web servers. These log files provide information about users access patterns and are largely exploited in research on Intrusion Detection or Pattern Extraction [Yamanishi and Maruyama, 2005, Facca and Lanzi, 2005]. However, there are many kinds of log files generated in other application domains which are are not systematically exploited in an effective way because of their special characteristics. For example, diagnostic imaging systems may also be configured to generate one or more log files [Thattil, 2008]. The log files may include functions and activities performed by the imaging system, often in a time-associated format. Accordingly, these log files may be used by technicians to facilitate detection of faults associated with the diagnostic imaging system and subsequent diagnosis and/or servicing. The generation of log files by computational applications is also common. These log files, usually called execution log files, report essential information to monitor application activities [Jiang et al., 2008]. Execution logs are widely available and helpful in monitoring, remote issue resolution, and system understanding of complex enterprise applications.

There are many proposals for standardized log formats such as W3C and SNMP formats [Jiang et al., 2008]. To generate Web server log files, according to general use of Web servers, there is a universal format. However, most log files generated in other fields use ad-hoc non-standardized logging formats.

1.2 Log Files and EDA

There are different types of log files based on the application domain. In this paper, we focus on log files generated by Electronic Design Automation (EDA) systems. Electronic design automation is a category of software tools for designing electronic systems such as printed circuit boards and Integrated Circuits (IC).

```
                Leakage    Internal     Net      Switching
 Instance  Cells Power(nW)  Power(nW)  Power(nW)   Power(nW)
-----------------------------------------------------------------------
aes_cipher_top  --   901.234       --      346        --

      Timing
      ------
Warning : Possible timing problems have been detected in this design.

 Slack     Endpoint   Cost Group
-------------------------------------
  +XXps sa22_reg[7]/D default
  -XXps sa22_reg[7]/D default


 Clock Description
 ----------------

 Clock                       Clock   Source    No of
 Name    Period  Rise   Fall  Domain  Pin/Port  Registers
------------------------------------------------------------
 clk     xx      xx     xx   domain_1  clk        xx
```

(a)

```
Operating Conditions: WCCOM   Library: power_sample
Wire Load Model Mode: top

Design         Wire Load Model            Library
------------------------------------------------
RIJNDAEL_TOP_ITER_32   10x10              power_sample


Global Operating Voltage = 4.75
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 0.100000ff
    Time Units = 1ns
    Dynamic Power Units = 100nW
    Leakage Power Units = 1nW


  Cell Internal Power  = 123.456 uW   (20%)
  Net Switching Power  = 789.012 uW   (80%)
                        ---------
Total Dynamic Power    = 345.678 uW  (100%)

Cell Leakage Power     =  901.234 uW
```

(b)

Figure 1: A small segment of two log files generated by two different IC design tools

Since EDA software runs a long time in batch mode, the generated log files by design tools are often the user's sole feedback. Users constantly need to check progress by listing these logs. Analysing and understanding these log files design is a daunting task. Design verification is also the process of going through each stage of a design and ensuring that it will do what the specification requires it to do. Here, users also need to look for information in verification logs to evaluate the produced IC. Design-quality monitoring and reporting has now become a field in itself. It can make the difference between meeting delivery schedules and not meeting them, between one-pass silicon and expensive response, and between meeting a market window and missing it entirely. Thus, an automatic and efficient solution to check the design quality based on the information contained in the log files is an essential requirement.

In this domain, to ensure the design quality, there are some quality check rules which should be verified. These quality check rules are usually formulated in the form of natural language questions (e.g., "Capture the total fixed cell STD" or "Captures the maximum Resistance value"). Verification of these rules is mainly performed by analysing the generated log files. In the case of large designs that the design tools may generate megabytes or gigabytes of log files each day, the problem is to wade through all of this data to locate the critical information we need to check the quality check rules.

These log files typically include a substantial amount of data. Accordingly, manually locating information is a tedious and cumbersome process. A wide array of techniques has been developed to help in the retrieval of relevant information from the log files. Unfortunately, the large amount of log data that must be analysed may overwhelm the presently available techniques, thereby resulting in a time-consuming and often error-prone process. This may be especially problematic in systems that handle high volumes of log data.

Furthermore, the particular characteristics of log files, specially those generated by EDA design tools, rise significant challenges in retrieval of information from the log files. The specific features of log files limit the usefulness of manual analysis techniques and static methods. Automated analysis of such logs is complex due to their heterogeneous and evolving structures and the large non-fixed vocabulary. Since the specificities of the log files is a primordial issue which requires to be developed. In the current systems, Information extraction on log files is typically done by manually-created regular expressions. But it is time-consuming and error-prone. These patterns are not flexible to the structure or vocabulary changes, which is frequently occurs in log files. Changing the design tool or even updating to a new version can results into a considerable change in vocabulary and structure of the corresponding generated log files. Creating the regular expression patterns also needs to locate the seeking information in log files. Beside being time-consuming and error-prone task, it needs a specialized knowledge about the structure and vocabulary of all types of log files.

Although information extraction in log files generated by IC design tools is attractive for automatic design management, monitoring and design quality verification, are not systematically exploited in an efficient way. Automatically locating information in huge log files can significantly help these domain engineers to understand and analysis the data contained in log files. Moreover, by automatically locating a requested information in log files, we do not need any more to build the complex and sophisticated extraction patterns which are used to avoid the extraction of structurally similar information.

In this paper, we describe our approach, named EXTERLOG (EXtraction of TERminology from LOGs), to extract the terminology of log files. We study within our approach the relevance of two main methods of terminology extraction. These methods are based on the extraction of co-occurrences with and without the use of syntactic patterns. Moreover, in order to automatically validate the relevant candidate terms, we present a method to filter the extracted terms based on a ranking function.

This paper is organised as follows. In Section 2, we detail the motivation of terminology extraction in our context. The characteristics and difficulties of this context are presented in Section 3. Our EXTERLOG approach is developed in Section 5 along with our term filtering method. Section 6 describes and compares the various experiments that we performed to extract terms from the log files and to evaluate the performance of EXTERLOG.

2 Motivation

2.1 EDA and Text-mining

Within our previous work to retrieve the relevant information from these log files [Saneifar et al., 2010], we observed that the domain lexical knowledge can improve the performance of information retrieval. Thus, we seek to integrate a kind of lexical knowledge on the log file domain to our Information Extraction approach. In this work, we particularly aim at proposing a relevant approach to explore the lexical structure of log files in order to extract the domain terminology. The domain terms are subsequently used to enrich the log file features. This terminological knowledge enables us to better characterize log files and identify their relevant features.

We note that the obtained terminological knowledge will also serve as a starting point to compiling dictionaries or even to create the EDA domain ontology in our future work. In fact, in order to build such an ontology, we first have to identify the domain terms which will be considered as instances of the ontology.

The large volume of logs and their special features limit the usefulness of manual analysis techniques and static methods. Automated analysis of such logs is complex due to their heterogeneous and evolving structures and the large non-fixed vocabulary. We note that the particularities of log files exist in most types of log files regardless of their application domain.

We consider log files as a kind of "complex textual data", i.e. containing *multi-source*, *heterogeneous*, and *multi-format* data. These particularities, detailed in Sect. 3, raise new challenges which make the classic methods of Information Extraction (IE) and Natural Language Processing (NLP) irrelevant. As an example, Figure 1 shows a small section of two log files generated by two different IC design tools. According to the vocabulary and structure of the two log files, it seems that they are reporting different information. However, the colored lines in the two log files report the same information concerning the "*Dynamic Power*". This is just an example of multi-vocabulary difficulties in processing log files which are due to the fact that log files usually contain multi-source data. Moreover, as we can see in Figure 1, we have to process different kinds of data (e.g., numerical data, textual data, structured data (tables), etc.) in log files. The particularities of log files and difficulties are detailed in Section 3.

In this context, a key challenge is to provide approaches that consider the *multi-source*, *heterogeneous* and *scalable structures* of log files as well as their *special vocabulary*. Furthermore, although the contents of these logs are similar to texts written in Natural Language (NL), they comply neither with the grammar nor with the NL structure. Therefore, in order to extract information from the logs, which is an essential task according to the application of log files, we need to adapt the methods of Natural Language Processing (NLP) and Information

Extraction (IE) to the specific characteristics of such textual data.

2.2 How to Integrate Linguistic Information?

Adding linguistic knowledge to Information Retrieval process can improve the retrieval performance. Methods for integrating linguistic content within information retrieval activities are receiving a growing attention [Moldovan et al., 2003]. In our context, we have observed during experiments a significant improvement in performance of passage retrieval as well as query expansion by using the domain terminological knowledge. Using the EDA domain-specific terms as textual features to characterize documents (e.g., passages, lexical words) provides a more relevant presentation of the documents.

When working on specialized languages and specific domains, terminology plays a crucial role as it aims at describing and organizing the knowledge of the domain through the concepts, and their lexical realizations, that are used [Déjean et al., 2005].

Regarding our work, we obtain the better results while the log file segments are characterized by multi-word terms besides the single words. This issue highly motivates us to construct this domain ontology in order to better determine the relevant multi-word terms. Moreover, using the domain-specific terms as index term in Information Retrieval systems is revealed to improve the retrieval performance. We can use the log file domain-specific terminological knowledge to better determine the features of the log files to be used as index terms.

The next motivation to extract the log file terminology is to use it as the starting point in the creation of the domain ontology. Log files usually contain multi-source data. That is, for example, in the IC design domain (*also some other domains like diagnostic imaging systems*) different tools can be used while each tool generates its own log files. Despite the fact that these logs report the same information, their structures and vocabulary can significantly differ depending on the tool used.

Patterns of Information Extraction can be obtained by using an ontology of the domain adapted to different vocabulary. In fact, several approaches are based on the domain ontology to guide the information extraction process [Even and Enguehard, 2002]. [Silvescu et al., 2001] study ontology-assisted approaches to customizable data integration and Information Extraction from heterogeneous and distributed data sources. SOBA, presented by [Buitelaar et al., 2008], is an ontology-based Information Extraction system. It can be used to query information contained in different sources, including plain text and tables in an integrated and seamless manner.

In our context, the ontology of the domain enhances identification of equivalent terms in logs generated by different tools. For instance, during Information Extraction from log files generated by IC design tools, we can have a query like "`check the absence of attributes`". To extract the information, we have to search for the following different sentences in the log files, 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 terms (i.e. instances) "`map_to_module attribute`", "`one_hot attributes`", and "`enum_ encoding attribute`" with the concept "`Absence of Attributes`", we can use a general pattern. With the help of such semantic associations between the domain terms, the query can be automatically expanded to adapt to different kinds of answer patterns.

Such a process, known as "ontology-driven expansion of query", has been studied in many works (see [Voorhees, 1994] & [Dey et al., 2005]). The domain ontology allows us to categorize terms associated with a concept sought in the logs. However, in order to build such ontology, we have to identify the domain terms which will be considered as instances of the ontology.

Considering all the described benefits of the domain terminology in retrieving and extracting information from the log files, we aim at proposing a terminology extraction appraoch adapted to the specifities of this complex textual data.

3 Features of Log Files

As described, the contents of some log files like network monitoring logs or web usage logs comply with standards according to the nature of the information and its global usage (*e.g.,* web usage area). However, in some areas such as IC design systems, generated log files, which are digital reports on configurations, conditions, and states of systems, have very heterogenous formats. The aim of the exploiting these log files is not to analyze the events, but rather to extract information about the system configuration and especially about the conditions of final products. Hence, information extraction in log files generated by IC design tools is attractive for automatic management and monitoring of production lines. However, several aspects of these log files have been less emphasized in current methods of information extraction and NLP.

As described in Section 2, log files contain multi-source data. The IC design consists of several levels with each one corresponding to some design rules. At every level, several design tools can be used. Although logs of the same design level report the same information, each design tool uses its own vocabulary and textual structure. For instance, at the so-called verification level, we produce both log files (*e.g.,* log "A" and log "B") using two different tools. Information about, for example, the "`Statement coverage`" will be expressed as follows in log "A":

```
TOTAL      COVERED PERCENT
Lines      10        11   12
statements 20        21   22
```

But the same information in log "B" is expressed as follows:

```
EC: 2.1%
```

As shown above, the same information in two log files is represented by dissimilar structures and vocabulary. In addition, design tools change over time, often unexpectedly. Therefore, the *format of the data* in the log files changes, which makes automatic data management difficult. Data *heterogeneity* occurs 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 of tables, change in a given log. Similarly, there are several kinds of punctuation, data formatting and representation of missing data. Therefore, intelligent and generalized methods are required, which can be applied on the different logs that have the heterogeneous vocabulary and structure. These methods must also take the variable vocabulary of these logs into account.

Moreover, the language used in these logs is a difficulty that impacts information extraction methods. Although the language used in these logs is English, their contents do not usually comply with "*classic*" grammar. In the processing of log files, we also deal with multi-format data: textual data, numerical data, alphanumerical, and structured data (e.g., table and data block). There are also many technical words that contain special characters. Due to these specific characteristics of log files, NLP methods, including terminology extraction tasks, developed for texts written in natural language, are not necessarily well adapted to log files.

4 Related Work

In this section, we first present and study the related work in this domain. Then, we will discuss the background methodes.

4.1 Studying the Related Work

The extraction of domain terminology from textual data is an essential task to establish specialized dictionaries of specific domains [Roche et al., 2004]. The extraction of co-occurring words is an important step in identifying terms. To identify co-occurrences, some approaches like [Penas et al., 2001] are based on syntactic techniques which initially rely on part-of-speech tagging. Candidate terms are then extracted using syntactic patterns (e.g. adjective-noun, noun noun). Part-of-speech (POS) tagging (also called grammatical tagging) is a NLP method used to analyse text files and annotate words based on their grammatical roles. In the same category, we have also Syntex, proposed by [Bourigault and Fabre, 2000], which performs syntactic analysis of texts to identify nouns, verbs, adjectives, adverbs, the noun phrases, and verbal phrases. It analyses the text by applying syntactic rules to extract terms. Exit, introduced by [Roche et al., 2004], is an iterative approach that finds nominal and verbal terms in an incremental way. A term found in an iteration is used in the next one to find more complex terms.

In [Déjean et al., 2005], authors present their work on extraction of bilingual lexicon (English and German) from parallel corpora in the medical domain. The extracted lexicons are semi-automatically used to enrich mono- or bilingual thesauri. In [Déjean et al., 2005], the main focus is on the extraction of lexicon from comparable corpora. The authors argue that their approach is relevant to the medical domain as there are bilingual thesauri in this domain. In order to evaluate the extracted lexicons, they manually extracted a reference lexicon comprising 1,800 translation pairs from the studied corpus. About 1,200 pairs are then reserved for estimating the mixture weights, and 600 pairs for the evaluation. The results are averaged over 10 different such splits.

In [Dorji et al., 2011] authors present a methodology that uses both statistical and linguistic methods to extract and select relevant compound as well as single Field Associated (FA) Terms from domain-specific corpora. An FA Term is defined as the minimum word or phrase that serves to identify a particular field. They use specially developed POS patterns to extract FA Term candidates from domain-specific corpora using a sliding window of ten words. Relevant FA Terms are then selected by corpora comparison and using a unique series of statistical formulas based on tf-idf.

Some approaches try to extract the collocations in a fixed size window (e.g. five words) based on lexical dependency of words. Collocations are linguistic phenomena that occur when two or more words appear together more often than by chance and whose meaning often cannot be inferred from the meanings of its parts [Petrović et al., 2010]. Xtract, a terminology extraction system which identifies lexical relations in the large corpus of English texts, avoids this problem by considering the relative positions of co-occurrences [Smadja, 1993]. In Xtract, pairwise lexical relations are first retrieved using only statistical information. After identification of multiple-word combinations and complex expression, by using the parsing and statistic technique, the found collocations are filtered.

More general than a collocation is the term word n-gram which denotes any sequence of n words. Extracting collocations usually proceeds by assigning each candidate n-gram a numeric value indicating how strongly the words within the n-gram are associated with each other [Petrović et al., 2010]. The higher this value, the more likely that then-gram is a collocation. The functions used to assign these values are called lexical association measures. The most known measures are those used in Information theory like Information Mutual and Dice value. [Pecina and Schlesinger, 2006] focus on extending these measures to make them suitable for extracting longer collocations than bi-grams. Bi-grams are also used in [Tan et al., 2002] as index-term to improve the performance of the text classification. Usually, the use of multiword expressions [Vincze et al., 2011] in text-mining process improves the quality of NLP tasks such as parsing [Constant et al., 2012], error checking in texts [Nazar and Renau, 2012, Dale et al., 2012], and so forth.

Finally, in order to evaluate the adequacy of candidate terms, statistical methods are generally associated with syntactic approaches [Daille, 2003]. These methods are based on statistical measures such as information gain to validate an extracted candidate as a term. Among these measures, the occurrence frequency of candidates is a basic notion.

4.2 Discussing the Background Methods.

In the domain of terminology extraction, most of approaches are based on a combination of some main methods like use of syntactic patterns or statistic measures. Many studies compare different techniques of terminology extraction and their performances. But most of these studies are tested on classical texts written in a natural language. Most of corpora used in the experiments of these approaches are consistently structured. Moreover, this textual data complies with NL grammar. However, in our context, due to the characteristics of logs, these methods have to be adapted to ensure that they are relevant for log files.

For instance, as we have previously seen, in the context of log files, there are some difficulties and limitations for applying grammatical tagging and hence using the syntactic pattern on such textual data. Indeed, the classic techniques of POS tagging are normally developed and trained using texts written in a standard natural language, such as journals. They are hence based on standard grammar of natural language in order to determine the grammatical role of words. For instance, they consider that a sentence ends with a full-stop while this is not the case in the log files that we handle. More specifically, in these log files, sentences and paragraphs are not always well structured.

Moreover, there are also some difficulties in using the statistic methods to evaluate and validate the candidate terms. The statistical methods used in classical term extraction methods cannot be applied to log files as they are. Indeed, statistical approaches can cope with high frequency terms, but tend to miss low frequency ones [Evans and Zhai, 1996]. Information is seldom redundant according to the characteristics of log files. Therefore, the domain terms often have very low occurrence frequency. Thus, in our context, we cannot use classic statistical measures which are often relevant to validate the frequent terms.

In the next section, we develop our approach of terminology extraction from log files. Within our approach, we explain how to pre-process the log files in order to prepare them to apply NLP methods. We describe how to adapt a POS tagger to the characteristics of log files. Then, we use a syntactic-based method to extract candidate terms. We finally propose an extended statistic measure and a term evaluation protocol by considering the specificities of log files. Using these adapted methods and the proposed evaluation protocol we overcome the difficulties seen in the extraction of terms in log file corpus.

5 Exterlog: EXtraction of TERminology from LOGs

Our approach, EXTERLOG, is developed to extract terminology in log files [Saneifar et al., 2009]. EXTERLOG consists of two main phases:

- Extraction of terms (text mining approach)
- Filtering relevant terms (Web mining approach)

Figure 2 shows the global processing chain of our approach. In the first phase, i.e. Extraction of Terms, after normalizing the log files by applying adapted and relevant methods (explained in Sections 5.1.1, 5.1.2), we extract co-occurrences as terminological candidates. These candidates will be evaluated in the next phase, i.e. Filtering in order to select the most relevant terms.

5.1 Extraction of Terms

The extraction process firstly involves normalization, preprocessing of log files and grammatical tagging of words. Then, the normalized logs are used in the co-occurrence extraction process. We detail the different steps of term extraction in the following sections.

5.1.1 Preprocessing & Normalization

The heterogeneity of log files can impact the performance of information extraction methods. In order to reduce the data heterogeneity and prepare them to extract terminology, we apply some preprocessing and normalization methods on the logs. The normalization task mainly concerns data representation formats and log files structure. In order to limit ambiguity in the structure and data representation format, we identify the same punctuations and symbols which are

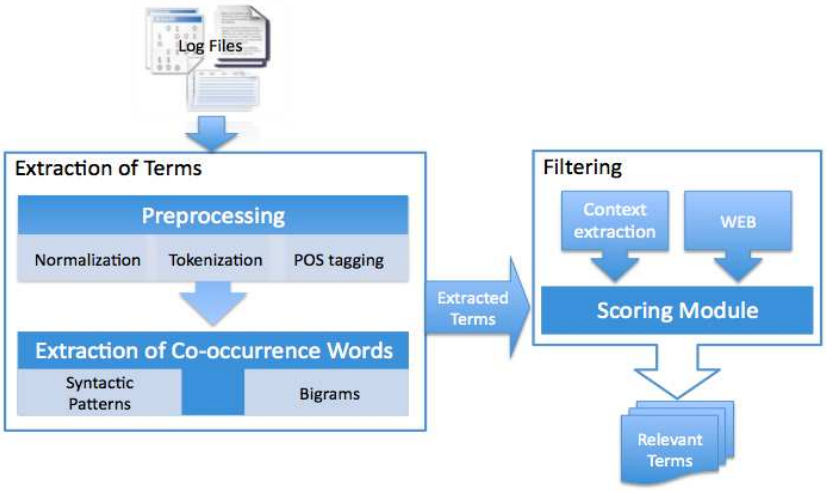

Figure 2: Processing chain of EXTERLOG based on two phases: (1) Terminology Extraction, (2) Filtering of terms with ranking functions.

used to represent different notions. According to the log files, we define some special rules that can be applied to distinguish the role of each symbol despite the fact that the symbol can be used for different reasons. For instance, we finally automatically distinguish lines representing a table header from the lines which separate different parts in a log file. Once the structural role of each symbol is identified, we replace them with a single notation form. There is less ambiguity and less common symbols used for different notions after the normalization process. This normalization streamlines the structure of log files produced by different tools.

Then we tokenize the texts of log files, while 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 the two words "`Circuit4`" and "`LED3`". We thus define some tokenization rules which define the border of words in different cases. These rules define when the system has to tokenize words detected in log files.

5.1.2 Grammatical and Structure Tagging

To identify the role of words in the log files, we use the BRILL rule-based POS tagging method [Brill, 1992]. As described in Section 4, existing taggers like BRILL which are trained on general language corpora give inconsistent results on specialized texts like log files. [Amrani et al., 2004] propose a semi-automatic approach for tagging corpora of speciality. They built a new tagger which modifies the base of rules obtained by the BRILL tagger and adapts it to a corpus of speciality.

Since the classic rules of BRILL are not relevant to log files, we have to adapt the BRILL tagger. For example, a word beginning with a number is considered as "*cardinal*" by BRILL, while there are many words like `12.1vSo10` in log files that must not be labelled as "*cardinal*". Therefore, in order to take such problems into account, we adapted BRILL to the context of log files by introducing new *contextual* and *lexical* rules. These rules are defined after an in-depth analysis of texts of log files. These new contextual and lexical rules represent grammatical rules existing in log files. They also determine exceptions. For example, we replaced the existing rule in BRILL "all terms beginning with a number are cardinal" with a new one which implies that a term is a cardinal if it does not contain a letter.

```
Leakage/NNP Internal/NNP Net/JJ Switching/VBG
Instance/NNP Cell/NNP Power(nW)/NNP Power(nW)/NNP Power(nW)/NNP Power(nW)/NNP
-----/TH
aes_cipher_top/NNM --/: 901234/CD --/: 346/CD --/:
---Line---/NEWLINE
Timing/NN
-----/SPL
Warning/NNP :/: Possible/JJ timing/NN problem/NNS have/VBP been/VBN detected/VBN in/IN this/DT design/NN
---Line---/NEWLINE
Slack/NNP Endpoint/NNP Cost/NN Group/NNP
-----/TH
+XXps/NNS sa22_reg[7]/D/NNM default/NN
-XXps/NNM sa22_reg[7]/D/NNM default/NN
---Line---/NEWLINE
---Line---/NEWLINE
Clock/NNP Description/NNP
-----/SPL
---Line---/NEWLINE
Clock/NNP Clock/NNP Source/NN No/DT of/IN
Name/NN Period/NN Rise/NN Fall/NNP Domain/NNP Pin/Port/NNP Register/NNP
-----/TH
clk/NN xx/NN xx/NN xx/NN domain_1/NNM clk/NN xx/NN
```

Figure 3: Part of a log file after applying the preprocessing and tagging methods

The log file structure could contribute important information for extracting relevant patterns in future work. Therefore, we preserve the structure of files during grammatical tagging. For this purpose, we introduce new tags, called "*Document Structure Tags*" representing different structures in log files. For example, the tag "\TH" represents table headers, or "\SPL" represents the lines separating different data blocks in log files.

In order to perform structural tagging, we determine the special structures and symbols in log files and normalize them based on some defined rules during the normalization task. Then they are identified during the tagging process by the new specific "contextual rules" defined in BRILL. We finally get the logs tagged by the grammatical roles of words and also by labels that help to determine the structure of logs. Figure 3 shows the log file shown in Figure 1(a) after applying the preprocessing, normalization and tagging methods. The structure tags are coloured in Figure 3.

5.1.3 Extraction of Co-occurrences

We look for co-occurrences in the log files with two different approaches:

1. Using defined *part-of-speech* syntactic patterns
2. Without using syntactic patterns

The first approach consists of filtering words according to syntactic patterns. The syntactic patterns determine adjacent words having the defined grammatical roles. Syntactic patterns are used by [Daille, 2003] to extract terminology. For complex term identification, [Daille, 2003] defines syntactic structures which are potentially lexicalisable. As argued by [Daille, 2003], base structures of syntactic patterns are not frozen structures and they accept variations. We call the co-occurrences extracted by the first solution, which is based on the syntactic pattern, "POS-candidates". According to the terms found in our context, the syntactic patterns that we use to extract POS-candidates from log files are:
"\JJ - \NN" (Adjective-Noun),
"\NN - \NN" (Noun-Noun).

Co-occurrences extracted by the second approach are called "bigrams". A bigram is extracted as a series of any two adjacent relevant words[1]. Bigrams are used in NLP approaches as representative features of a text [Tan et al., 2002].

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 noise rate. In this method, we do not filter words according to their grammatical roles.

5.2 Filtering of Candidates

There are many extracted terminological candidates due to the size of log files and the large vocabulary of this domain. However, all extracted terms are not necessarily relevant to the domain. Thus, we need to evaluate and score extracted

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

terms according to their relevance to the context. In order to evaluate extracted terms, we develop and extend our evaluation method proposed in [Saneifar et al., 2011]. Here we take the determination of context into account as a factor which can influence the evaluation of extracted terms. Thereafter, we present our evaluation function and then how we determine the context of documents from which the terms are extracted.

5.2.1 Web Mining Ranking

According to the particular features of such data, in spite of the adapted normalization and tagging methods that we have used, some noise exists which result the extraction of irrelevant terms. Moreover, we are focused on a specialized domain where just some terms are really associated with the domain's context. Thus, we evaluate and score the extracted terms according to their relevance to the context. Then we filter the terms having a low score in order to favor the most relevant terms. In order to evaluate the terms, statistical measures are often used in the terminology extraction field (see [Daille, 1996]). The following are the most widely used:

Mutual Information: One of the most commonly used measures to compute a kind of relationship between words composing what is called a **co-occurrence** is Church's Mutual Information (MI) [Church and Hanks, 1990]. The simplified formula is the following where nb designates the number of occurrences of words and pairs of words:

$$MI(x,y) = \log_2 \frac{nb(x,y)}{nb(x)nb(y)}$$

Cubic Mutual Information: Cubic Mutual Information is an empirical measure based on MI that enhances the impact of frequent co-occurrences, which is absent in the original MI [Daille, 1996].

$$MI3(x,y) = \log_2 \frac{nb(x,y)^3}{nb(x)nb(y)}$$

Dice's Coefficient: An interesting quality measure is Dice's coefficient. It is defined by the following formula based on the frequency of occurrence of terms [Smadja et al., 1996].

$$Dice(x,y) = \frac{2 \times nb(x,y)}{nb(x) + nb(y)}$$

This measure is used in several studies related to noun or verb terms extraction in texts [Roche and Kodratoff, 2009].

These measures are based on the occurrence frequency of terms in the corpus. Scoring terms based on frequencies of terms in the log corpus is not a relevant approach in our context. As we have already explained, techniques based on the *occurrence frequency* of terms in a *corpus* are not relevant to this context as a *representative term* does *not* necessarily have a *high frequency* in log files. Then we score terms according to their occurrence frequency on the Web as a large corpus where the frequency of a term can be representative.

We define the occurrence frequency of a given term on the Web as the number of pages in which the term is present. However, we obtain scores based on the simple count of occurrences of a term on the Web as we are dealing with a specialized domain. Indeed, on the Web, we capture occurrences of terms regardless of the context in which they are seen. Thus, we should only consider occurrences of terms on the Web which are located in the IC design context. We therefore use an extension of described measures called $AcroDef$, for which the context and Web resources are essential characteristics to be taken into account (see [Roche and Prince, 2010]). The formulas presented below, define $AcroDef$ measures, based on MI and Cubic MI respectively.

$$AcroDef_{MI}(a^j) = \frac{nb(\bigcap_{i=1}^{n} a_i^j + C)}{\prod_{i=1}^{n} nb(a_i^j + C|a_i^j \notin M_{stop-words})} \quad \text{where } n \geq 2 \tag{1}$$

$$AcroDef_{MI3}(a^j) = \frac{nb(\bigcap_{i=1}^{n} a_i^j + C)^3}{\prod_{i=1}^{n} nb(a_i^j + C|a_i^j \notin M_{stop-words})} \quad \text{where } n \geq 2 \tag{2}$$

In formulas (1) and (2), we have:

- $\bigcap_{i=1}^{n} a_i^j$ represents the set of words a_i^j ($i \in [1, n]$) seen as a string (using *brackets* and illustrated as follows: "$a_1^j...a_n^j$"). Then an important point of this formula is that the order of the words a_i^j is taken into account to calculate their dependency.
- M_{stop} is a set of stop-words (prepositions, determiners, etc). Then the pages containing only these words are not taken into account.

The nb function used in the preceding measures represents the number of Web pages provided by a search engine with a given query. Thus, $nb(a_i^j + C)$ stands for the number of pages (i.e. links) returned by applying the query $a_i^j + C$ to a search engine. This query means all words of the term a^j in addition to those of context C. In $AcroDef$, the context "C" is represented by a set of significant words. In our case, for example, for a term x^j like "`atpg patterns`" consisting of two words (i.e. $i = 2$), $nb(atpg \bigcap patterns + C)$ is the number of pages returned by applying " "`atpg pattern`" AND C " as a query to a search engine. Here C is a set of words representing the IC design context.

The $AcroDef_{Dice}$ formula [Roche and Prince, 2010] based on Dice's formula is written as follows:

$$AcroDef_{Dice} = \frac{\left|\{a_i^j + C | a_i^j \notin M_{stop-words}\}_{i\in[1,n]}\right|}{\sum_{i=1}^{n} nb(a_i^j + C | a_i^j \notin M_{stop-words})} \times nb(\bigcap_{i=1}^{n} a_i^j + C) \quad \text{where } n \geq 2 \tag{3}$$

In formula (3), we have:

- $\bigcap_{i=1}^{n} a_i^j$ represents the set of words a_i^j ($i \in [1,n]$) seen as a string.
- M_{stop} is a set of stop-words.
- $|.|$ represents the number of words of the set.

The extracted terms are ranked according to their $AcroDef$ scores. We favor the most ranked terms by filtering those having the lowest $AcroDef$ scores. The choice of words representing the context impacts the results obtained by $AcroDef$. In [Roche and Prince, 2010], context "C" is represented as a set of words (*e.g.* "encryption", "information", and "code" to represent the Cryptography context)[2]. The right and exact choice of the domain has a great impact on the evaluation of the results obtained by $AcroDef$. As described, the main motivation of using $AcroDef$ is to consider only the occurrence of terms on the Web, which are bound to the studied domain. Working on a specialized domain where each log file corresponds to a more specialized sub-domain, the choice of context requires expertise to obtain the best results. Since human expertise is not often available, we aim at selecting the most relevant words representing the contextual domain in an automatic way.

$AcroDef$ is close to the algorithm PMI-IR (Pointwise Mutual Information and Information Retrieval) described in [Turney, 2001]. This method queries the Web via the AltaVista search engine to determine appropriate synonyms to a given query. For a given word, noted *word*, PMI-IR chooses a synonym among a given list. These selected terms, noted $choice_i$, $i \in [1,n]$, correspond to the TOEFL questions. The aim is to compute the $choice_i$ synonym that gives the better score. To obtain scores, PMI-IR uses several measures based on the proportion of documents where both terms are present. Turney's formula is given

[2] In this section, we use simply the term "context" as the set of words representing it.

below (formula (4)): It is one of the basic measures used in [Turney, 2001]. It is inspired from Mutual Information.

$$score(\ choice_i\ ) = \frac{nb(\ word\ NEAR\ choice_i\ )}{nb(\ choice_i\ )} \tag{4}$$

– $nb(x)$ computes the number of documents containing the word x,
– $NEAR$ (used in the 'advanced research' field of AltaVista) is an operator that precises if two words are present in a 10 words wide window.

With this formula (4), the proportion of documents containing both *word* and $choice_i$ (within a 10 words window) is calculated, and compared with the number of documents containing the word $choice_i$. The higher this proportion is, the more *word* and $choice_i$ are seen as synonyms. More sophisticated formulas have also been applied: They take into account the existence of negation in the 10 words windows. For instance, the words 'big' and 'small' are not synonyms if, in a given window, a negation associated to one of these two words has been detected, which is likely to happen, since they are antonyms (opposite meanings).

In our approach, there are two important improvements: (1) we use and compare different statistical measures (i.e., MI, MI3, Dice), and we take into account a context. In the next section, we describe how we select words representing the context.

5.2.2 Context Extraction to Extend Statistical Measures

To specify the words which represent the context of log files, we need to select the most significant words occurring in the log files. For this task, we use a basic measure of Information Retrieval domain: *tf-idf* function.

5.2.2.1 Basic measure to select context

tf-idf scoring function measures the relevance of words to the domain in which they appear [Salton and Buckley, 1987]. This measure is based on the hypothesis that a significant word of a domain is frequent in the text of that domain, but less frequent in the text of other different domains.

In a corpus consisting of different documents, the number of times a term occurs in a document is called the *Term Frequency* (*tf*). Thus, we have *tf*, defined as follows:

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

where $n_{i,j}$ is the number of occurrences of the considered term t_i in document d_j. In order to normalize the *tf* value, we use the sum of the number of occurrences of all terms in document d_j ($\sum_k n_{k,j}$).

Inverse Document Frequency (*idf*) corresponds to the number of documents (in the corpus) which contain the given term. We show below how *idf* is calculated:

$$\mathrm{idf_i} = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

$\mid D \mid$ is the total number of documents in the corpus and $|\{d : t_i \in d\}|$ represents the number of documents (d) where the term t_i appears. Finally, the *tf-idf* score is calculated as:

$$(\text{tf-idf})_{\mathrm{i,j}} = \mathrm{tf_{i,j}} \times \mathrm{idf_i}$$

A high *tf-idf* weight value is obtained by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. The *tf-idf* value for a term will always be greater than or equal to zero.

5.2.2.2 Using tf-idf to identify a context

In order to identify the most significant words of the context by *tf-idf*, we build a corpus of documents including reference documents of Integrated Circuit design and also some documents of other different domains like sports and biology. The diversity of domain of documents in the corpus lets us to identify words which are common in most domains (by using *tf-idf*). These words, which have a low *tf-idf* score, are not relevant for representing the context.

We have chosen two main methods in order to determine which kind of words are more relevant to present the context of log files and thus to be scored by *tf-idf*. In the first method, we only extracted all "*nouns*" from the created corpus and scored them by *tf-idf*. In the second method, in order to identify the most relevant words, we scored all words of the corpus which belong to "*nouns*", "*adjectives*", or "*verbs*" parts-of-speech.

Once the selected words of the corpus are scored using the *tf-idf* measure, from the *IC design documents* we choose n terms having top scores as representing words of the context.

Moreover, the choice of context words is possible based on the selection of the most frequent words of the domain documents[3]. In this case, the *tf-idf* score is not considered and the only factor to select the most representative terms is the number of occurrences of terms in the domain documents.

5.2.2.3 Context and AcroDef

In the *AcroDef* calculation, in order to formulate the query which will be used in a search engine, we can use different search operators (e.g. AND or OR). By using the AND operator, for example, we query pages containing *all* words in

[3] stop-words are filtered

"*C*". However, working on a very specialized domain which contains some more specific sub-domains, we do not get the best results by using an "AND" operator for the words of context.

Actually, the Web is often used in order to enrich a poor context for query expansion tasks [Belkin et al., 2006], or for processing short texts [Cano et al., 2014]. Our method shares this point of view in order to enrich a context by selecting the relevant terms.

Due to the nature of the Web, we argue that pages which are related to a context do not contain all words representing the context. Hence, we look for Web pages containing a given term and *two* or *more* words of the context (*i.e. we use both operators "OR" and "AND"*).

All of these methods have been experimented to choose the best approach of context determination. The results of experiments are presented in Section 6.

6 Experiments

We evaluate our approach in two main directions:

- Evaluation of both chosen approaches for co-occurrence extraction (see section 6.1)
- Evaluation of $AcroDef$ (see section 6.2)

In all experiments, the log corpus is composed of logs of five IC design levels. In fact, in Integrated Circuit production, there are several design levels. Log files generated at each level contain different information, structures, and vocabularies. For each level, we considered two logs generated in different design conditions. The size of the log corpus is about 950 KB while each log file contains 10000 words in average.

6.1 Evaluation of Co-occurrence Extraction

We tested two different methods in order to extract terminology from logs:

- Extraction of co-occurrences based on syntactic patterns (POS candidates)
- Extraction of co-occurrences based on bigrams of words

Here we deal with the evaluation of the relevance of each method. Hence, in order to analyze the performance of both approaches, we evaluate the terms extracted by each one. At this stage, we prefer an automatic evaluation of candidates (extracted terms) for two reasons: (1) The huge number of candidates, especially those extracted by the second method, make human expertise difficult; (2) Since our goal, at this level, is just to evaluate the performance of each method and not to measure the real precision of our approach (see section 6.1.1). However, in order to accurately measure performance of our approach, a validation by a human expert, is carried out to complete the automatic validation (see section 6.1.2).

6.1.1 Automatic Evaluation

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 integrated circuits design level, we use certain documents, which explain the principles and the details of the design tools. We use these documents as "*reference experts*" in an automatic validation context. In fact, if a term extracted from logs (*i.e. a candidate*) is used in the reference documents, we can consider it as being a valid domain term.

Note that, to extract the domain terminology, we have to use log files and not reference documents because there are some terms that do not appear in reference documents according to their nature. Hence, we could use references as a validation tool, but not as the basis of the domain terminology.

Moreover, in order to assess whether the number of occurrences of terms in log files is significant information, we perform a pruning task. We filter the extracted terminological candidates based on their frequency of occurrences in the logs. Therefore, we select terminological candidates having an occurrence frequency of at least 2 (*i.e.* we do not consider terms that have occurred just once in log files).

Finally, we calculate the precision for the extracted candidates as shown below:

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

Table 1 shows the precision of POS-candidates and bigrams before and after pruning. At this experimental level, in order to evaluate the candidate, the precision is the most adapted measure regarding our context. Indeed, this measure

		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
After	Precision	81.1	10.1	18.0	5.0	37.2	5.9	27.3	7.1	37.1	5.5

Table 1: Precision of terminological candidates before and after pruning based on reference documents and automatic evaluation.

gives the general trend of the quality of terms extracted by each method. Note that to calculate a perfectly adapted precision, we have to manually evaluate all terms proposed by EXTERLOG.

Comparison of terminological candidates with the reference terms (see Tab.1) 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 that of bigrams. Our experiments show that an effort in normalization and POS tagging tasks is quite useful for extracting relevant terms.

At this stage, we do not calculate the Recall because there is not a set of domain terms to be used as reference. The building of such set of domain terms from log files which can be used as reference in the Recall calculation requires a manual and complete extraction of the domain ontology by domain experts. Such a task is very expensive.

Note that the pruning of terms based on their occurrence frequency in the log corpus does not significantly improve the results. As we have already explained, in our context, terms are not generally repeated in log files. Therefore, a *representative term* does *not* necessarily have a *high frequency* in the log corpus.

6.1.2 Validation by Experts

In order to validate the "automatic evaluation protocol" using the reference documents, we asked two domain experts to evaluate terms. First, extracted terms are tagged by a domain expert as *relevant* or *not relevant* according to the context and their usefulness in the logs. Then another expert reviewed the tagged terms by the first expert.

We calculated the percentage of terms extracted by EXTERLOG and validated using reference documents (automatic evaluation protocol), which are also annotated as relevant by experts. The results show that 84% to 98.1% of the terms validated by our automatic evaluation protocol are really relevant terms according to the experts. 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 the experts. If these terms are not taken into account in the calculation, then 98.1% of the terms are really relevant.

As a conclusion to this experiment, extraction of co-occurrences based on syntactic patterns is more relevant to obtain relevant domain terms. The frequency of occurrences of terms in *log files* is not representative information. Hence, the subsequent experiments are carried out for the terms extracted based on syntactic patterns (i.e. POS candidates).

6.2 Evaluation of $AcroDef$ Measure

6.2.1 How to Evaluate a Ranking Function?

The extracted terms by EXTERLOG from log files are so numerous, which complicates validation by domain experts. Thus, we performed the experiments by selecting a sample of extracted terms into our benchmark. Thus, from the logs of every IC design level, we select the 200 most frequent terminological candidates. Since there are less than 200 extracted terms for some levels, the taken sample consists of 700 terms overall.

6.2.1.1 ROC Curves and AUC

In our experiments, we aim to study the $AcroDef$ ranking function and its ability to give a high score to relevant terms and low score to irrelevant ones. We evaluate the ranking function used to score the terms (*i.e.* $AcroDef$) using ROC curves (Receiver Operating Curve).

The ROC curve depicts the tradeoff between both objectives and represented in the False Positive, True Positive plane. The ideal hypothesis corresponds to point (0,1), with no false positive and 100% true positive examples. A ROC curve allows us to compare the ranking functions (here $AcroDef$) that classify elements of a data-set into both groups, i.e. *positive* and *negative*. It indicates the ability to put the positives before the negatives.

In our case, the ROC curve indicates the ability of $AcroDef$ to give a higher score to relevant terms than to irrelevant ones. An effective ranking function should lead to distributions where positives and negatives are well separated. Using ROC curves, we evaluate how much $AcroDef$ is relevant as a measure to distinguish positive and negative terms.

The area under the ROC curve (AUC – Area Under Curve) is thus viewed as a global measure of the ranking functions. The area under the ROC curve is equivalent to the Wilcoxon rank statistics, the probability of ranking correctly a pair of (positive, negative) examples. So AUC is the area between the curve and the horizontal axis. If we order individuals at random, the AUC will be equal to 0.5.

6.2.1.2 Examples of ROC Curves

We explain, with an example, how ROC curves work. Let L_1 and L_2 be two lists of terms ranked by two different functions. We indicate each term (*element of list*) by "+" (i.e. relevant term) or "−" (i.e. irrelevant term).
$L_1 = \{(+), (+), (-), (+), (-), (-)\}$,
$L_2 = \{(-), (+), (-), (-), (+), (+)\}$

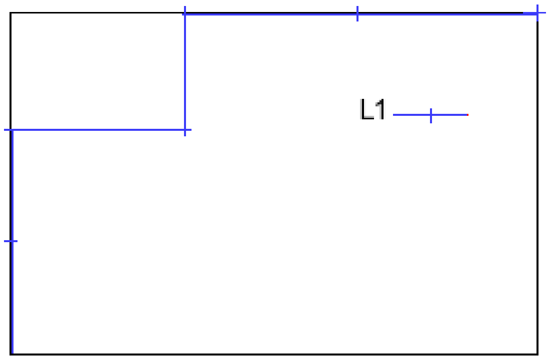

Figure 4: ROC curve obtained from L_1

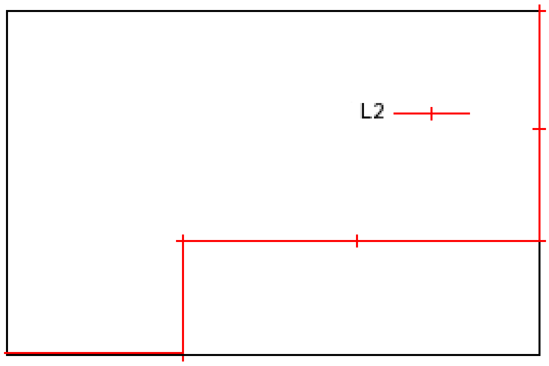

Figure 5: ROC curve obtained from L_2

Since the two lists are ordered with different functions, the terms have different positions. To illustrate the ROC curve, for each "+" we increase the curve one unit in the Y axis direction. Also, for each "−", the curve is continued one unit in the X axis direction. As shown in Figures 4 and 5, the ROC curve corresponding to L_1 is increased on the Y axis more than the ROC curve of L_2. On the other terms, the AUC of L_1 is greater than that of L_2. This shows that the ranking function based on which L_1 is ordered, is more relevant for classifying the positive elements (relevant terms). Moreover, the AUC value of the ROC curve of L_1 is 0.88 when the AUC value of L_2 is 0.22.

6.2.1.3 Characteristics of ROC Curves

The advantage of the ROC curves comes from its resistance to imbalance (for example, an imbalance in number of positive and negative examples). We can illustrate this fact with the following example. Let us suppose that we have 100 examples. In the first case, we have an imbalance between the positive and negative examples with only 1 positive and 99 negative examples. In the second case, we have 50 positive and 50 negative examples. Let us suppose that for these two cases, the positive examples are presented at the top of the list ranked with statistical measurements. In both cases, the ROC curves are strictly similar with AUC = 1.

6.2.2 Criteria to Evaluate

We asked two domain experts to evaluate the terms ranked by $AcroDef$. The terms were at first tagged by a domain expert as *relevant* or *irrelevant* according to the "IC design domain" and their usefulness in the logs. Then, another expert reviewed the tagged terms.

As described in Section 5.2, in order to calculate $AcroDef$ values, we use the Google search engine to capture the number of pages containing one given term and *two* or *more* words of context. With one given term like "CPU time" where C_i $i \in \{1-n\}$ are the context words and we take the five top-ranked words (i.e. $n = 5$), the query used in Google search engine is "CPU time" AND C_1 AND (C_2 OR C_3 OR C_4 OR C_5).

To apply $AcroDef$, we determine the context words C, as described in Section 5.2, in different ways:

- based on *tf-idf* score:
  - top-ranked words belonging to the POS category "noun"
  - top-ranked words belonging to the POS categories "noun", "adjectives", or "verbs"
- based on the *occurrence frequency* of words (stop-words filtered):
  - most frequent words belonging to the POS category "noun"
  - most frequent words belonging to the POS categories "noun", "adjectives", or "verbs"

In order to determine the best context, we test each method of context determination.

6.2.3 Results

6.2.3.1 *Evaluation of AcroDef where the context is determined on the basis of the tf-idf score*

Here, we test the $AcroDef$ function based on using two different contexts obtained by using the *tf-idf* measure. In the first case, we determine the context by

m	AUC_{MI}	AUC_{MI3}	AUC_{Dice}
200	0.50	0.50	**0.58**
300	0.48	**0.64**	0.60
400	0.58	**0.66**	0.63
500	0.60	**0.68**	0.67
600	0.67	**0.72**	**0.72**
700	0.71	**0.75**	0.74

Table 2: AUC obtained at each filtering level based on the *AcroDef* while the context contains just the most ranked nouns (using the *tf-idf* score)

m	AUC_{MI}	AUC_{MI3}	AUC_{Dice}
200	0.53	**0.60**	0.59
300	0.61	**0.70**	0.66
400	0.62	**0.71**	0.68
500	0.66	**0.74**	0.71
600	0.72	**0.75**	**0.75**
700	0.74	**0.77**	0.76

Table 3: AUC obtained at each filtering level based on the *AcroDef* while the context contains the most ranked words (nouns, adjectives, verbs) by using the *tf-idf* score

selecting the most ranked words which present "noun" parts-of-speech. In the second case, the context is determined by choosing the most ranked words from a set of words which belong to "noun", "adjective", or "verb" POS categories. In both cases, the words are ranked by a *tf-idf* score.

We calculate the ROC curves according to different filtering thresholds. That is, the number of top-ranked terms by *AcroDef* which are selected as relevant. We consider six thresholds ($m = 200$, $m = 300$, $m = 400, ..., m = 700$).

Tables 2 and 3 show AUC according to the ROC curves based on $AcroDef_{MI}$, $AcroDef_{MI3}$, and $AcroDef_{Dice}$, while the context is determined by *tf-idf*.

As described above, the parameter m is the filtering threshold. With $m = 500$, for example, we take the 500 top-ranked terms. According to the AUC values, for example, when we use $AcroDef_{MI3}$ and *tf-idf* measures to determine the context, with $m = 500$, if the context is determined by choosing the representative word belonging to *noun*, *adjective*, or *verb* POS categories, it is 74% likely that relevant terms have a higher *AcroDef* score than irrelevant terms. In the same conditions, if the context is represented just by "*nouns*", in 68% of cases relevant terms have a higher *AcroDef* score than irrelevant ones.

According to the results, we see that while the context is determined by words which belong to noun, adjective, or verb POS categories, we have more relevant *AcroDef* functions. This means that this method of context determination is

m	AUC_{MI}	AUC_{MI3}	AUC_{Dice}
200	**0.57**	0.50	0.48
300	0.51	**0.65**	0.59
400	0.52	**0.64**	**0.64**
500	0.58	**0.67**	**0.67**
600	0.68	0.70	**0.71**
700	0.72	**0.74**	**0.74**

Table 4: AUC obtained at each filtering level based on *AcroDef* while the context contains the most *frequent* nouns

m	AUC_{MI}	AUC_{MI3}	AUC_{Dice}
200	**0.56**	0.55	0.53
300	0.50	**0.66**	0.62
400	0.51	0.62	**0.63**
500	0.57	**0.66**	**0.66**
600	0.68	**0.72**	0.70
700	0.72	**0.74**	**0.74**

Table 5: AUC obtained at each filtering level based on *AcroDef* while the context contains the most *frequent* words (nouns, adjectives, verbs)

more relevant than others that use the words belonging just to the "noun" POS category.

6.2.3.2 **Evaluation of** *AcroDef* **while the context is obtained based on the word occurrence frequency**

In this section, we have focused on the study of the use of other methods to determine the context. In the last section, context words were scored by the *tf-idf* measure. But here we choose the most frequent words to represent the context. So, the only factor is the number of occurrences of words in domain documents. As described before, we build two different contexts. The first one contains the most frequent words belonging to the "noun" POS category. The second context contains the most frequent words belonging to the "noun", "adjective", or "verb" POS categories.

Tables 4 and 5 show AUC corresponding to ROC curves based on $AcroDef_{MI}$, $AcroDef_{MI3}$, and $AcroDef_{Dice}$ while the context is determined by selecting the most frequent words.

When the context is determined based on the *occurrence frequency* of words and we are using $AcroDef_{MI3}$, according to the AUC results, if the context is represented by words belonging to *noun*, *adjective*, and *verb* POS categories, it is 66% likely that relevant terms have a higher *AcroDef* score than irrelevant ones (when $m = 500$). While, according to our previous experiment (cf. Tab. 3), in the same conditions, if the context is determined by using *tf-idf*, we have an AUC of 74%.

To conclude, according to the results, the best method to choose the context is to rank words of documents by *tf-idf* measure and select the most ranked words which belong to noun, adjective, or verb POS categories. Moreover, $AcroDef$ calculated based on $MI3$ is more relevant than both other types of $AcroDef$. In our benchmark, in 77% of cases (i.e. with $m = 77\%$), by using $AcroDef_{MI3}$, a relevant term has a higher score than an irrelevant one.

Finally, in the following section, we evaluate the performance of our terminology extraction in order to find the best filtering threshold (i.e. value of m).

6.2.3.3 Performance of term filtering

In these experiments, we use $AcroDef_{MI3}$ and the context contains words belonging to *noun*, *adjective*, and *verb* POS categories (based on the tf-idf measure). These conditions are chosen according to the results of previous experiments. Here we aim at calculating the precision and recall of our approach in terms of relevant term extraction. We focused on determination of the best filtering threshold for our approach. We ranked terms based on their $AcroDef$ score. Then we filter the terms by selecting the top-ranked ones. Terms having low score are filtered (i.e. pruned). The terms are evaluated by two domain experts as relevant or irrelevant. We used classical evaluation measures of data mining and text mining fields, i.e. precision, recall, and F-measure [Hotho et al., 2005]. This evaluation measures are computed for each threshold.

• The **precision** is calculated as a percentage of remaining terms (*after pruning based on* $AcroDef$ *scores*) which are tagged as "relevant" by experts.

$$Precision = \frac{|Terms_{relevant} \cap Terms_{remained}|}{|Terms_{remained}|}$$

$Terms_{relevant}$ = terms validated by experts
$Terms_{remained}$ = terms remaining after filtering

• We calculate **the recall** as the percent of all relevant terms (*in benchmark* scale) which remain after filtering. Actually, with $m = 700$, we do not filter terms. Then all terms are proposed to the experts. Of course, in this case, a lot of noise (e.g. irrelevant terms) is returned. But all relevant terms are given. So the recall is equal to 100%.

$$Recall = \frac{|Terms_{relevant} \cap Terms_{remained}|}{|Terms_{relevant}|}$$

$Terms_{relevant}$ = terms validated by experts in the benchmark
$Terms_{remained}$ = terms remaining after filtering

m	Precision	Recall	F-score
200	86 %	41 %	56 %
300	79 %	57 %	67 %
400	76 %	74 %	75 %
500	72 %	87 %	**79** %
600	66 %	95 %	78 %
700	59 %	100 %	74 %

Table 6: Precision, Recall, and F-score of terms in each level m of filtering

• We have also calculated the **F-measure** as the harmonic average of precision and recall.

$$F - measure = \frac{2 * (Precision * Recall)}{Precision + Recall}$$

Table 6 shows the filtering results with different m values. For instance, with $m = 300$, we take the 300 top ranked terms. The results highlight that a good compromise between Precision and Recall is obtained with $m = 500$. Indeed, by using $AcroDef_{MI3}$ as the ranking function while the context is determined by *tf-idf*, the F-score of our approach is 79% (Precision=72% & Recall=87%) if we take the 500 most ranked terms. To obtain better precision, we have to decrease the filtering threshold, but the recall will decrease.

7 Conclusion

In this paper, we describe a specific type of textual data: Log files generated by tools for integrated circuit design. Since these log files contain multi-source, multi-format, heterogeneous, and changing textual data, NLP and IE methods are not necessarily well suited to extract information.

To extract domain terminology, we extracted the co-occurrences. For that, we apply specific preprocessing, normalization, and tagging methods. To reduce the noise ratio in extracted terms and favor more relevant terms of this domain, we score terms using a Web and context based statistical measure. Then we select the top ranked terms based on their score and filter (pruning) the terms having low score. The experiments show that our approach for terminology extraction from log files, Exterlog, can achieve a F-score equal to 0.79 after filtering terms. Moreover, the $AcroDef_{MI3}$ ranking function is more relevant than other measures for classifying relevant terms.

Finally, we plan to take the terminology extracted using our system into account to enhance information extraction from log files. In fact, within a platform of Information Extraction system, we aim to use extracted terminological knowledge to better determine the features of queries applied to log files. This terminological knowledge will also be used to better represent fragments of log files in order to retrieve those corresponding to a given query. The enrichment of textual data by terminological knowledge is an important aspect of the Information Extraction system.

References

[Adedayo and Olivier, 2015] Adedayo, O. M. and Olivier, M. S. (2015). Ideal log setting for database forensics reconstruction. *Digital Investigation*, 12(0):27 – 40.

[Amrani et al., 2004] Amrani, A., Kodratoff, Y., and Matte-Tailliez, O. (2004). A semi-automatic system for tagging specialized corpora. In *PAKDD*, pages 670–681.

[Belkin et al., 2006] Belkin, N. J., Cole, M., Gwizdka, J., l. Li, Y., j. Liu, J., Muresan, G., Roussinov, D., Smith, C. A., Taylor, A., and j. Yuan, X. (2006). Rutgers information interaction lab at trec 2005: Trying hard. In *In Proceedings of TREC 2005. On-line at http://trec.nist.gov.*

[Bourigault and Fabre, 2000] Bourigault, D. and Fabre, C. (2000). Approche linguistique pour l'analyse syntaxique de corpus. *Cahiers de Grammaire - Université Toulouse le Mirail*, pages 131–151.

[Brill, 1992] Brill, E. (1992). A simple rule-based part of speech tagger. In *In Proceedings of the Third Conference on Applied Natural Language Processing*, pages 152–155.

[Buitelaar et al., 2008] Buitelaar, P., Cimiano, P., Frank, A., Hartung, M., and Racioppa, S. (2008). Ontology-based information extraction and integration from heterogeneous data sources. *Int. J. Hum.-Comput. Stud.*, 66(11):759–788.

[Cano et al., 2014] Cano, A. E., He, Y., and Alani, H. (2014). Stretching the life of twitter classifiers with time-stamped semantic graphs. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, pages 341–357.

[Church and Hanks, 1990] Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. In *Computational Linguistics*, volume 16, pages 22–29.

[Constant et al., 2012] Constant, M., Sigogne, A., and Watrin, P. (2012). Discriminative strategies to integrate multiword expression recognition and parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 204–212, Jeju Island, Korea. Association for Computational Linguistics.

[Daille, 1996] Daille, B. (1996). Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. In *The Balancing Act: Combining Symbolic and Statistical Approaches to Language, MIT Press*, pages 49–66.

[Daille, 2003] Daille, B. (2003). Conceptual structuring through term variations. In *Proceedings of the ACL 2003 workshop on Multiword expressions*, pages 9–16, Morristown, NJ, USA. Association for Computational Linguistics.

[Dale et al., 2012] Dale, R., Anisimoff, I., and Narroway, G. (2012). Hoo 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 54–62, Montréal, Canada. Association for Computational Linguistics.

[Déjean et al., 2005] Déjean, H., Gaussier, E., Renders, J. M., and Sadat, F. (2005). Automatic processing of multilingual medical terminology: applications to thesaurus enrichment and cross-language information retrieval. *Artificial Intelligence in Medicine*, 33:111–124.

[Dey et al., 2005] Dey, L., Singh, S., Rai, R., and Gupta, S. (2005). Ontology aided query expansion for retrieving relevant texts. In *AWIC*, pages 126–132.

[Dorji et al., 2011] Dorji, T., Atlam, E.-s., Yata, S., Fuketa, M., Morita, K., and Aoe, J.-i. (2011). Extraction, selection and ranking of field association (fa) terms from domain-specific corpora for building a comprehensive fa terms dictionary. *Knowledge and Information Systems*, 27:141–161.

[Evans and Zhai, 1996] Evans, D. A. and Zhai, C. (1996). Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 17–24, Morristown, NJ, USA. Association for Computational Linguistics.

[Even and Enguehard, 2002] Even, F. and Enguehard, C. (2002). Extraction d'informations à partir de corpus dégradés. In *Proceedings of 9ème conference sur le Traitement Automatique des Langues Naturelles (TALN'02)*, pages 105–115.

[Facca and Lanzi, 2005] Facca, F. M. and Lanzi, P. L. (2005). Mining interesting knowledge from weblogs: a survey. *Data Knowl. Eng.*, 53(3):225–241.

[Hotho et al., 2005] Hotho, A., Nrnberger, A., and Paa, G. (2005). A brief survey of text mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology.*

[Jiang et al., 2008] Jiang, Z. M., Hassan, A. E., Hamann, G., and Flora, P. (2008). An automated approach for abstracting execution logs to execution events. *Journal of Software Maintenance*, 20(4):249–267.

[Moldovan et al., 2003] Moldovan, D., Paşca, M., Harabagiu, S., and Surdeanu, M. (2003). Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems*, 21:133–154.

[Nazar and Renau, 2012] Nazar, R. and Renau, I. (2012). Google books n-gram corpus used as a grammar checker. In *Proceedings of the Second Workshop on Computational Linguistics and Writing (CL&W 2012): Linguistic and Cognitive Aspects of Document Creation and Document Engineering*, pages 27–34, Avignon, France. Association for Computational Linguistics.

[Pecina and Schlesinger, 2006] Pecina, P. and Schlesinger, P. (2006). Combining association measures for collocation extraction. In *Proceedings of the COLING/ACL on Main Conference poster sessions*, COLING-ACL'06, pages 651–658, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Penas et al., 2001] Penas, A., Verdejo, F., and Gonzalo, J. (2001). Corpus-based terminology extraction applied to information access. In *In Proceedings of Corpus Linguistics 2001*, pages 458–465.

[Petrović et al., 2010] Petrović, S., Šnajder, J., and Bašić, B. D. (2010). Extending lexical association measures for collocation extraction. *Comput. Speech Lang.*, 24:383–394.

[Roche et al., 2004] Roche, M., Heitz, T., Matte-Tailliez, O., and Kodratoff, Y. (2004). Exit: Un système itératif pour l'extraction de la terminologie du domaine à partir de corpus spécialisés. In *Proceedings of JADT'04 (International Conference on Statistical Analysis of Textual Data)*, volume 2, pages 946–956.

[Roche and Kodratoff, 2009] Roche, M. and Kodratoff, Y. (2009). Text and Web Mining Approaches in Order to Build Specialized Ontologies. *Journal of Digital Information*, 10(4):6.

[Roche and Prince, 2010] Roche, M. and Prince, V. (2010). A web-mining approach to disambiguate biomedical acronym expansions. *Informatica*, 34(2):243–253.

[Salton and Buckley, 1987] Salton, G. and Buckley, C. (1987). Term weighting approaches in automatic text retrieval. Technical report, Ithaca, NY, USA.

[Saneifar et al., 2009] Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., and Roche, M. (2009). Terminology extraction from log files. In *DEXA '09: Proceedings of the 20th international conference on Database and Expert Systems Applications.* Springer-Verlag.

[Saneifar et al., 2010] Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., and Roche, M. (2010). Passage retrieval in log files: an approach based on query enrichment. In

Proceedings of Advances in Natural Language Processing, 7th International Conference on NLP, IceTAL'10, pages 357–368. Springer-Verlag.

[Saneifar et al., 2011] Saneifar, H., Bonniol, S., Laurent, A., Poncelet, P., and Roche, M. (2011). How to rank terminology extracted by exterlog. In *In Series: Communications in Computer and Information Science (CCIS), Springer-Verlag, revised selected paper of KDIR'09*, pages 121–132.

[Silvescu et al., 2001] Silvescu, A., Reinoso-castillo, J., and Honavar, V. (2001). Ontology-driven information extraction and knowledge acquisition from heterogeneous, distributed, autonomous biological data sources. In *In Proceedings of the IJCAI-2001 Workshop on Knowledge Discovery from Heterogeneous, Distributed, Autonomous, Dynamic Data and Knowledge Sources*.

[Smadja, 1993] Smadja, F. (1993). Retrieving collocations from text: Xtract. *Comput. Linguist.*, 19(1):143–177.

[Smadja et al., 1996] Smadja, F., McKeown, K. R., and Hatzivassiloglou, V. (1996). Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1–38.

[Tan et al., 2002] Tan, C.-M., Wang, Y.-F., and Lee, C.-D. (2002). The use of bigrams to enhance text categorization. In *Inf. Process. Manage.*, pages 529–546.

[Thattil, 2008] Thattil, Jiny Antony [IN], B. I. (2008). Context-based information retrieval. Patent Application. US 2008/0201350 A1.

[Turney, 2001] Turney, P. (2001). Mining the Web for synonyms: PMI–IR versus LSA on TOEFL. *Proceedings of the 12th European Conference on Machine Learning (ECML), LNCS*, 2167:491–502.

[Vincze et al., 2011] Vincze, V., Nagy T., I., and Berend, G. (2011). Multiword expressions and named entities in the wiki50 corpus. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 289–295, Hissar, Bulgaria. RANLP 2011 Organising Committee.

[Voorhees, 1994] Voorhees, E. M. (1994). Query expansion using lexical-semantic relations. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69, New York, NY, USA. Springer-Verlag New York, Inc.

[Yamanishi and Maruyama, 2005] Yamanishi, K. and Maruyama, Y. (2005). Dynamic syslog mining for network failure monitoring. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 499–508, New York, NY, USA. ACM.