Science of Computer Programming 105 (2015) 26–43

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

IRISH: A Hidden Markov Model to detect coded information islands in free text

Luigi Cerulo [a,c,*], Massimiliano Di Penta [b], Alberto Bacchelli [d], Michele Ceccarelli [a,e], Gerardo Canfora [b]

[a] Dep. of Science and Technology, University of Sannio, Benevento, Italy
[b] Dep. of Engineering, University of Sannio, Benevento, Italy
[c] BioGeM, Institute of Genetic Research "Gaetano Salvatore", Ariano Irpino (AV), Italy
[d] Dep. of Software Technology, Delft University of Technology, The Netherlands
[e] QCRI – Qatar Computing Research Institute, Doha, Qatar

ARTICLE INFO

Article history:
Received 20 December 2013
Received in revised form 31 July 2014
Accepted 20 November 2014
Available online 16 December 2014

Keywords:
Hidden Markov Models
Mining unstructured data
Developers' communication

ABSTRACT

Developers' communication, as contained in emails, issue trackers, and forums, is a precious source of information to support the development process. For example, it can be used to capture knowledge about development practice or about a software project itself. Thus, extracting the content of developers' communication can be useful to support several software engineering tasks, such as program comprehension, source code analysis, and software analytics. However, automating the extraction process is challenging, due to the unstructured nature of free text, which mixes different coding languages (*e.g.*, source code, stack dumps, and log traces) with natural language parts.

We conduct an extensive evaluation of IRISH (InfoRmation ISlands Hmm), an approach we proposed to extract islands of coded information from free text at token granularity, with respect to the state of art approaches based on island parsing or island parsing combined with machine learners. The evaluation considers a wide set of natural language documents (*e.g.*, textbooks, forum discussions, and development emails) taken from different contexts and encompassing different coding languages. Results indicate an F-measure of IRISH between 74% and 99%; this is in line with existing approaches which, differently from IRISH, require specific expertise for the definition of regular expressions or grammars.

1. Introduction

Mailing lists and issue tracking systems are communication tools widely adopted by developers to exchange information about implementation details, high-level design, bug reports, code fragments, patch proposals, erroneous behavior, *etc.* Some software projects, such as Linux Kernel, adopt mailing lists as the main tool for managing and storing software documentation [1]. Further communication means, very popular nowadays, are web forums, such as StackOverflow.[1]

* Corresponding author at: Dep. of Science and Technology, University of Sannio, Benevento, Italy.
E-mail addresses: lcerulo@unisannio.it (L. Cerulo), dipenta@unisannio.it (M. Di Penta), A.Bacchelli@tudelft.nl (A. Bacchelli), ceccarelli@unisannio.it (M. Ceccarelli), canfora@unisannio.it (G. Canfora).

[1] http://stackoverflow.com.

http://dx.doi.org/10.1016/j.scico.2014.11.017
0167-6423/

Communication regularly used to support software development, such as free text development emails, is a very attractive sources of knowledge to support program comprehension and development, because it contains discussions, for example, on how some parts of the system work or should not be used. Communication data has been exploited to develop recommender systems, for example aimed at supporting bug triaging [2], at providing examples of how some APIs should work [3,4], and at aiding program comprehension when documentation is scarce [5].

Nevertheless, extracting relevant information from developers' communication is challenging, due to the mix of different languages adopted by developers, *e.g.*, to describe system failures or code changes. Many development emails, for example, include natural language text interleaved with a detailed stack trace to describe a failure and source code to propose a solution. The email text can appear mixed up with code when it is part of a thread of discussion, in which more developers participate and consider alternative solutions. Also in web forums, such as Stack Overflow, natural language is interleaved with code snippets and stack traces; to improve readability, users are invited to use tags to separate communication from code snippets and stack traces, but this does not happen consistently in practice.[2]

Separating different pieces of information contained in developers' communication improve the quality of data extraction. This is because different parts of the communication require appropriate ways of being analyzed. Just to make an example, sentences expressed in natural language would benefit of analysis performed using Information Retrieval (IR) techniques or maybe natural language parsing, whereas source code fragments should be analyzed using parsers. In some cases—see for example the duplicate bug report detection approach proposed by Wang et al. [6]—different kinds of information such as natural language text and stack traces contribute to the approach accuracy. In other cases, some elements contained in the discussion should be isolated. This could be for instance the case when one wants to mine source code snippets contained in developers' communication.

On the one hand, many software engineering approaches [7,8] have treated free text using traditional IR models such as Vector Space Models (VSM) [9] or Latent Semantic Indexing (LSI) [10]. Although this simplification works well for pure natural language documents, it may easily fail for software engineering artefacts [11], where free text is interleaved with source code, stack traces and other elements. On the other hand, in recent years authors have proposed alternative approaches to treat text based on island parsers, regular expressions, and supervised learning [12,13]. Recently, Bacchelli et al. [11] proposed a hybrid approach, combining island parsers and machine learning. Such an approach outperforms the use of the two techniques in isolation, when not well formed languages (*e.g.*, noise and random characters) appear together with more structured ones.

This paper describes Irish (InfoRmation ISlands Hmm), an approach that we initially proposed in our previous work [14], based on Hidden Markov Models (HMM) to extract islands of coded information from free text at token granularity. Tokens are particles of the text, such as natural language words, programming language keywords, digits, and punctuation. In Irish, we consider the sequence of tokens of a textual document (*e.g.*, a development email) as the emission generated by the *hidden states* of an HMM. Hidden states are adopted to model a specific coded information content, *e.g.*, source code and natural language text. We adopt the Viterbi algorithm [15] to search for the path that maximizes the probability of switching among hidden states. Such a path allows us to classify each observed token in the corresponding coded information category. If appropriately modeled with hidden states and given a proper set of training examples, the approach can, in principle, include an arbitrary number of different text interleaved languages, for example stack traces, patches, and markup languages.

The specific goal of this paper is to provide an extensive evaluation of Irish and to highlight its points of strength and weaknesses, by comparing it to the current state-of-the-art, *i.e.*, two methods (PetitIsland and Mucca) proposed by Bacchelli et al. [11], based respectively on island parsing and island parsing and machine learners combined. The contributions of this paper are:

- The evaluation of Irish on two new datasets: (1) the mailing list dataset built and used by Bacchelli et al. [11], and (2) the Stack Overflow dataset used to build a traceability recovery approach [16].
- A direct comparison with the approaches by Bacchelli et al. [11]. To this aim we evaluate the methods proposed by Bacchelli et al. on datasets previously used to evaluate Irish [14].
- A qualitative evaluation of the learning curve necessary to set up both approaches.

Results of the evaluation indicate that, overall, Irish exhibits performance in line with the state-of-the-art approach. However, differently from these approaches, it only requires a manual classification of a training set, rather than writing island grammars.

Structure of the paper Section 2 summarizes backgrounds and Section 3 introduces Irish. Section 4 details the empirical evaluation procedure. Section 5 reports and discusses the obtained results. Section 6 discusses threats to the validity of the evaluation. Section 7 discusses related work about approaches for extracting encoded information from unstructured sources. Section 8 concludes the paper and outlines directions for future work.

[2] Performing manual analysis of the Stack Overflow posts used in this paper, we found approximately 5–10% of code tags to not enclose a piece of code or named code entity.