Application of Inductive Logic Programming for Learning ECG Waveforms

Gabriella Kókai[1], Zoltán Alexin[2], and Tibor Gyimóthy[3]

[1] Institute of Informatics, József Attila University
[2] Department of Applied Informatics, József Attila University
Árpád tér 2, H-6720 Szeged, Hungary
e-mail: alexin@inf.u-szeged.hu

[3] Research Group on Artificial Intelligence
Hungarian Academy of Sciences,
Aradi vértanuk tere 1, H-6720 Szeged, Hungary
e-mail: gyimi@inf.u-szeged.hu.

Abstract. In this paper a learning system is presented which integrates an ECG waveform classifier (called PECG) with an interactive learner (called IMPUT). The PECG system is based on an attribute grammar specification of ECGs that has been transformed to Prolog. The IMPUT system combines the interactive debugging technique IDT with the unfolding algorithm introduced in SPECTRE. Using the IMPUT system we can effectively assist in preparing the correct description of the basic structures of ECG waveforms. [4]

Keywords: inductive logic programming, program specialization, syntactic pattern recognition

1 Introduction

In this paper a complex system is presented that is able to classify ECG waveforms described as a combination of primitives. The system helps the user to improve the classifier program if it cannot recognize correctly the given input waveforms.

The system integrates an ECG waveform classifier (called PECG) with an interactive learner (called IMPUT).

The PECG system is based on an attribute grammar specification of ECG waveforms published by Skordalakis et al [9]. The basic idea of PECG, introduced in [3], is to integrate the ECG classifier program (implemented in Prolog) with the Interactive Diagnosis and Testing (IDT) module [7] and a graphic viewer. This integrated tool can recognize whether any modification in the classifier program is needed. If the system cannot analyze the input then the user is helped by the built-in IDT debugger to find the false clause. The learning part of the new system is based on an Inductive Logic Programming (ILP) [4, 5] method (called IMPUT) introduced in [1].

---

[4] This work was supported by the grant OTKA T14228 and the project PHARE TDQM 9305-02/1022 ("ILP2/HUN").

In the PECG system the ECG waveforms are described in Definite Clause Grammars (DCGs). The construction of these grammars for different waveforms is a very sophisticated task, therefore a tool which can assist in this work would be very useful. That was the reason for us to develop an extended version of the PECG system.

The user of this system has to prepare an *over-generalized* grammar in the first step. This grammar may accept not only correct but many incorrect waveforms as well. Of course it is much more easier to prepare such *over-generalized* initial grammar than the correct one. The inputs of the extended PECG system are this initial grammar, some background knowledge, positive and negative examples of ECG waveforms to be inferred. The system uses an ILP learning method to improve the inital grammar such that the modified grammar accepts each positive example but no negative ones.

In Section 2 we briefly describe the integrated system Sections 3 contains a brief summary and comments on possible future studies.

2 The architecture of the learning system

In this section we give a overview of the extended PECG system which involves an integration of PECG with the IMPUT interactive learning system. The structure of the system can be seen in Figure 1.

The main parts comming from the PECG are the following:

- *the ECG processing module:* this part of the system uses a syntactic approach to recognize an electrocardiographic pattern. In this part Skordalaki's work was used to decompose the waveform, to select the peaks and present the waveform as a string from a alphabet. The classifier module recognizes the ECG waveforms from their linguistic representation
- *the IDT module:* this part is an interactive debugging and testing tool that is based on the Shapiro's Interactive Diagnosis Algorithm [8] and Category Partition Testing Method (CPM) [6]. It can be divided into three submodules. The *test database* module generates the initial test database from a CPM specification. In the *testing* part a predicate can be chosen and the user can give a representing element of the testframe generated for this predicate. Then he or she can decide whether, for the given input, the output is correct or not. The *debugging* part is based on the idea that if the program has already been tested then the test results can be directly applied without asking the user *difficult* questions.
- *the graphic viewer module:* this part makes visible the current subpatterns being analyzed.

The learning part of the new system consists of two main parts. The specialization algorithm comes from the SPECTRE [2] system while the interactive debugger part is imported from the IDT.

The main idea of IMPUT is that the identification of the next clause to be unfolded has a crucial importance in the effectiveness of the specialization

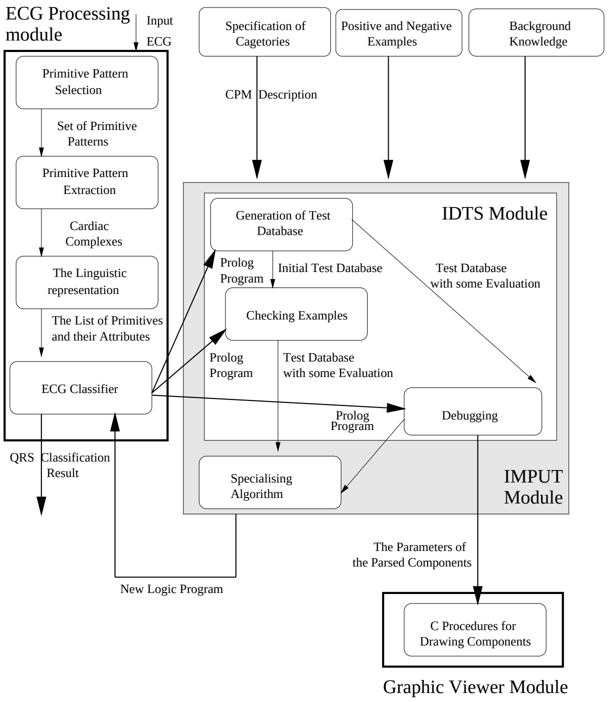

Fig. 1. The structure of the extended PECG system

process. The specialization steps are invoked by the negative examples that are covered by the target predicate. We assume that when a negative example is covered by the current version of the program, then there is at least one clause which is responsible for this incorrect covering. IMPUT uses the IDT debugging algorithm to identify a buggy clause of the program. The clause identified in this process will be unfolded in the next step of the specialization algorithm.

3 Conclusion, Further Work

In this paper an integrated system has been presented which can classify and learn ECG waveforms. Combining the IMPUT method with the PECG system the integrated system was able to suggest correct clauses to replace the *buggy* clauses recognized during the debugging process. In general the graphic viewer helps to find bugs, and in our case shows the ECG subpattern is currently being analyzed.

The main advantage of our system comparing with Skordalakis' approach is that our integrated tool can positively assists the developer (user) working with the ECG classifier program because the program itself is "self-correcting".

Applying the IMPUT learning method to the ECG domain we found that correct DCG programs can be inferred for the basic structures of complex ECG waveforms. In these approaches the user only needs to prepare very general initial DCG programs. The extended PECG system is able to specialise these programs into the correct ones in a very effective way. .

The current version of the PECG system is about 3500 lines in SICStus Prolog, and 500 lines written in C for the implementation of the graphic viewer. The IDT module is about 1400 Prolog lines, containing 150 predicates while the classifier program is more than 2000 lines in Prolog. The IMPUT system itself contains more than 800 lines without the IDT module. [5]

At present the PECG system does not contain any preprocessing module for the linguistic representation of ECG waveforms. To make our system a complete ECG recognizer we plan to implement these preprocessing modules. We also plan to ensure an interactive connection between the graphic viewer and the user. It means that the user may control the analyzing process from the graphic window by marking the subparts of the waveforms to be analysed.

References

1. Alexin, Z., Gyimóthy, T., Boström, H.: Integrating Algorithmic Debugging and Unfolding Transformation in an Interactive Learner in Proceedings of the 12th European Conference on Artificial Intelligence ECAI-96 ed. Wolfgang Wahlster, Budapest, Hungary (1996) 403–407 John Wiley & Son's Ltd. 1996.
2. Boström, H., Idestam-Almquist, P.: Specialization of Logic Programs by Pruning SLD-trees., Proc. of the Fourth International Workshop on Inductive Logic Programming (ILP-94) Bad Honnef/Bonn Germany September 12-14. (1994) 31–47
3. Kókai, G., Alexin, Z., Gyimóthy, T.: Classifying ECG Waveforms in Prolog Proc. of the Fourth International Conference on The Practical Application of PROLOG (PAP96) London, United Kingdom April 23-25, (1996) 193–221
4. Lavrač, N., Džeroski, S.: Inductive Logic Programming: Techniques and Applications Ellis Horwood, (1994)
5. Muggleton, S., De Raedt, L.: Inductive Logic Programming: Theory and methods, Journal of Logic Programming 19 (20) (1994) 629–679
6. Ostrand, T. J., Balker, M. J.: The Category-Partition Method for Specifying and Generating Functional Tests CACM 31:6 June (1988) 676–686
7. Paakki, J., Gyimóthy, T., Horváth T.,: Effective Algorithmic Debugging for Inductive Logic Programming. Proc. of the Fourth International Workshop on Inductive Logic Programming (ILP-94) Bad Honnef/Bonn Germany September 12-14. (1994) 175–194
8. Shapiro, E. Y.: Algorithmic Program Debugging MIT Press (1983)
9. Skordalakis, E.: ECG Analysis in Syntactic and Structural Pattern Recognition Theory and Applications ed. Bunke, H. and Sanfeliu, A. World Scientific (1990) 499–533

This article was processed using the LaTeX macro package with LLNCS style

[5] The implementation language currently is SICStus Prolog Version 3.0 #3 running on a SUN SPARCStation.