

Inductive Inference of Structure for Automated conversion of RTF data to XML

Jon D. Patrick, Dusan Palko, Robert Munro

Sydney Language Technology Research Group
School of Information Technologies, University of Sydney, Sydney, NSW, Australia
{jonpat, dpalko, rmunro}@it.usyd.edu.au

Abstract

This project presents an application tool, FERRET III, which allows users to identify structure in a stream of text. The structure can then be applied to other text streams in the same document or other documents. Experimental work demonstrates the effectiveness of the system for priming an automaton with rules for text segmentation from a few entries in a dictionary and then reapplying them automatically throughout the remainder of the dictionary. The system greatly enhances the speed at which entries can be segmented for conversion into other mark-up codes, such as XML, representing a significant labour saving mechanism compared to manual mark up. Key features of the software design are ease of use, fast performance and robustness of the system.

1 INTRODUCTION

Automated learning and discovery by inductive inference of the inherent meaningful structure of an arbitrary stream of text is a very complex task, both on a theoretical level and from the perspective of designing and implementing a workable software solution. The general solution will inevitably be complex because it not only involves an element of statistical modeling but also has to achieve the requirements of flexibility, ability to handle a large volume of data. Since the software system is intended in gen-

eral for users not technically sophisticated, user-friendliness and ease of operation are also key requirements.

2 Problem statement

The problem statement is to develop a software tool for automated learning which derives an inherent structure in a collection of data records. The tool is to provide a user with full functionality to read a data source file and specify an inherent structure present in the data. The application should be able to record and save the knowledge derived from user actions as they prime the structure with rules. The rules should be part of a mechanism of easy and straightforward application of a powerful statistical apparatus for further automated inductive inference of structure. The case study on which the software has been tested is an English-Basque multilingual dictionary stored in an RTF format. A study of the literature shows little attention to building software to support rapid inference of structure in dictionaries, although many general inferencing strategies have been developed in the field of Machine Learning.

3 Architectural Design of FERRET

The processing problem required an approach from Machine Learning where it is necessary to learn the structure of the data by analysing features in the data stream. We have previously attempted to solve this problem from a bottom-up perspective where the data was considered to be a stream of characters and the learners would assemble character by character compound terms (RTF tags) which in turn are subjected to repeated inferences of compound terms. This strategy was first presented by Solomonoff in a general

inductive inference context. However in our case, this approach failed due to the very large variety of combinations of mark-up tags and a lack of user input to guide the direction of a search.

Our second solution, described here, is a top-down approach, where the user indicates positions for subdividing the data records and the system infers a rule to match it. Such an approach exploits high quality human knowledge about the structure of the data stream. However, we wish to overcome the labour intensive process of marking the structure of every single entry in a large document, for example the 22,000 entries in our dictionary. Hence we would rather reuse the rules defined for one entry on all other entries in the document. This approach requires three structural elements: a method for storing the rules described by the user, a method for reapplying a rule defined for one entry on other entries, and a mechanism for allowing the user to express the rules or structure of an entry. The solution to the first and second elements has been to devise a single strategy whereas the third element is an interface design problem ultimately resolved by our own intuitions about the easiest way to perform the functions.

The solution to storing and reapplying the structural rules of the entries is achieved by representing them as transitions in a Finite State Automaton (FSA). An enhancement on this solution is to extend the FSA to a Probabilistic FSA (PFSA) and so capture the frequency at which different rules are exploited throughout the full set of entries in the data stream. This information can be used in turn for a number of other functions, such as inferring a minimal structure for the PFSA, and identifying rarely used structures that suggest errors in either the dictionary organisation or rule annotation.

4 Implementation Outline

As the software is expected to deal with significant amounts of data, that is tens of thousands of dictionary entries, there is a need to save as much CPU resources as possible in order to provide a reasonable balance between performance and user-friendli-

ness. By “reasonable” balance we mean that the user is provided with an easy to use graphical environment yet not be distracted by sluggish performance while the application is performing simple operations.

The system consists of 3 modules. The Parser module converts the initial data from the RTF file format to a standardized and in some way structured form. It transforms the source file into a collection of homogeneous data items. This collection is then passed to the PFSA module. The PFSA module is the core part of the application and it contains a set of methods which provide functionality for priming and learning. The third module is the GUI which provides the mechanisms for the user to insert rules and view the semantic segmentation of the data.

5 Performance

The system performs to its basic specifications. It is elegantly designed with separation of the data from the inferred automata that represents the breakdown of records into semantic fields. This allows for reuse of an automaton with other data records. Further separation of the interface from the two processing modules improves the maintainability of the system and the quality of the design. The User Interface functions are limited by theoretical considerations of the underlying FSA model and will require further development. The tool is usable for the conversion of documents into XML formats according to user-defined semantic fields. Whilst the system may not capture all segmentation desired by a user due to some segmentation not identifiable as RTF tags, it does represent a significant labour saving device compared to total manual mark-up.

The system is written for Java 1.3 Runtime environment.