An interface for Text Mining with the RAP system

Nelson Prazeres Costa

Porto, February 2008
Abstract

This report introduces a dissertation whose main assignment will be the development of an interface to the RAP system to facilitate text mining tasks. The interface will provide the user the results of the RAP algorithm.

With the growing of Inductive Logic Programming and the appearance of several ILP systems, the implementation of a powerful interface to this system, that can improve its potentiality and visibility, is essential to improve the applicability and further success of this system.

The report makes a brief analysis on Data Mining, Inductive Logic Programming, and presents the RAP system. Its purpose is to give a solid background of knowledge to the following dissertation. Requirements and related works are presented. The discussion, work plan and conclusions will introduce the dissertation work and text.
Advisors

Rui Camacho
camacho@fe.up.pt
Faculty of Engineering, University of Porto, Portugal

Lubomir Popelinsky
popel@fi.muni.cz
Faculty of Informatics, Masaryk University, Czech Republic
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Data Mining</td>
<td>2</td>
</tr>
<tr>
<td>3 Text Mining</td>
<td>3</td>
</tr>
<tr>
<td>4 ILP</td>
<td>4</td>
</tr>
<tr>
<td>5 The RAP System</td>
<td>5</td>
</tr>
<tr>
<td>6 RAP Algorithm</td>
<td>6</td>
</tr>
<tr>
<td>7 Prolog</td>
<td>8</td>
</tr>
<tr>
<td>8 The interface</td>
<td>9</td>
</tr>
<tr>
<td>9 Wrappers</td>
<td>9</td>
</tr>
<tr>
<td>10 Discussion</td>
<td>10</td>
</tr>
<tr>
<td>11 Dissertation work plan</td>
<td>11</td>
</tr>
<tr>
<td>12 Conclusion</td>
<td>12</td>
</tr>
</tbody>
</table>
1 Introduction

The development of an interface to text mining using the RAP system presuppose a complete understanding of this system and general know-how about data mining and Inductive Logic Programming (ILP). The theoretical work is therefore essential for the success of the work. In a technical approach, a solid programming experience is necessary.

1.1 Motivation

Analyzing large collections of text is a complex task. Machine Learning (ML) based algorithms have been developed recently that help in the task of Text Mining. One drawback of such algorithms is that they usually require a ML expert to use them adequately. There are usually a lot of parameters to tune that require expertise. This drawback strongly limits their applicability. To increase the use of such algorithm one has to hidden the parameter from the user. That can be achieved with a proper interface.

1.2 Work proposal

The work consists in the design and development of an interface capable of making Text Mining tasks much more easy. The interface will hide the details of the text mining tool, will assist the user in the text mining experiments and will provide an improved presentation of the results. The interface will help the user in: i) the data preparation stage of text mining tasks; ii) develop the required background knowledge and; iii) automatically tune the parameters of the text mining algorithm.

1.3 Report structure

After a briefly presentation of data mining, with its definition, purpose and general applications, a more detailed attention is performed in a particular area of data mining named text mining. ILP (Inductive Logic Programming) is presented, introducing the field in which RAP system operates. RAP system features and algorithm are presented as well as an introduction to Prolog, the language in which the RAP algorithm was written. At last, a general explanation about graphical user interfaces and the role of wrappers in the construction of interfaces above existent source code is described in order to improve the know-how on the work done in this area.
Data mining is defined as the nontrivial extraction of implicit, previously unknown, and potentially useful information from data” [1], mechanized process of identifying or discovering useful structure in data [2] and "the science of extracting useful information from large data sets or databases” [3]. In other words it can be described as the practice of automatically searching large amounts of data for patterns using computational techniques from statistics and pattern recognition. To extract useful information efficiently of a large amount of data, a selection and pre-processing is commonly necessary before the mining.

The concept can be divided into two main areas, propositional (classical) and relational data mining [4]. While in the first case data is stored in single tables and patterns involve intra-tuple relations, in the second, data is stored in multiple tables with patterns involving inter-tuple or inter-table relations [5]. Propositional and relational systems build two types of data mining models: predictive and descriptive models [4]. In predictive data mining, the goal is to build a model that maps an instance to a valid prediction. Descriptive data mining algorithms are designed to discover interesting knowledge from the data, such as frequently occurring patterns or clusters of similar instances [4].

The process of data mining can be generally explained in four major tasks: class identification (groups database objects into similarity subclasses), classification (description of data in a more compact way), dependency analysis (prediction of values of some attributes if knowing the values of others) and deviation detection (discovers deviations from the expectations) [5]. The main methods in data mining are supervised and unsupervised learning and descriptive methods [5].

They are several tools available in the market that are orientated to data mining. The analytic tools WEKA (Waikato Environment for Knowledge Analysis) [6] and YALE (Yet Another Learning Environment), also known as RapidMiner [7], are open-source software that provides powerful mining algorithms and complete visualization tools to extract knowledge through analysis.

Data mining and relational learning can be applied to many fields like bioinformatics, medicine, engineering and management through different forms such as text, web and spatial mining. [8].
3 Text Mining

Text expresses a vast, rich range of information, but encodes this information in a form that is difficult to decipher automatically [9]. The use of text directly to discover heretofore unknown information is a difficult challenge but an evermore appealing area. Note that 90% of the world data is held under unstructured text format.

Text Mining, in which the dissertation will focus, refers to the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources [10] or the extract of useful information from data sources through the identification and exploration of interesting patterns [11]. Text Mining uses techniques from data mining, machine learning, natural language processing, information retrieval and knowledge management [11]. It involves the pre-processing of document collections (text categorization, information extraction, term extraction), the storage of the intermediate representations, the techniques to analyze these intermediate representations (such as distribution analysis, clustering, trend analysis and association rules), and visualization of the results [11]. Therefore classification, filtering, clustering and association of the text information are the main applications of text mining.

Although the rapidly growing in size and importance of this area, great limitations are implicit attached to text mining processing. The majority of information is in a complete unstructured textual form, they are very high number of possible ”dimensions”, inherit to the nature of our languages. Word ambiguity, context sensitivity and noisy data are important challenges to be overlapped.

Processing text data is performed by three major steps: i) modeling semi-structured data, ii) information retrieval from unstructured documents and iii) text mining, with the classifying, clustering and patterns finding across documents. They are several techniques on classifying documents, such as decision trees or naive bayes classifier, on clustering (finding groups of similar documents) and patterns finding. This techniques may be applied independently or together in order to optimize the results.
4 ILP

Inductive Logic Programming (ILP) [Muggleton, 1992] was created in order to solve the major problems of text mining, such as the representation of whole sentences in a structured way, the incorporation of additional information (morphology, morpho-syntactic relations,...) and the impossibility to use an unique method for solving different tasks (e.g., classification and information extraction).

Inductive Logic Programming (ILP) is a research area formed at the intersection of Machine Learning and Logic Programming [12]. Machine Learning (defined as the acquisition of structural descriptions from examples [13]) provides the technical basis of data mining, extracting information from the raw data in databases [13] and Logic Programming refers to a variety of computer languages and execution models based on the traditional concept of Symbolic Logic [14]. It proceeds by proof search according to a fixed strategy [15]. Inductive Logic Programming, covering this two items, is defined as a class of machine learning algorithms where the agent learns a first-order theory from examples and background knowledge [16]. ILP systems develop predicate descriptions from examples and background knowledge [17]. The use of first-order logic programs as the underlying representation makes ILP systems more powerful and useful than the conventional empirical machine learning systems [16]. The examples, background knowledge and final descriptions are all described as logic programs [17]. The advantages of ILP are the possibility to obtain data and hypothesis in first-order logic, the introduction of new information into data and the possibility to the user to incorporate very complex domain specific background knowledge.

Several ILP Systems are available in the market [18], many oriented to performing specific tasks. Softwares such as Aleph [19], Chilin [20], Forte [21], Golem [22], CProgol [23] are implementations of ILP. While Aleph (A Learning Engine for Proposing Hypotheses) integrates numerous functionalities of other ILP Systems in a way to provide easy and personalized knowledge to the user, Progol combines Inverse Entailment with general-to-specific search through a refinement graph [24]. Several other systems are available and well documented [18].
The RAP System

Oriented to be efficient with text data containing long patterns, the goal of RAP project is to implement an any-time system, capable to find (some of) maximal patterns fast (i.e. faster than level-wise algorithms) and eventually find all maximal patterns [25].

RAP is an ILP system for mining maximal frequent patterns in first-order logic. It uses generate-test paradigm for finding first-order clauses which cover at least $N$ examples [26]. The value $N$ is so called minimal frequency threshold given by user [26]. For a given minimal frequency $\text{minfreq}$ RAP returns conjunctions of literals from background knowledge that cover at least $\text{minfreq}$ examples [27]. Three types of search are implemented: random, depth-first search and best-first search [27]. RAP may store the information about infrequent patterns and class distribution of patterns to optimize the system performance [27].

While relational data mining systems execute a huge number of queries on the database, consuming much CPU time [4], in propositional systems, data has to be converted into a single table, where each row describes a specific instance with a fixed set of features or attributes [4]. This conversion process is called propositionalization. Propositionalization, that consists on the conversion of data to an attribute-value form, followed by a propositional learner function [27] is used by RAP system and can be either partial or complete (with/without loss of information respectively). This conversion permit to achieve efficiently frequent patterns that can be successfully used as new boolean features [27].

For mining frequent patterns the level-wise algorithms have been shown successful for many domains. However, this strategy is usually inefficient if the data set contains long patterns, such as biochemical or molecular domain or in (hyper)text [25].
An example of generation of two maximal patterns is above presented. Maximal patterns are constructed for minimal frequency 4 and a random selection of candidates. Each path in a tree represents a pattern. The number in a node is equal to frequency of the pattern. Solid lines represent the maximal pattern. Dashed lines identify possible candidates and dotted lines identify known candidates.

RAP can work in two regimes, GENERATE (a new query is being built step-by-step employing some of the refinement operators implemented), and COMBINE (two or more maximal queries are used for building a prefix of the next maximal query) [25]. The inductive database of RAP consists of three predicates - freq(Query,Vars,Types,Instances) for the frequent patterns known, infreq(Query,Vars,Types) for the known infrequent ones, max(Query,Vars,Types,Instances) for the known maximal queries. Depending on the settings employed, RAP builds and uses some of or all of these tables [25].

### 6 RAP Algorithm

The RAP algorithm that is presented describes generically how one maximal pattern is obtain. It is necessary to attach three input files: i) a learning set file (filestem.kb) ii) a domain knowledge file (filestem.bg) and iii) a settings file (filestem.s). In this files, for input it is necessary to introduce the database r, a language bias definition L, the key predicate used to address examples key, the threshold minfreq and the maximal pattern length maxlength obtaining the output, maximal pattern Q. The structures set are infrequent patterns Infreq and known patterns Known [27].
1. make initial pattern $Q$ (default $Q = key(Key)$)
2. while pattern $Q$ is shorter than $maxlength$ do
3.   generate set $Q_{SPECS}$ of new refinements of $Q$
4.   discretize continuous arguments in added literals
5.   if $Q_{SPECS} \neq 0$ then
6.     select one refinement $Qe$ from $Q_{SPECS}$ such that $Qe \notin Infreq$
    and $Qe \notin Known$
7.   else
8.     pattern $Q$ is maximal
9.   end if
10.  compute support of $Qe$ on database $r$
11.  if support of $Qe$ is great or equal than $minfreq$ then
12.     $Q = Qe$ and update structure $Known$ (add $Qe$ to $Known$)
13.  else
14.     add pattern $Qe$ to $Infreq$ and go to Step 5
15.  end if
16. end while

Parallel and distributed data mining have become very important in the recent time. It can overcome many problems of serial algorithms, particularly the lack of operation memory for storing data or hypothesis, or immoderate time requirements. Both problems play an important role in Inductive Logic Programming (ILP). Almost all ILP systems are designed to run in main memory, therefore it is difficult to use them for mining in large volume of data.

dRAP, a framework for distributed mining first-order frequent patterns extends the RAP system for running on parallel shared-nothing architecture. It utilizes several well-known methods for parallel mining propositional frequent patterns and a new algorithm that minimizes communication overhead. The new algorithm require significantly smaller number of messages passed than the earlier methods.
7 Prolog

RAP is written in the Prolog language. Prolog is a logic programming language. It is a simple, general purpose language often associated with artificial intelligence and computational linguistics [28]. Its base on first-order logic, also known as predicate logic and it stands away from machine logic to approach to human logic [29]. Prolog single data type is the term. Terms are either atoms, numbers, variables or compound terms [30]. Prolog programs describe relations, defined by means of clauses. Pure Prolog is restricted to Horn clauses, a Turing-complete subset of first-order predicate logic. There are two types of clauses: Facts and Rules. Clauses with empty bodies are called facts. [30].
8 The interface

The implementation of a graphical interface for text mining with the RAP system is the main objective of the dissertation and it will consume the majority of resources in the work. The Graphical User Interface (GUI) will be Java and PHP based and a data base connection is required (based on the SQL language). The system should be usable by any non ILP text mining practitioner. The interface will allow a easy and quite assembly of the background knowledge by the user and should provide a graphical exploration of data (in a text form) and ideation (advanced visualization to permit the user the formation of ideas or concepts) [5]. The interface will allow the user to inspect where in the texts the induced patterns will match.

9 Wrappers

Undertaking the application of the interface, wrapper(s) will be implemented. A wrapper refers to a piece of code that ‘adapts’ one interface for a class into one that a client expects [31]. An adapter allows classes to work together that normally could not because of incompatible interfaces by wrapping its own interface around that of an already existing class. The adapter is also responsible for handling any logic necessary to transform data into a form that is useful for the consumer. For instance, if multiple boolean values are stored as a single integer but a consumer requires a ‘true’/‘false’, the adapter would be responsible for extracting the appropriate values from the integer value [31].
10 Discussion

The research made revealed numerous articles in the data mining/machine learning area. The RAP System is revealed to be a promising tool in mining first-order maximal patterns with successful results for feature mining. It can be used as new features for propositional learns.

The Java programming language is the main tool in which the work will be done. PHP and SQL programming languages will be used for a more specific work in the project. The Eclipse software [32] presents to be an interesting platform to write, compile, test and document the code, and therefore it is chosen to support the dissertation work. The dissertation text will be presented with latex application.

The creation of an initial prototype is not already a require to the project but the advantages that can be inherit will this approach leaves to its implementation in an initial phase of the work. In a same way, the creation of a SRS (Software Requirements Specification) may be required. A SRS is an organization’s understanding (in writing) of a customer or potential client’s system requirements and dependencies usually prior to any actual design or development work.
11 Dissertation work plan

The work will consist in the development of an interface and a wrapper to make the use of the Text Mining system RAP easy. For that purpose we will design an interface with user interface techniques that facilitate the interaction between the user and the system. It will be used methods to translate documents from different formats into a format suitable for RAP. The interface will have facilities to assist the user in the development of the background knowledge required by RAP. RAP’s parameters will be automatically tuned using a wrapper. We will also provide the interface with procedures to make the presentation of the patterns found more appealing.

The generally activity plan that was traced is here described. The final project and the dissertation text should be presented at July 2008.

1. Implementation of the interface with the user
2. Implementation of the RAP wrapper
3. Deployment and test of the interface
4. Writing of the dissertation text

With approximately 16 weeks to work into the project, its essential to resume each item for no more than 4 weeks. Weekly reunions must be scheduled with the advisors in order to ensure a well performed work. With the proper strategy, not yet defined, its important to build a scalable, robust and adaptable interface. On the midterm of the project (at the ending of April) a prototype must be presented.
12 Conclusion

The state-of-art in data mining and ILP systems was presented. With a special attention to the RAP System, it is a responsibility of the dissertation work to provide a powerful, efficient, scalable interface as a complement to the system, providing the user access to the knowledge obtained. The place where the work will be done will be at the Department of Computer Science at Masaryk University, Czech Republic.

The development of a web page about the RAP system may be implemented. The idea is to present more visibility to the system, introducing it, explaining its purpose and features. The idea of an online interface, through a PHP or Java server, in complement to an offline interface. This way, the RAP system could be an important tool in mining maximal frequent patterns in web texts (web mining).
References


[7] Rapidminer (YALE) Project (http://rapid-i.com/)

[8] Sašo Džeroski, Nada Lavrač, Relational Data Mining, Chapter 14 Relational Data Mining Applications: An Overview, pgs 339-364


(http://www.cs.cmu.edu/ fp/courses/lp/lectures/lp-all.pdf)

[16] Grigoris Antoniou, Aditya Ghose, Miroslaw Truszczynski, Machine
Learning - Learning and Reasoning With Complex Representations, pg 92

[17] Inductive Logic Programming
(http://www.doc.ic.ac.uk/ shm/ilp.html)


(http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph)


[23] CProgol (http://www.doc.ic.ac.uk/ shm/Software/)


[25] RAP: Framework for Mining Frequent Datalog Patterns
(http://www.cs.bris.ac.uk/ ILPnet2/Tools/Reports/Abstracts/2002-
blavtak.html)

[26] Jan Blat’ák, RAP: System for finding maximal frequent patterns in
first-order logic

[27] Jan Blat’ák and Lubos Popelinsky, Feature construction with RAP,
pgs 3-4 http://www.fi.muni.cz/kd/papers/blatak_ILP03.pdf

[28] The Prolog Language
(http://www.sics.se/sicstus/docs/3.7.1/html/sicstus_5.html)

[29] Prolog (http://www.inf.ufg.br/ gustavomota/docs/prolog.pdf)


(http://en.wikipedia.org/wiki/Wrapper_pattern)

[32] Eclipse - an open development platform (http://www.eclipse.org/)