Mining Hierarchical Pathology Data using Inductive Logic Programming

Tim Op De Beéck[1], Arjen Hommersom[2], Jan Van Haaren[1], Maarten van der Heijden[2], Jesse Davis[1], Peter Lucas[2], Lucy Overbeek[3], and Iris Nagtegaal[4]

[1] Department of Computer Science, KU Leuven, Belgium
{tim.opdebeeck,jan.vanhaaren,jesse.davis}@cs.kuleuven.be
[2] Institute for Computing and Information Sciences, Radboud University, NL
{arjenh,m.vanderheijden,peterl}@cs.ru.nl
[3] Registry of Histo- and Cytopathology in the Netherlands, Utrecht, NL
[4] Department of Pathology, Radboud University Medical Centre, Nijmegen, NL

Abstract. Considerable amounts of data are continuously generated by pathologists in the form of pathology reports. To date, there has been relatively little work exploring how to apply machine learning and data mining techniques to these data in order to extract novel clinical relationships. From a learning perspective, these pathology data possess a number of challenging properties, in particular, the temporal and hierarchical structure that is present within the data. In this paper, we propose a methodology based on inductive logic programming to extract novel associations from pathology excerpts. We discuss the challenges posed by analyzing these data and discuss how we address them. As a case study, we apply our methodology to Dutch pathology data for discovering possible causes of two rare diseases: cholangitis and breast angiosarcomas.

1 Introduction

The nationwide network and registry of histo- and cytopathology in the Netherlands (PALGA) aims to facilitate communication and information flow within the field of pathology and to provide information to others in health care. PALGA began collecting pathology reports generated in The Netherlands in 1971 and has complete coverage of all pathology laboratories in both academic and non-academic hospitals in The Netherlands since 1991 [3]. Currently, its database contains approximately 63 million excerpts of pathology reports, which are coded using a variant of the SNOMED classification system that was originally developed by the College of American Pathologists [5]. Each year, approximately three million excerpts are added.

The pathology database provides a rich data source for answering medically relevant questions, usually in the form of testing associations between concepts in the data like diagnoses, morphology, etc. This currently leads to roughly 25 to 30 publications per year. Typical examples that use the PALGA data include studying incidence of rare diseases (e.g., Brenner tumours of the ovary), as well as mortality of diseases, co-morbidities, and cancer. However, using data mining

and machine learning techniques to find completely novel associations, instead of testing predefined hypotheses, is completely unexplored with this data. The sheer size of the data presents significant opportunities to find medically relevant associations by mining the data.

It is well recognized in the literature that medical data is one of the most challenging types of data to analyze [4]. Typical challenges associated with medical databases include their sheer volume and heterogeneity, inconsistencies, selection biases, and significant amounts of missing data. In this paper, we discuss the specific challenge of how to effectively cope with the specific structure (e.g., relationships, time dependencies, hierarchies, etc.) present in pathology data. Structure within data can be beneficial as it may be exploited during learning, however, standard data analysis techniques typically assume the data are flat.

In this paper, we focus on discovering novel associations within pathology data using inductive logic programming (ILP) techniques. Using an inductive logic programming approach provides several benefits compared to both propositional machine learning approaches (e.g., decision trees, rules sets, etc.) and traditional pattern mining approaches (e.g., association rule mining, sequence mining, etc.). Propositional machine learning approaches require that each example is defined by a fixed-length feature vector. For the PALGA data, it is non-trivial to define such a feature set because different patients can have different numbers of entries in the database. The relational nature of ILP allows us to avoid this problem by simply creating one fact for each entry in the database. Furthermore, it is well known that by introducing the appropriate background knowledge, ILP can more naturally capture the hierarchical organization of the codes [14, 16] compared to both propositional learners and pattern mining.

2 Description of the Data

In this section, we present two case studies and the structure of the data.

2.1 Case Studies

The goal of this work is to investigate whether it is possible to automatically extract useful relationships between pathologies. In particular, we are interested in finding possible causes of certain pathologies. In collaboration with pathologists, we selected two case studies to evaluate the techniques proposed in the remainder of this paper. The first case study deals with *cholangitis*, which is an inflammation of the bile ducts. The second case study aims to learn associations with *breast angiosarcomas*, which is a tumour in the walls of blood vessels. For both cases studies, we obtained data from patients who were diagnosed with these diseases. As controls, we obtained data from patients with colitis ulcerosa and Crohn's disease for cholangitis, and patients with angiosarcomas not in the breast and neavus mamma for breast angiosarcomas. We also acquired data representing a general population to avoid a selection bias.

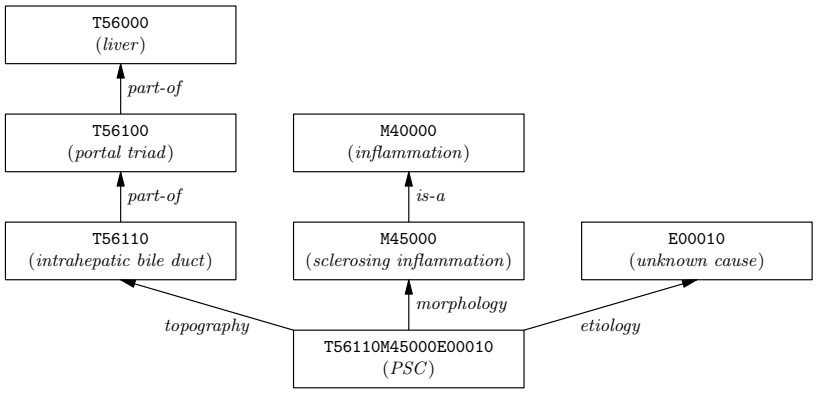

Fig. 1. Hierarchy of the primary sclerosing intrahepatic cholangitis (PSC) code.

2.2 Hierarchical Coding

The data consist of diagnoses from excerpts of pathological reports that are coded based on the SNOMED classification system. Each diagnosis consists of multiple codes including at least the topography (i.e., location), the procedure for obtaining the material, and a finding (i.e., pathological morphology or diagnosis). For example, a particular instance of colitis ulcerosa may be coded by the following code: `colon,biopsy,colitis ulcerosa`.

The coding poses significant challenges. First, the number of codes per diagnosis varies. For example, there may be multiple topographies such as colon and duodenum as well as multiple morphologies within the same diagnostic rule. Combinations of codes are also relevant. For example, skin (`T01000`) and breast (`TY2100`) should be interpreted as *skin of the breast*. Also, combinations of topographies and morphologies are important. For example, angiosarcoma of the breast may be coded by `TY2100`, `T01000` (skin of breast) and `M91203` (angiosarcoma). Moreover, a code itself can contain a hierarchical structure as is illustrated in Figure 1. Exploiting the structure within a code is a key challenge.

3 Background on Inductive Logic Programming

In this section, we give some background on sequential data mining using ILP.

3.1 First-Order Logic and Logic Programming

First-order logic (FOL) is a formalism to represent objects and their relations in a structured format. Due to its expressiveness, FOL is widely used in machine learning applications. This project only requires a subset of FOL, limiting the alphabet to three types of symbols. *Constants* (e.g., a diagnosis $\mathtt{d_i}$), referring to specific objects in the domain, start with a lower-case letter. *Variables* (e.g., `Patient`), ranging over objects in the domain, are denoted by upper-case letters.

Predicates $\mathtt{p}\backslash n$, where n is the arity (i.e., number of arguments) of the predicate, represent relations between objects.

Given these symbols, several constructs can be defined: *atoms* $\mathtt{p}(\mathtt{t}_1, ..., \mathtt{t}_n)$, where each $\mathtt{t}_i$ is either a constant or a variable; *literals*, i.e., an atom or its negation; *clauses*, a disjunction over a finite set of literals; and *definite clauses*, i.e., clauses that contain exactly one positive literal. Definite clauses can be written as an implication $\mathtt{B} \Rightarrow \mathtt{H}$. The body $\mathtt{B}$ consists of a conjunction of literals, whereas the head $\mathtt{H}$ is a single literal. Variables in definite clauses are presumed to be universally quantified. For example, the rule $\mathtt{diagnosed}(\mathtt{Patient}, \mathtt{carcinoma}) \Rightarrow \mathtt{diagnosed}(\mathtt{Patient}, \mathtt{angiosarcoma})$ states that if a `Patient` has been diagnosed with a `carcinoma`, he or she has also been diagnosed with an `angiosarcoma`.

In the following, clauses are given a logic programming semantics. In brief, a logic programming engine first replaces each variable by an object of the variable's domain. Then, the engine checks whether the resulting instantiation of the head of the rule is true if the instantiation of the body is true. In this case the rule is said to *cover* an example (i.e., a set of variable instantiations). The *coverage* of the rule is the number of examples covered.

3.2 Inductive Logic Programming

Inductive logic programming (ILP, [9]) aims to learn hypotheses for a given concept. More formally, we define ILP as follows:

Given: A concept `C`, a background knowledge `K`, a language specification `L`, an optional set of constraints `I`, a non-empty set of *positive* examples `E+` of `C` (i.e., examples of `C`), and a set of *negative* examples `E−` of `C` (i.e., examples of not `C`).

Learn: A set of clauses `S`, in the form of a logic program, that respects the constraints `I` and covers all of the positive examples in `E+` and none of the negative examples in `E−`.

ILP is well-suited to be used in this medical setting as it yields interpretable rules and allows a user to iteratively narrow down the search space by defining background knowledge `K` and constraints `I` on acceptable hypotheses.

3.3 Aleph

We use the Aleph ILP system to learn the hypotheses [12]. The system iteratively learns first-order definite clauses from positive and negative examples.

To learn a clause, Aleph proceeds as follows. First, it randomly picks a positive example, which is called the seed example, and searches the background knowledge for facts known to be true about the seed (saturation step). The system combines these facts to construct the most-specific clause that covers the seed (i.e., the bottom clause). Second, Aleph generalizes these facts as the generalizations of facts explaining the seed might also explain other examples (search step). Typically, Aleph employs a breadth-first search that considers the

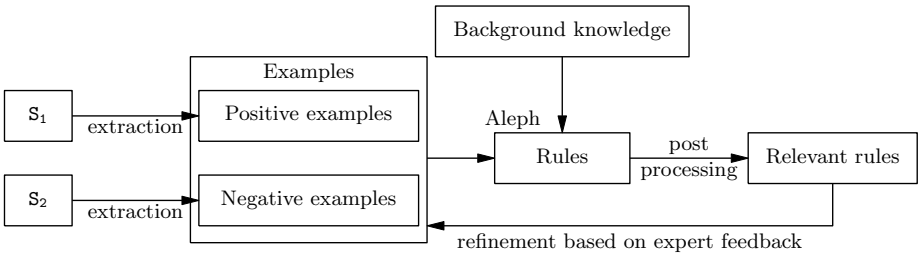

Fig. 2. Overview of our method for extracting knowledge from PALGA data using ILP.

shortest clauses first. From these clauses, the system works its way through the search space in a general-to-specific manner towards the bottom clause. The search process can be further constrained by bounding the clause length and the number of clauses that should be tested.

Several metrics exist to evaluate each generated clause. Usually, these metrics try to capture how well the clause discriminates between positive and negative examples. A commonly used metric for this purpose is the m-estimate [8], which is a smoothed ratio between the number of positive examples covered and the total number of examples covered (i.e., the precision of the clause).

Aleph's global search strategy affects how it proceeds from one iteration to another. One commonly used strategy is the "cover-removal" approach which removes all positive examples that are covered by a previously learned clause from the set of positive examples. Hence, successive iterations focus on learning rules that apply to currently uncovered examples (i.e., none of the learned rules covers them). Another approach is to use every positive example as a seed once such that more rules are learned.

4 Empirical Evaluation

4.1 Methodology

More formally, we address the following learning task in this paper:

Given: A disease of interest $\mathtt{d_t}$, SNOMED-structured patient records, and hierarchical domain knowledge.

Learn: Novel associations in the data that provide clinical experts with new insights about disease $\mathtt{d_t}$.

In the following, we discuss the steps taken to address this task. Figure 2 provides a schematic overview.

Sampling the medical database. Our approach requires two samples of the database. We extract positive examples from a sample $\mathtt{S_p}$, which are patients who suffer from the target disease $\mathtt{d_t}$, and negative examples from a sample $\mathtt{S_n}$, which are patients who have **not** been diagnosed with $\mathtt{d_t}$. While $\mathtt{S_p}$ mainly

Table 1. The records of a patient X.

Patient ID	Date	Diagnosis
X	06/06/2011	T67000—P11400—T67000M40030
X	22/06/2011	T56000—P11400—P30700—T56110M45000E00010—M55800

consists of patients who have been diagnosed with $\mathtt{d_t}$ or a strongly related disease, $\mathtt{S_n}$ represents the general population. This approach avoids a selection bias that could make it harder to discriminate between positive and negative examples.

Extracting examples. An example corresponds to a patient $\mathtt{p_i}$'s records. If a patient $\mathtt{p_i}$ was diagnosed with $\mathtt{d_t}$ at a time $\mathtt{t_i}$, we label the example as *positive* and add a ground fact $\mathtt{d_t}(\mathtt{p_i}, \mathtt{t_i})$ to the set of positive examples. We add all diagnoses $\mathtt{d_i}$ of $\mathtt{p_i}$ at time $\mathtt{t_j} < \mathtt{t_i}$ (i.e., before the first diagnosis of the target disease) as background knowledge as ground facts $\mathtt{diagnosis}(\mathtt{p_i}, \mathtt{normD_i}, \mathtt{t_j})$.

While each diagnosis $\mathtt{d_i}$ is a highly-structured sequence of codes, we need to normalize the sequence to $\mathtt{normD_i}$. Our normalization procedure sorts the codes alphabetically and ensures they were entered consistently. While not essential, this procedure allows to more easily detect duplicate diagnoses, which avoids learning hypotheses consisting of multiple atoms describing the same diagnosis.

We construct the *negative* examples in a similar way. We search $\mathtt{S_n}$ for patients $\mathtt{p_j}$ who have never been diagnosed with the target disease $\mathtt{d_t}$ and add a ground fact for each such patient to the set of negative examples.

We now illustrate this process for the records of a patient X, which are shown in Table 1, to learn rules about cholangitis. We search X's records chronologically for codes corresponding to cholangitis. The second record, which dates from 22 June 2011, mentions the code T56110M45000E00010 referring to primary sclerosing cholangitis. Hence, we label X as a positive example and add the following fact to the set of positive examples: $\mathtt{cholangitis}(\mathtt{x}, 22/06/2011)$. In addition, we add all X's records that were recorded before 22 June 2011 as background knowledge. For this example, we generate the following three ground facts: $\mathtt{diagnosis}(\mathtt{x}, \mathtt{M40030T67000}, 06/06/2011)$, $\mathtt{procedure}(\mathtt{x}, \mathtt{P11400}, 06/06/2011)$, and $\mathtt{topography}(\mathtt{x}, \mathtt{T67000}, 06/06/2011)$.

Adding background knowledge. To exploit the structure of the data during learning, we add a hierarchy of codes as background knowledge, which we provide as a set of clauses (see Figure 1). For example, the clause $\mathtt{diagnosis}(\mathtt{P}, \mathtt{normD_j}, \mathtt{T}) \Rightarrow \mathtt{diagnosis}(\mathtt{P}, \mathtt{normD_i}, \mathtt{T})$ specifies that diagnosis $\mathtt{d_i}$ is more general than diagnosis $\mathtt{d_j}$. This approach allows us to learn more generally applicable rules.

Configuring and running Aleph. Aleph has many parameters that influence the number of learned clauses. If configured too strictly, it learns no clauses at all. If configured too loosely, it learns many uninteresting clauses which makes manually inspecting the learned clauses a slow and tedious process.

Table 2. The top-five rules for cholangitis with their coverage of positive and negative examples, and m-estimate. The abbreviated notation is explained in the text.

Rule	$\|\mathtt{E}+\|$	$\|\mathtt{E}-\|$	**m-est.**
brush ∧ colon ∧ no tumour ⇒ cholangitis	55	0	0.90
extra hepatic bile duct ∧ liver ∧ no abnormalities ⇒ cholangitis	51	0	0.90
brush ∧ liver ∧ no tumour ⇒ cholangitis	50	0	0.89
ductus choledochus ∧ colon ∧ no tumour ⇒ cholangitis	50	0	0.89
ductus choledochus ∧ colon ∧ no abnormalities ⇒ cholangitis	45	0	0.89

The following parameters highly influence the number of generated clauses: `minpos` denotes the minimum number of positive examples that a clause *must* cover, `noise` denotes the maximum number of negative examples that a clause *can* cover, and `minacc` denotes the minimum precision (i.e., the percentage of covered examples that should be positive).

As a global search strategy, the `induce_max` setting is a good choice when performing knowledge discovery as it picks each positive example as the seed once. Hence, it generates more clauses and ensures that the order in which the seeds are picked does not influence the clauses that are learned. The `explore` parameter forces Aleph to continue the search until all remaining elements in the search space are definitely worse than the current best element [12].

Scoring the learned clauses. We sort the clauses according to their coverage of positive examples. In case of a tie, we sort the clauses according to their m-estimate. This approach allows us to easily discover and inspect the top clauses.

4.2 Experimental Results

We applied the above methods to both case studies. For cholangitis, we constructed 1,292 positive examples from $\mathtt{S_{p1}}$ containing 402,939 records of 78,911 patients. For angiosarcoma of the breast, we constructed 303 positive examples from $\mathtt{S_{p2}}$ containing 28,557 records of 14,424 patients. We constructed 7,958 negative examples for cholangitis and 7,963 negative examples for angiosarcoma of the breast from $\mathtt{S_n}$ containing 53,439 records of 7,963 patients.

Running Aleph with a `minpos` of `10`, a `minacc` of `0.5`, and `noise` values of `10`, `50`, and `100` resulted in a total of 6,775 rules for cholangitis, and 945 rules for angiosarcoma of the breast. Among the best-scoring rules, there are several examples where the background knowledge capturing the hierarchical structure was used to construct rules. For example, a high-scoring rule for cholangitis is:

$$\mathtt{diagnosis}(P, \mathtt{auto\text{-}immune\ disease}, T_1) \ \wedge \ \mathtt{topography}(P, \mathtt{liver}, T_2) \wedge \mathtt{morphology}(P, \mathtt{fibrosis}, T_3) \Rightarrow \mathtt{cholangitis}(P, T)$$

where $T_1, T_2, T_3 < T$. While "auto-immune disease" does not explicitly appear in the data, Aleph used the background knowledge to derive it as a generalization of the more specific code "auto-immune hepatitis".

Table 3. The top-five rules for cholangitis after feedback from the pathologist with their coverage of positive and negative examples, and m-estimate.

Rule	$\|E+\|$	$\|E-\|$	**m-est.**
liver ∧ colitis ulcerosa ⇒ cholangitis	91	6	0.87
cholestasis ∧ colon ⇒ cholangitis	30	0	0.87
cirrhosis ∧ external revision ⇒ cholangitis	25	0	0.86
auto-immune hepatitis ⇒ cholangitis	49	3	0.85
cirrhosis ∧ fibrosis ⇒ cholangitis	31	1	0.85

We presented the 50 top-scoring rules containing at least one morphology or diagnosis to a gastro-intestinal specialist of the Radboud University Medical Centre in The Netherlands. Table 2 presents the top-five rules for cholangitis using a shorthand notation. For example, the first rule corresponds to:

$$\texttt{procedure}(P, \texttt{brush}, T_1) \ \wedge \ \texttt{topography}(P, \texttt{colon}, T_2) \ \wedge$$
$$\texttt{diagnosis}(P, \texttt{no tumour}, T_3) \Rightarrow \texttt{cholangitis}(P, T)$$

where $T_1, T_2, T_3 < T$.

The specialist confirmed that the high-scoring rules are in accordance with existing medical knowledge. In particular, as expected, rules for cholangitis are related to inflammatory bowel diseases and rules for angiosarcoma of the breast are related to breast cancer. Yet, for finding novel disease associations, we identified three limitations. First, there is a large number of rules that can be found within the data, which makes medical validation challenging. Second, some of the high-scoring rules are irrelevant because they are, for example, an artifact of *diagnosis by exclusion* (e.g., no tumour, no abnormalities). For example, the method of obtaining the sample (e.g., brush) is predictive but medically irrelevant. Third, for exploratory data analysis, it is only of interest which other morphologies occur before the diagnosis of interest. Repetition of the same morphology or diagnosis leads to a large number of rules, which may only differ in, for example, location.

The interpretability of ILP-learned rules facilitates the interaction with and feedback from domain experts. Inspired by the above limitations, we added a post-processing step by removing rules containing particular codes (e.g., no tumour) and rules for which there are higher-scoring rules that contain the same morphology or diagnosis. For the cholangitis case study, this results in a list of only 30 rules that cover at least 10 positive examples. Table 3 presents a few of the best-scoring rules using the same shorthand notation as in Table 2. Some of the associations, such as the strength of the association between Crohn's disease and cholangitis (with 16 positive and no negative examples), were considered surprising and warrant further investigation.

4.3 Comparison with sequential pattern mining

Sequential pattern mining is an alternative to the proposed ILP approach. We applied several sequential pattern mining variants to the cholangitis case study

(i.e., CMRules, RuleGrowth, ERMiner, TNS, and TopSeqRules), using the data mining framework SPMF [6]. To do so, we used sample $\mathtt{S}_{\mathtt{p1}}$ to construct sequences of diagnoses of patients who have been diagnosed with cholangitis. Each patient corresponds to one sequence, while each record corresponds to an itemset in the sequence. We only consider the records up until the first diagnosis of cholangitis.

Running the rule mining algorithms yields 389 distinct rules. However, the post-processing procedure from the previous section retains none of these rules. Inspecting the rules before post-processing, allows us to identify three limitations of propositional mining algorithms: (*i*) no abstractions are found using these algorithms, which makes the rules less interpretable, (*ii*) many of the rules found are irrelevant for a specific disease of interest (e.g., the rule `liver => biopt`), and (*iii*) no specific rules for cholangitis are discovered.

5 Conclusions

In recent years, there has been considerable interest in extracting knowledge from the data collected in electronic health records (EHRs), though it is recognized that there are considerable challenges involved [7, 10]. Nonetheless, several attempts have been made to mine the EHR, typically used for predictive modelling, e.g., in order to support clinical decisions or for the identification of risk factors [1, 2, 13, 15]. However, exploratory data mining to find novel relationships between diseases is something which has not been studied as far as we are aware.

This paper presented a case study of applying inductive logic programming (ILP) to extract interesting rules from a pathology data set. This paper posited that ILP is particularly suited to this task for three reasons. First, is its ability to handle structure such as the hierarchical organization of diagnosis codes and time. This not only allows improving the learning process, as in [11], but also to abstract from specific codes to more abstract ones. Second, ILP returns interpretable rules which facilitate an iterative mining process with domain experts. Third, ILP is a discriminative approach so it can focus its discovery efforts on target variables of interest. In the case studies, we found that ILP was able to exploit the structure in the data and that it produced more meaningful patterns than sequential pattern mining. Furthermore, we illustrated how we were able to revise the mining process based on the feedback of a domain expert. In the future, we will continue to explore adding additional domain knowledge to further guide the search process. Finally, rules considered interesting by the medical experts will be verified using traditional statistical methods to make them acceptable to the medical literature.

Acknowledgments

AH and MVDH are supported by the ITEA2 MoSHCA project (ITEA2-ip11027). JVH is supported by the Agency for Innovation by Science and Technology in Flanders (IWT). JD is partially supported by the Research Fund KU Leuven (CREA/11/015), EU FP7 MC-CIG (#294068) and FWO (G.0356.12).

References

1. Bellazzi, R., Zupan, B.: Predictive data mining in clinical medicine: current issues and guidelines. International journal of medical informatics 77(2), 81–97 (2008)
2. Bennett, C., Doub, T.: Data mining and electronic health records: Selecting optimal clinical treatments in practice. In: Proc. of DMIN'2010. pp. 313–318 (2010)
3. Casparie, M., Tiebosch, A., Burger, G., Blauwgeers, H., Van de Pol, A., van Krieken, J., Meijer, G.: Pathology databanking and biobanking in the netherlands, a central role for PALGA, the nationwide histopathology and cytopathology data network and archive. Analytical Cellular Pathology 29(1), 19–24 (2007)
4. Cios, K., Moore, W.: Uniqueness of medical data mining. Artificial intelligence in medicine 26(1), 1–24 (2002)
5. Cote, R., Robboy, S.: Progress in medical information management: Systematized nomenclature of medicine (SNOMED). Jama 243(8), 756–762 (1980)
6. Fournier-Viger, P.: Spmf: A sequential pattern mining framework. http://www.philippe-fournier-viger.com/spmf (2011)
7. Jensen, P., Jensen, L., Brunak, S.: Mining electronic health records: towards better research applications and clinical care. Nature Reviews Genetics 13(6), 395–405 (2012)
8. Lavrač, N., Dzeroski, S., Bratko, I.: Handling imperfect data in inductive logic programming. Advances in Inductive Logic Programming 32, 48–64 (1996)
9. Muggleton, S., De Raedt, L.: Inductive logic programming: Theory and methods. The Journal of Logic Programming 19, 629–679 (1994)
10. Ramakrishnan, N., Hanauer, D., Keller, B.: Mining electronic health records. Computer 43(10), 77–81 (2010)
11. Singh, A., Nadkarni, G., Guttag, J., Bottinger, E.: Leveraging hierarchy in medical codes for predictive modeling. In: Proc. of ACM Conference on Bioinformatics, Computational Biology, and Health Informatics. pp. 96–103. ACM (2014)
12. Srinivasan, A.: The Aleph manual. Machine Learning at the Computing Laboratory, Oxford University (2001)
13. Sun, J., Hu, J., Luo, D., Markatou, M., Wang, F., Edabollahi, S., Steinhubl, S., Daar, Z., Stewart, W.: Combining knowledge and data driven insights for identifying risk factors using electronic health records. In: Proc. of AMIA Annual Symposium. vol. 2012, p. 901. American Medical Informatics Association (2012)
14. Vavpetič, A., Lavrač, N.: Semantic subgroup discovery systems and workflows in the sdm-toolkit. The Computer Journal 56(3), 304–320 (2013)
15. Wang, F., Lee, N., Hu, J., Sun, J., Ebadollahi, S.: Towards heterogeneous temporal clinical event pattern discovery: a convolutional approach. In: Proc. of the 18th ACM SIGKDD. pp. 453–461. ACM (2012)
16. Žáková, M., Železnỳ, F.: Exploiting term, predicate, and feature taxonomies in propositionalization and propositional rule learning. In: Proc. of ECML 2007. pp. 798–805. Springer (2007)