DICE: a Discovery Environment integrating Inductive Bias

Jean-Daniel ZUCKER Vincent CORRUBLE Jérôme THOMAS Geber RAMALHO

LAFORIA-IBP, Université Paris VI
4, Place Jussieu - Boîte 169
75252 PARIS Cedex 05
FRANCE
email: {zucker,corruble,thomas,ramalho}@laforia.ibp.fr
Telephone: 33-1-4427-7009 Fax: 33-1-4427-7000

Abstract: Most of Knowledge Discovery in Database (KDD) systems are integrating efficient Machine Learning techniques. In fact issues in Machine Learning and KDD are very close allowing for a natural straightforward integration. However, there are specific problems related to KDD that require a specific approach to machine learning techniques integration. Overabundance of patterns and complexity of the discoveries is a central problem that we attempt to tackle. Our approach is to select several learning biases that are particularly relevant to KDD and to integrate them in a Discovery process. These learning bias are integrated into a KDD system, called DICE, that uses two Machine Learning Algorithm CHARADE and ENIGME. DICE offers an interface that allows the user to experiment with the learning bias. DICE is currently experimented on a medical database and a Chinese characters database.

Keywords: Discovery Environment, Inductive Learning, Inductive Bias, Medicine.

Introduction

In Knowledge Discovery in Database (KDD), the data representation associated with a given database is considered as the initial representation to be used by the discovery system (Piatetsky-Shapiro, 91). For most KDD systems, inductive learning plays a central role in the process of discovering knowledge from the initial representation. In that perspective, discovering can be viewed as the process of learning yet unknown concept definitions. Nevertheless, traditional machine learning cannot simply be used as a magic tool for knowledge discovery. Indeed, if one applies an inductive algorithm to discover all the correlation between concepts in a real database, one will generally observe the production of a set of results whose size is just too large to be handled in a useful manner. A major problem is therefore to provide the system with means to reduce the overabundance of patterns or results, and to guide the learning process toward the most interesting areas. In the machine learning community, such means are termed as *learning bias* (Mitchell 80, Utgoff 86, Ganascia 88, Russell et al 90).

(Russell et al 90) presents a typology of the different kinds of bias a ML system can use. The first type is the "instance language bias" (i.e., the language which is used to describe the learning examples), the second is the "concept language bias" (i.e., the language which is used to describe the learned definitions and thus defines the search space), the third is the "restriction type bias" (i.e., all the constraints that restrict the program to a delimited search space). At last, one can use a "preference type bias" which allows the system to choose between several possible definitions.

The work presented here is an environment that integrates an inductive algorithm and an interactive incremental characterization of inductive bias relevant to discovery. This environment, called DICE, offers an interface that allows the user to experiment with the first three types of learning bias above mentioned. DICE is not an autonomous discovery system, its purpose is to enable the user to define a first rough learning bias and then to refine it step by step according to the results of the previous learning phase. In order to enable such a refinement cycle, we need to have a learning bias which is interpretable both by the user and the system. Therefore, we have drawn a parallel between the first three kinds of bias and three different types of knowledge about the domain and the task to be solved. Thus, DICE will use this knowledge to define the learning examples, define the concept language and reduce the search space.

In the first section, we present why the concept of inductive bias can contribute to a KDD system, and we show the need for an experimental environment that would aid the incremental construction of such inductive bias. In section two, we present the DICE environment. This section is divided into three parts, each corresponding to a different kind of bias and thus to a different type of knowledge. In section three, we briefly present two projects for which we are currently using DICE, namely an analysis of Chinese characters and a study on a complex medical database.

1. An approach to inductive bias for KDD

Three types of bias have been identified, each corresponding to a well-defined means to control the induction. A first kind of bias is concerned with the *reformulation* of the initial database into an appropriate example base that focuses on a potentially pertinent data structure. A second kind of bias is concerned with the *enrichment* of the example base using background knowledge. Finally the third bias is concerned with the learning bias *restricting* the exploration of the learning base.

1.1. Reformulation of the Database Representation

Reformulating to modify the representation

To find a good representation adapted to the discovery process is a key aspect in KDD. In the field of ML, *representation synthesis* is precisely concerned with *modifying* the existing representation to improve the quality of learning (Subramanian, 89). Improving the quality of learning is either to produce better generalizations (Jianping and Michalski, 89) or to find a new representation to solve problems within some computational constraints. One well-known approach to improve learning is called constructive induction (Matheus, 91a), it aims at lifting some of the limitations of the initial description language of the instances (Matheus, 91b; Lapointe, Ling & Matwin, 93). We shall not study this kind of reformulation w.r.t. the possible learning bias since constructive induction does not restrict the overabundance of patterns but is rather concerned with the quality of the description itself. On the other hand, the work on abstraction is concerned with abstracting the languages used for describing instances to reduce the complexity of initial representations (Drastal and Czako, 89). The approach consists in generating various *abstractions* that progressively introduce details of the initial representation (Riddle, 90). Subramanian, for example, proposes an *irrelevance principle* to minimize a formulation by *"removing all facts that are either logically or computationally irrelevant to the specified goals"* (Subramanian, 90). However, in attempting to *discover* as opposed to learn, facts cannot be removed on the basis of such an irrelevance principle because the concepts to be discovered might precisely involve knowledge initially considered as *irrelevant*..

Our approach to find a good representation is to provide the user with an expressive language, namely a subset of Horn Clauses, to represent its initial data and using an inductive bias to reformulate the initial representation in attribute-value representations. Regarding expressivity, it is obvious that such language is more expressive than classical attribute-value representation. Nevertheless, representation languages subset of First Order Logic lead often to costly algorithms. To be precise, if no learning bias is introduced, the complexity of algorithms supporting representation allowing for more than one matching is exponential with the number of examples (Ganascia, 87). The complexity of such algorithms is partly due to the matching problem. Indeed, for a given predicate repeated three times in each description of N examples, there exists 3^N possible matches of the given predicate. To address this problem and obtain tractable algorithms, without loosing the benefits of Horn Clauses representation power, we propose to explore the different matches using a strong inductive bias. We exploit the fact that re-representing the domain examples can lead to performance gains (Lavrac, Dzeroski & Grobelnik, 91) but what mostly interests us is the entity of the domain that will be described to form the learning examples. We have introduced in (Zucker & Ganascia, 94) the neologism *morion*, (from μοριον meaning "part" in Greek) to designate a part or entity of the Database that are potentially interesting to match together. The morion is defined as: any entity, be it composed of, or part of, domain examples. It establishes a link between the Database records and the machine learning examples by qualifying the part of the domain that will be the object of description. The *morion* captures dynamically the nature and evolution of the structure of the machine learning example.

A bias to restrict the potentially high number of matches

The first bias considered in our approach to the discovery process modeling consists in defining morions, i.e. the entities that will be used to reformulate the database so as to concentrate the knowledge discovery process on potentially interesting entities, namelly composed of morions. In (Piatetsky-Shapiro,91), two kinds of representation for discovered concept are identified : Interfield patterns (if d1 then d2) and interrecords pattern (for records for which property P1 an P2 are true then P3). We claim that inter-subrecords patterns may be added and that a way to obtain such description is to perform reformulation of the initial database into a learning base on the basis of the structure called morion.

Let us take a concrete example of a medical database describing patients where amongst others four different attributes are used: an initial diagnosis called InitDiag, a transient diagnosis called TransDiag, a final diagnosis called FinalDiag and the patient age. In classical inductive systems using an attribute-value based representation, the values of a given attribute ATTi are subject to be generalized but only with other values of the ATTi. But the nature of the three diagnosis are similar. The value of a DI-diagnosis might be interested to match with the DF-diagnosis of another patient.

Our approach to this problem consists in *structuring* the initial set of attributes so as to identify entities to be matched. After the definition of such an entity it becomes possible to reformulate the initial *database* to produce an *example base*. In this new representation, the examples to be matched are the ones that have been identified to potentially interesting to match to one another. In the medical case, the implicit structuration of the initial database is a flat structure where the entity Patient is represented with its four attributes (See Figure 2).

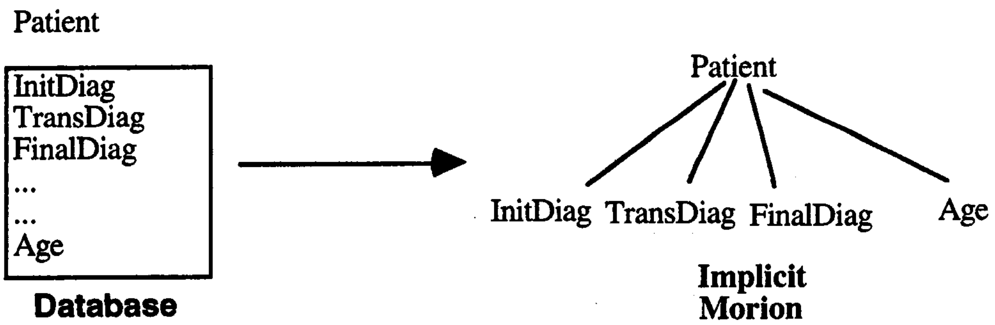

Figure 2: The implicit morion *Patient* associated to the records of Patients

In such representation, Patients will be matched to one another and relations between the different attributes will not be detected. By changing the implicit morion and acquiring a more complex morion (See Figure 3) it becomes possible to define the entity Diagnosis as the entity to be matched. After reformulating using this entity, the example base is a Diagnosis example base. The entities to be matched are therefore the diagnoses.

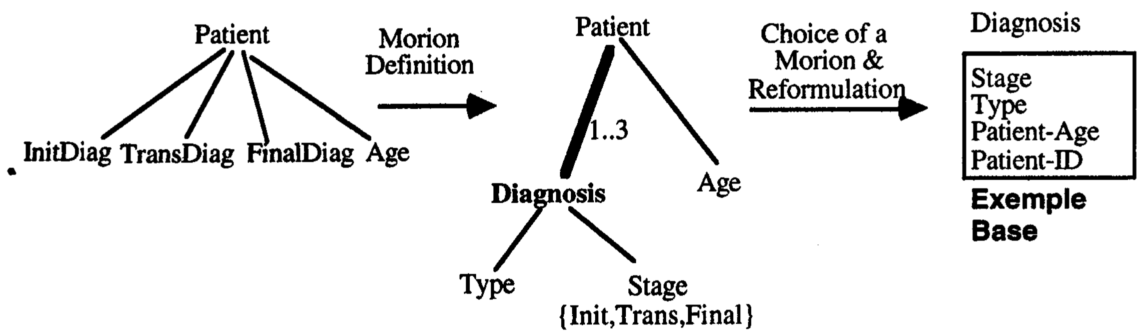

Figure 3: Definition of the morion Diagnosis and associated Reformulation

The moriological bias is therefore a way to define the entities that will be matched during the discovery process: the morion. Once the morion defined, it becomes possible to reformulate the example base into a new example base where the entities described are the ones to be compared.

1.2. Deduction

A database memorizes some key features of the natural examples. These features are however not always sufficient to perform useful inductions. Some background knowledge can be necessary to either fill the gaps left by the missing values in the current database description, or to introduce the values of attributes not previously present in the database. This process can be performed using a set of axioms representing an important part of the background knowledge. Given an example base, the set of axioms can be used to *deduce* additional information about the examples before the induction is performed.

Taking into account this kind of background knowledge can be crucial to simulate the human discovery process. As we have been able to illustrate in a study of the discovery of the causes of scurvy (Corruble V. & Ganascia J.G., 1993), human induction cannot be simulated if one takes only the facts observed by a scientist into account. The background theoretical framework is also of primary importance and needs to be represented. This framework is not explicit and may require some particular attention to be formalized. This work should not be neglected because its absence can originate a gap between the inductive behaviors of the KDD system and of the human user.

In the field of Knowledge Discovery in Medical Databases, the need for background knowledge appears obviously. A database of patient records will only gather the salient features of the cases. The experience of the physician acquired by years of clinical studies will lead him/her to interpret these salient features at another level of abstraction, using more abstract concepts and medical theories. The passage from the salient features (symptoms, treatment,...) to the comprehensive description is often made implicitly and needs to be formalized as much as possible if one aims at producing induction of interest for the physician. Most of the elements of this passage can be represented using an axiomatic formalism and deduction.

Since we are interested in discovering new knowledge, the area of knowledge that we are exploring is bound to be imprecise. Therefore, the background knowledge represented in axioms form must be given the status of hypotheses rather than fixed knowledge. Also, it is potentially very fruitful to experiment on the content of this formalized background knowledge, knowing that this set of hypotheses has direct influence on the knowledge induced. What appears here is the need for a discovery environment which permits an easy visualization and modification of this axiomatic background knowledge, and a efficient and fast mean to see its effect on induction. That is one interesting aspect of the DICE system.

1.3. *Induction*

The previous parts have described the way DICE allows the user to define both the instance and concept languages. However, experiences in Machine Learning (Mitchell 80, Ganascia 88, Morik 91, and so on) have shown that all the learning systems need further constraints. Furthermore, if the learning base does not sample well-enough the possible cases, it is possible to find correlations which appear to be true only by chance. The first way to eliminate such correlations is to introduce counter-examples. However, in the KDD framework such counter-examples may not be available and, anyway, this process can be very tedious. Another way is to introduce further constraints on the induction phase. Such constraints are able to eliminate whole classes of correlations. In DICE, the idea is to enable the user to describe, through the definition of the Problem Solving Method (see section 2.4), the kind of correlations the tool has to look for and hence to eliminate all the other ones. The user can first acquire a very rough PSM, providing a minimal guidance, and then, thanks the analysis of the rules produced, refine little-by-little this PSM or the other bias described in section 2.1 and 2.2.

2. The DICE System

2.1. *An overview of the system*

The DICE system has been designed to implement the approach presented in section 1. From a given Database, DICE allows the user for the introduction of three different types of bias that contribute to precise the direction where discoveries ought to be looked for. On a practical point of view, the user is given the possibility of realizing a number of operations within the environment. Each of these actions is partly assisted and partly automated and operates on the learning base and/or the learning bias. One major interest of the DICE environment is that it leaves open to the user the choice of any kind of sequence of operations, and that the choice of this sequence can be revised at any time during the experimentation, according to the results previously obtained.

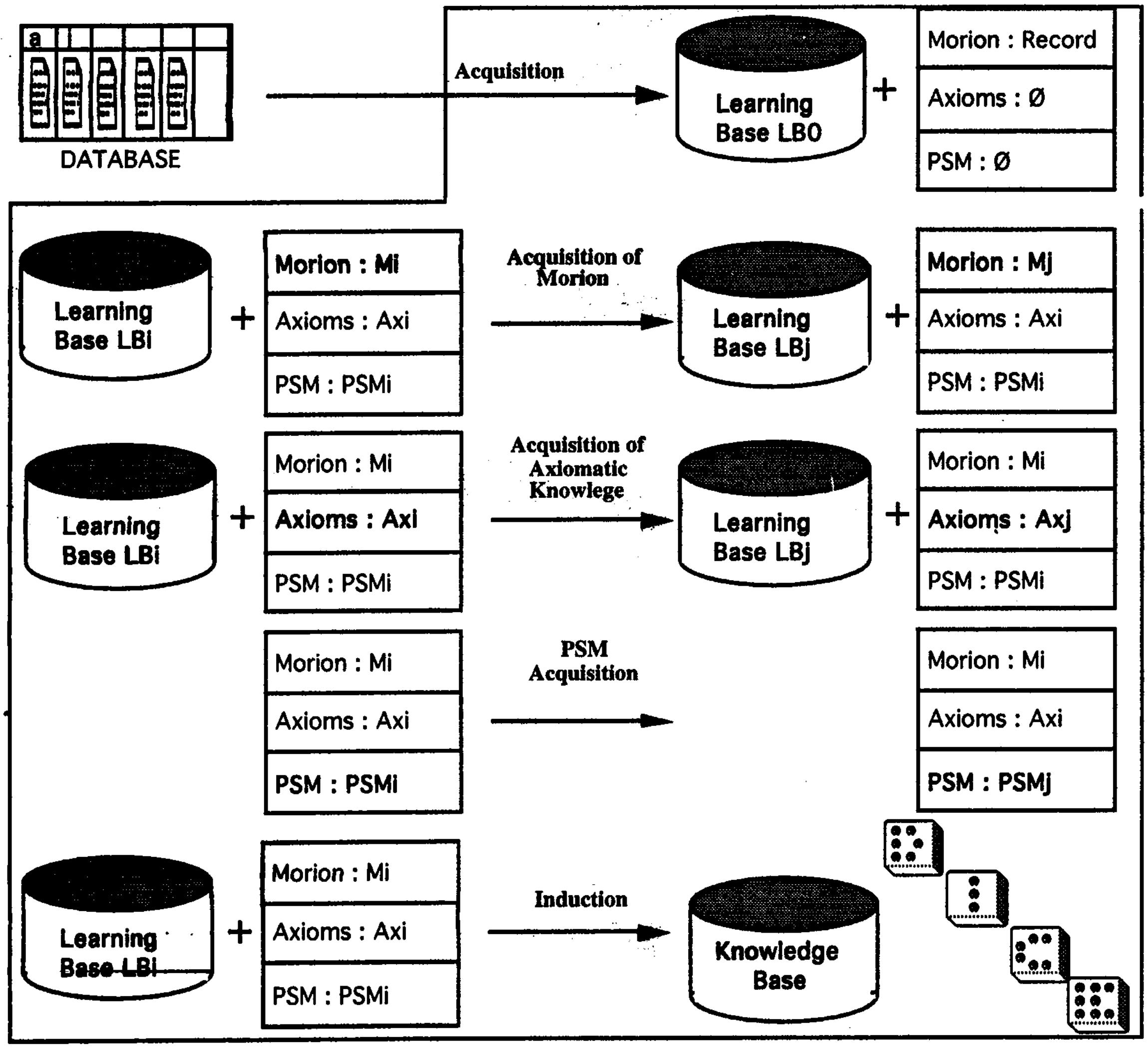

Figure 4: The five Interactive Processes in DICE

2.2. *Database and morion acquisition*

In DICE, the first stage is concerned with the representation of the attributes of the inputted DB so as to import the DB in the system. *Field names* become *attribute names* and *field types* become *attribute type* (Piatetsky-Shapiro,91). Another stage that can be performed at any time consists in defining a morion, i.e. to restrain the example base to a base containing only entities that are interesting to compare. To do so, DICE offers an interface to define new entities as sub-entities of the implicit one provided initial database structure. A simple language allows to describe the new entities with attributes of the initial morion (See figure

3). Once a complex morion defined, it becomes possible to select one entity and to perform an automatic reformulation that will use the chosen morion to produce a new example base.

2.3. Using background knowledge

The axiomatic knowledge is a set of axioms that is used to add some information to the example base before the inductive learning is actually performed. Given this set of axioms and the example base, an inference engine is used (CLIPS from NASA in ENIGME) to enrich the descriptions of the examples. In the DICE environment, a first step is to acquire the set of axioms. The language used to express these axioms corresponds to the CLIPS language. Using DICE, the acquisition is made quite straightforward because it is possible to pick directly the attributes among the list already defined. The possible values are also defined so that a minimum of work is needed and the possibility of mistakes is minimized. Once an axiom is defined, it is possible to visualize what examples it covers, and for one example which it does not cover, we can see which premise is actually not matched by the example.

With DICE, we then see that both the acquisition of the axiomatic background knowledge and the visualization of its effects on the learning base is made easier and faster. These are two features which are crucial to an efficient discovery environment. Once the set of axioms is defined, it is given to ENIGME by the environment. Enigme itself will use CLIPS, the inference engine, to enrich the example base using the axioms, thereby creating what we have called the learning base, i.e. the set of learning examples used to perform the inductive learning.

2.4. Induction

Restricting the Exploration Space

In this section, we will mainly be concerned with the constraints that allow the system to go only through subparts of the whole search space defined by the concept language bias. As has been asserted before, we want the constraints to be interpretable by the user. To this end, the Enigme system (Thomas et al 93) is integrated in DICE. Enigme uses a Problem Solving Method (PSM) to constrain the search. PSMs have been highlighted in recent research in knowledge acquisition (Wielinga, 92) in order to guide the elicitation of the domain knowledge. A PSM can be seen as meta-knowledge for this domain knowledge. Its purpose is to describe the task the final expert system has to solve. Furthermore, a PSM decomposes this task it in a set of problem solving steps (i.e., primitive subtasks) and describes the control over these PS-steps thanks to a few control primitives (usually loops, conditional statements, and so on). In the framework of KDD, such a PSM allows the user to describe the purpose of the rules to be produced.

For instance, the simpler (and the less effective) way to use the PSM is to define the target concept(s) and the attributes to be used in this definition through one rough step linking directly the usable attributes to the target concept(s). However, if the user has further knowledge, he can introduce intermediate steps (hence intermediate concepts[1]), and so describe the reasoning the final system has to follow in order to decide if a particular case is an instance of the target concept(s). Since the size of the search space is exponential with the number of attributes, if all the attributes are not involved as input of one of the step, the

[1]This intermediate concepts can either be present in the database, either be computed by the axiomatic knowledge (see section 2.3).

gain will be significant. Furthermore, if there are several ways to reach the solution, this reasoning can include control statements. ENIGME uses such control statements to choose the relevant examples on each step[2]. This is particularly important in the KDD framework since the choice of only relevant examples on one step allows the system to not include the control information in the rules themselves, and so to produce more readable rules.

Exploration parameters definition

The choice of the remaining induction parameters is somewhat more complex since they do not have a semantic content as rich as the other ones described here. Their values are usually obtained by experimentation, considering as one of the main criteria the number of rules induced by the system compared to the desired number of rules. The experimental nature of this choice is one subsequent justification for the need for an environment such as DICE, enabling a quick and efficient evaluation of the effect of these parameters on induction.

2.5. *Induction and Results Presentation*

A very crucial operation allowed within DICE is the induction from a learning base and a learning bias. This is done by the inductive algorithm CHARADE (Ganascia,). We do not want to go into the detail of the functioning of this algorithm here, but what is important to remember is that the a priori bias of CHARADE has been put to a minimum, so that most of the inductive bias is defined as semantically rich knowledge. This induction produces a rule base that will eventually contain the knowledge discovered. Though the learning bias has been used to restrain the number of possible induction and to guide the induction, during the experimentation, it is likely to happen that the number of induced rules remains relatively high. Therefore, it is particularly important to have available some means to visualize, organize and analyze the results. With DICE, this is done through the output interface. Rules can be studied according to the set of examples they cover, and some of them can optionally be transferred to an axiomatic knowledge base so that they can be used as part of a new learning bias in the subsequent experimentations.

3. Applications: experimenting with DICE

3.1. *An experiment with Chinese characters*

DICE is currently experimented within the framework of an interdisciplinary research program called EURECAR whose goal is to provide a system based on machine learning techniques to support the analysis of the construction of Chinese characters (Zucker & Mathieu, 93). Chinese is radically different from other languages as it is a non-alphabetic language and is not a phonetic description of the language (Granet, 88). In Chinese, a sentence is a set of characters that can be a word or a component of a word. The mains facets of a character are its meanings, pronunciations and written form. For example the Chinese character meaning *feelings or situation* is pronounced qīng (this notation, called

[2]The user can also use these control statements to learn rules using only examples meeting a certain condition.

Pinyin, is one of the three main representations of character pronunciation) and is written 情. It can be described as containing the two radicals 忄 and 青 which are respectively identified as "radical 61" and "radical 174" in the Kang Xi dictionary (Zhang, 79) which contains 214 radicals.

To discover concepts related to the characters, it appeared natural to take the entity *character* as the initial morion. Characters written form can be represented by a set of *radicals* (忄 and 青) assembled in various ways (left /right) to form the character (情). We have therefore considered *radical* as a sub-morion. Each radical can be decomposed into a set of elementary strokes (丨, ㇕, 一, ...). We have therefore considered *stroke* as a sub-morion of *radical*. Figure 5 summarizes the structure of the initial morion acquired using the DICE system. Having acquired the initial morion, we then acquired the 400 characters contained in the Bellassen method for learning Chinese characters (Bellassen & Pengpeng, 91). They are amongst the most frequent characters and represent 67% of the characters printed in standard publications in China. In collaboration with sinologues, we have identified about twenty predicates to describe the characters.

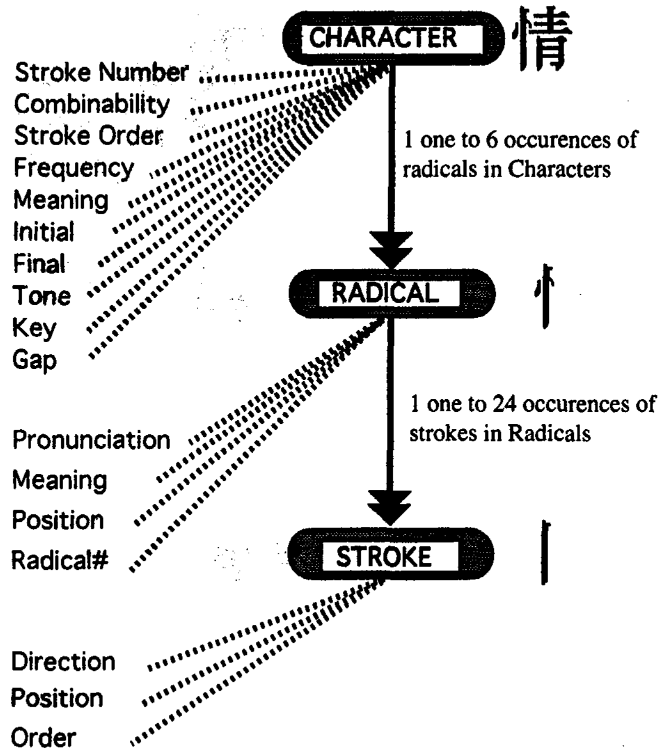

Figure 5: The main morions identified in characters

DICE has reformulated the initial database of characters using various morions: a 1-radical entity to discover similarities in characters that could be expressed using one radical at a time. Other morions have been then used such as a 2-radical morion, a 3-radical morion, etc. In the first reformulation, *radicals* were matched to one another and many relations between the presence of a radical and some properties of the characters have been discovered, for instance:

If Radical(x,y) and Radical#(y,174) and Position(y,"right") Then Final(x,"ing")

If Radical(x,y) and Radical#(y,174) and Position(y,"right") Then (x,"palatal")

3.2. *An experiment with a complex medical database*

Most AI, machine learning or KDD studies applied to the field of medicine have tackled the tasks of diagnosis or patient monitoring. Our own approach is to confront and develop techniques in the field of medical research. More specifically, within the framework of a collaboration with a team of medical experts, we are currently studying how formal induction techniques can aid the research on certain types of human leukemia (Corruble V. & Ganascia, J.G., 1994).

Some databases of patient records have already been established by different teams of physicians. However, the complexity of the data, and also the absence of any available comprehensive theories on leukemia has rendered very difficult the exploitation of these data by traditional techniques (e.g. statistics). These are two reasons why some physicians have shown great interest in the application of KDD techniques to this problem.

The data on which we have experimented so far contain records of patients suffering from Myelodysplasia. This syndrome, which regroups different affections, leads in a certain number of cases to a real leukemia. This is why it is often referred to as pre-leukemia. Each patient record contains a large number of fields describing the background of the patient, the evolution of the blood analyses, of the diagnosis, of the treatment, the development or absence of leukemia, the survival, etc. Since the reason why some patients develop a leukemia and other do not has not yet been formally established, the use of KDD techniques could be extremely ground-breaking for this problem.

The DICE system appears very useful for this project. Indeed, as we have seen earlier, though the database implicitly suggests to take a patient record as natural learning example, DICE allows to experiment on the use of different morions (one consultation, ...).

At another level, we want to insist here on the use of axiomatic knowledge on two points. Since many of the attributes are numerical, we need to discretize them by creating symbolic attributes for two reasons : on one hand, symbolic attributes make more sense to the human expert so that the expert using the environment will be able to interpret easily the results provided, on the other hand, the computational cost of induction on symbolic attributes (having therefore a limited number f possible values) is much reduced. For these two reasons, the use of axioms which express the values of the symbolic attributes given the values of the numerical attributes appears very useful. In this respect, DICE appears interesting because it provides an easy way to experiment on the values of the thresholds of the mapping from the numeric to the symbolic attribute. The possibility of experimentation is crucial: since we are actually researching, there is not certainty that the threshold currently used are adequate for our particular problem.

Another use of axiomatic knowledge for this problem is the introduction of hypothetical abstract concepts that could take part in the explanatory/discovery process that we are trying to perform. As we have shown in our study of scurvy (Corruble V. & Ganascia, J.G., 1993), all the concepts needed are not necessarily given in the database, so that it can be necessary to introduce them as functions of other concepts. This is another possibility offered by DICE. In our case, a particular model that could be of interest is a model of cancer inspired by an analogy with the theory of immunity. In this framework, we can introduce for example concepts that represent the fighting power of the human body against the developing cancer. Having described precisely the underlying mechanism, we can then test directly (inductively) the explanatory power of this model compared to other competing ones.

Conclusion

Many KDD systems taking inspiration from Machine Learning research have concentrated on the algorithmic part of the knowledge discovery process. The work presented here results from a more comprehensive approach to knowledge discovery that encompasses the need for a learning bias to guide and restrain the induction of new knowledge. In order to take this learning bias into account efficiently, we must be able to express it in such a way that it is as semantically rich as possible, so that it becomes meaningful to manipulate this bias through experimentations. The DICE environment is precisely designed in this purpose: it provides both the means to express three essential kinds of learning bias which are semantically rich, and the means to induce knowledge given a learning base and a learning bias.

The expression of the three kinds of bias is done through three distinct modules within the DICE environment which permit respectively to reformulate the initial database, to enrich it through the use of some axiomatic knowledge, and to define a problem solving method that guides further the induction. The choice of the sequence in which these modules are used is left to the user who can therefore navigate and experiment with each one of them.

A significant part of the knowledge discovery process is left to the user by DICE, and there is currently no hope of finding a fully automated method of bias construction. But DICE contributes significantly to the expression and the experimentation on this biases. A potentially fruitful direction for investigation would be to look for heuristics that could guide the construction of the bias by the user.

References

Bellassen J. & Pengpeng Z., "*Perfectionnement à la langue et à l'écriture chinoises*", pp. 296-347, La Compagnie, Paris, France, 1991.

Corruble V., Ganascia J.G., 1993. Discovery of the Causes of Scurvy: Could artificial intelligence have saved a few centuries ? AAAI 1993. Proceedings of the Workshop on Knowledge Discovery in Databases.

Corruble V., Ganascia J.G., 1994. Aid to discovery in medicine using formal induction techniques. *Blood Cells*. 19. 1994.

Drastal G. & Czako G., "Induction in an abstraction space: A form of constructive induction", in *Proceedings* of *the twelfth International Jointed Conference in Artificial Intelligence (IJCAI)*, pp. 708-712, 1989.

Ganascia J-G., "CHARADE: A rule System Learning System", in *Proceedings of the tenth International Jointed Conference in Artificial Intelligence (IJCAI)*, Milan, 23-28 August 1987.

Ganascia J-G., "Deriving the Learning Bias from Rule Properties.", Machine Intelligence (12), Oxford University Press, Oxford, 1991.

Granet M., "*La pensée chinoise*", Editions Albin Michel, pp. 33-42, 1988.

Jianping J. and Michalski R., "A Description of preference Criterion in a constructive Learning", in *Proceeding of the sixth international Workshop on Machine Learning*, pp. 17-22, Morgan Kaufmann Publishers, California, 1989.

Lapointe S., Ling C. & Matwin S., "Constructive Inductive Logic Programming", *Proceedings of the 14th IJCAI*, pp. 1030-1036, 1993.

Lavrac N. , Dzeroski, S., & Grobelnik M., "Learning non-recursive definitions of Relations with LINUS.", in *Proceedings of the European Working Session on Learning,* pp. 265-281, Springer-Verlag (482), 1991.

Matheus J., "The Need for Constructive Induction", in *Proceeding of the eighth international Workshop on Machine Learning*, pp. 173-177, Morgan Kaufmann Publishers, California, 1991a.

Matheus J., "Adding Knowledge to SBL through Feature Construction", in *Proceedings of the European Working Session on Learning,* pp. 803-808, Springer-Verlag (482), 1991b.

Michalski R.S., "A theory and methodology of inductive learning", in *Machine Learning: An artificial intelligence approach,* Tioga, 1983.

Mitchell T. ,"Generalization as Search", *Artificial Intelligence Journal 18.* pp. 203-226, 1982.

Morik, K., and Sommer, E., "Balanced Cooperative Modelling using Mobal 1.0: user guide", MLT ESPRIT project deliverable GMD/P2154/21/1, 1991.

Piatetsky-Shapiro G., "Knowledge Discovery in Databases", AAAI Press, distributed by MIT Press, 1991.

Riddle P., "Automating Problem Reformulation", *Change of Representation and Inductive Bias*, pp. 105-124, edited by D. Benjamin, Kluwer Academic Publisher, Boston, 1990.

Russell S.J. and Grossof B.N., Declarative Bias: An Overview, in Change of Representation and inductive bias, Benjamin D.P. (ed), Kluwer Academic Publisher, 1990

Subramanian D., "A theory of justified reformulation", *Change of Representation and Inductive Bias*, pp. 147-167 edited by D. Benjamin, Kluwer Academic Publisher, Boston, 1990.

Subramanian D., "Representational Issues in Machine Learning", in *Proceeding of the sixth international Workshop on Machine Learning*, pp. 426-429, Morgan Kaufmann Publishers, California, 1989.

Thomas J., Ganascia J.G. and Laublet P., Model-driven Knowledge Acquisition and Knowledge-biased Machine Learning: An example of a principled association, in the proceedings of the IJCAI'93 workshop 16, Chambery, 1993

Utgoff P., "Shift of Biais for Inductive Concept Learning", In Machine Learning 2. Morgan Kaufman. 107-148, 1986.

Wielinga, B., Schreiber, T. and Breuker, J. "KADS: a modelling approach to knowledge engineering", *Knowledge Acquisition*, Vol. 4, n. 1, 1992.

Zhang Y., "*Kang Xi Dictionary*", The Commercial Press, Beijing, 1979.

Zucker J-D. & Mathieu J., "Machine Learning Contributions to a Guided Discovery Tutoring Environment for Chinese Characters", in *Proceedings of the International Conference on Computers in Education*, pp. 25-30, Taiwan, 1993.

Zucker J-D. & Ganascia J-G., "Selective Reformulation of Examples in Concept Learning." In William W. Cohen & Haym Hirsh, eds., *Machine Learning: Proceedings of the eleventh International Conference,* Morgan Kaufmann Pubilshers, San Fransisco, CA.