A MODEL OF RULE-BASED LOGICAL INFERENCE

Xenia Naidenova

Abstract: *We proposed a unified model for combining inductive reasoning with deductive reasoning in the framework of inferring and using implicative logical rules. The key concept of our approach is the concept of a good diagnostic test. We define a good diagnostic test as the best approximation of a given classification on a given set of examples. The analysis of the good tests construction allows demonstrating that this inference engages both inductive and deductive reasoning rules.*

Key Words *learning logical rules from examples, machine learning, rule-based inference, good diagnostic test*

Introduction

Here we describe a model of common reasoning that has been acquired from our numerous investigations on the human reasoning modes used by experts for solving diagnostic problems in diverse areas such as pattern recognition of natural objects (rocks, ore deposits, types of trees, types of clouds e.t.c.), analysis of multi-spectral information, image processing, interpretation of psychological testing data, medicine diagnosis and so on. The principal aspects of this model coincide with the rule-based inference mechanism that is embodied in the KADS system (Ericson, et al., 1992), (Gappa and Poeck, 1992). More details related to our model of reasoning and its implementation can be found in (Naidenova and Syrbu, 1984), (Naidenova and Polegaeva, 1985a), (Naidenova and Polegaeva, 1985b), and (Naidenova, 2006).

We need the following three types of rules in order to realize logical inference:

INSTANCES or relationships between objects or facts really observed. Instance can be considered as a logical rule with the least degree of generalization. On the one hand, instances serve as a source of an expert's knowledge. On the other hand, instances can serve as a source of a training set of positive and negative examples for inductive inference of generalized rules.

RULES OF THE FIRST TYPE. These rules describe regular relationships between objects and their properties and between properties of different objects. The rules of the first type can be given explicitly by an expert or derived automatically from examples with the help of some learning process. These rules are represented in the form "if-then" assertions. They accumulate generalized knowledge in a problem domain.

RULES OF THE SECOND TYPE or inference rules with the help of which rules of the first type are used, updated and inferred from data (instances). The rules of the second type are reasoning rules.

Using the rules of the first type is artificially separated from the learning process. But it is clear that there is no reason to separate the process of learning rules from the process of using these rules for class identification or pattern recognition problems: both processes are interdependent and interconnected. Anyone of these processes can require executing the other. Anyone of these processes can be built into the other.

Any model of reasoning must also include STRATEGIES or the sequences of applying rules of all types in reasoning. The application of rules is conditioned by different situations occurring in the reasoning process and it is necessary to identify these situations. Strategies have a certain freedom as so it is possible to apply different rules in one and the same situation and the same rule in different situations. The choice of a strategy determines the speed, completeness, deepness and quality of reasoning.

Rules Acquired from Experts or Rules of the First Type

An expert's rules are logical assertions that describe the knowledge of specialists about a problem domain. Our experience in knowledge elicitation from experts allows us to analyze the typical forms of assertions used by experts. As an example, we give the rules of an expert's interpretation of data obtained with the use of Pathological Character Accentuation Inventory for Adolescents. This psycho-diagnostic method was elaborated by Lichko (1983) and is a classical example of an expert system.

Some examples of the expert's rules are:

"If (D - F) $\geq$ 4, then DISSIMULATION decreases the possibility to reveal any character accentuation and completely excludes the CYCLOID and CONFORM types of character".

"If the index E $>$ 4, then the CYCLOID and PSYCHASTENOID types are impossible".

"If the type of character is HYPERTHYMIA, then ACCENTUATION with psychopathies is observed in 75%, with transit disturbances – in 5%, and with stable adaptation – in 5% of all cases".

"If the index A $>$ 6 and the index S $>$ 7 and the index Con = 0 and the index D $>$ 6, then the LABILE type is observed".

"If the index E $\geq$ 6, then the SCHISOID and HYSTEROID types are observed frequently".

"If after the application of rules with the numbers *x*, *y*, *z* the values of at least two indices are greater than or equal to the minimal diagnostic threshold, then the mixed types are possible with the following consistent combinations of characters: Hyp - C, Hyp - N, Hyp - Hyst, C - L, L - A, L - S, and L - Hyst".

We used the following abbreviations: Hyp - hyperthymia, C - cycloid, L - labile, A – asthenia, N – neurotic, S - schizoid, Con - conformable, Hyst - hysteroid, Sens - sensitive, D - dissimulation, F - frankness, E - emancipation, and P - psychasthenia.

It is clear that an expert's assertions can be represented with the use of only one class of logical rules, namely, the rules based on implicative dependencies between names.

Implication: *a*, *b*, *c* $\rightarrow$ *d*. This rule means that if the values standing in the left side of the rule are simultaneously true, then the value in the right side of the rule is always true.

An implication *x* $\rightarrow$ *d* is satisfied if and only if the set of situations in which *x* appears is included in the set of situations in which d appears.

Interdiction or forbidden rule (a special case of implication) *a*, *b*, *c* $\rightarrow$ false (never). This rule interdicts a combination of values enumerated in the left side of the rule. The rule of interdiction can be transformed into several implications such as *a*, *b* $\rightarrow$ not *c*; *a*, *c* $\rightarrow$ not *b*; and *b*, *c* $\rightarrow$ not *a*.

Compatibility: *a*, *b*, *c* $\rightarrow$ *rarely*; *a*, *b*, *c* $\rightarrow$ *frequently*. This rule says that the values enumerated in the left side of the rule can simultaneously occur rarely (*frequently*). The rule of compatibility presents the most frequently observed combination of values that is different from a law or regularity with only one or two exceptions.

Compatibility is equivalent to a collection of assertions as follows:

a, *b*, *c* $\rightarrow$ *rarely*	*a*, *b*, *c* $\rightarrow$ *frequently*,
a, *b* $\rightarrow$ *c rarely*	*a*, *b* $\rightarrow$ *c frequently*,
a, *c* $\rightarrow$ *b rarely*	*a*, *c* $\rightarrow$ *b frequently*,
b, *c* $\rightarrow$ *a rarely*	*b*, *c* $\rightarrow$ *a frequently*.

Diagnostic rule: x, $d \rightarrow a$; x, $b \rightarrow$ not a; d, $b \rightarrow$ *false*. For example, *d* and *b* can be two values of the same attribute. This rule works when the truth of '*x*' has been proven and it is necessary to determine whether '*a*' is *true* or not. If '*x* & *d*' is true, then '*a*' is *true*, but if '*x* & *b*' is true, then '*a*' is *false*.

Rule of alternatives: a or $b \rightarrow$ *true* (*always*); a, $b \rightarrow$ *false*. This rule says that '*a*' and '*b*' cannot be simultaneously true, either '*a*' or '*b*' can be true but not both.

Structure of the Knowledge Base

We describe a very simple structure of a knowledge base that is sufficient for our illustrative goal. The knowledge base (KB) consists of two parts: the Attribute Base (*AtB*), containing the relations between problem domain concepts, and the Assertion Base (*AsB*), containing the expert's assertions formulated in terms of the concepts.

The domain concepts are represented by the use of names. With respect to its role in the KB, a name can be one of two kinds: name of attribute and name of attribute value. However, with respect to its role in the problem domain, a name can be the name of an object, the name of a class of objects and the name of a classification or collection of classes. A class of objects can contain only one object hence the name of an object is a particular case of the name of a class. In the KB, names of objects and of classes of objects become names of attribute values, and names of classifications become names of attributes.

For example, let objects be a collection of trees such as asp, oak, fir-tree, cedar, pine-tree, and birch. Each name calls the class or the kind of trees (in a particular case, only one tree). Any set of trees can be partitioned into the separate groups depending on their properties. '*Kind of trees*' will be the name of a classification, in which '*asp*', '*oak*', '*fir-tree*', '*cedar*', '*pine-tree*', and '*birch*' are the names of classes. Then, in the KB, '*kind of trees*' will be used as the name of an attribute the values of which are '*asp*', '*oak*', '*fir-tree*', '*cedar*', '*pine-tree*', and '*birch*'. The link between the name of an attribute and the names of its values is implicative. It can be expressed by the following way:

$$(\langle\text{name of value}_1\rangle, \langle\text{name of value}_2\rangle, \ldots, \langle\text{name of value}_k\rangle) \rightarrow \langle\text{name of attribute}\rangle,$$

where the sign "$\rightarrow$" denotes the relation "is a".

In our example (*asp*, *oak*, *fir-tree*, *cedar*, *pine-tree*, *birch*) $\rightarrow$ *kind of trees*, and, for each value of '*kind of trees*', the assertion of the following type can be created: "*asp* is a *kind of trees*".

The set of all attributes' names and the set of all values' names must not intersect. This means that the name of a classification cannot simultaneously be the name of a class. However, this is not the case in natural languages: the name of a class can be used for some classification and vice versa. For example, one can say that '*pine-tree*', '*fir-tree*', '*cedar*' are '*conifers*'. But one may also say that '*conifers*', '*leaf-bearing*' are '*kinds of trees*'. Here the word '*conifers*' serves both as the name of a classification and as the name of a class. In this setting, class is a particular case of classification like object is a particular case of class.

By using names in the way we do in real life we permit the introduction of auxiliary names for the subsets of the set of an attribute's values. Let *A* be an attribute. The name of a subset of values of *A* will be used as the name of a new attribute which, in its turn, will serve as the name of a value with respect to *A*.

The *AsB* (Assertion Base) contains the expert's assertions. Each assertion links a collection of values of different attributes with a certain value of a special attribute (SA) that evaluates how often this collection of values appears in practice. The values of a special attribute are: *always*, *never*, *rarely*, and *frequently*. Assertions have the following form:

$$(\langle\text{name of value}\rangle, \langle\text{name of value}\rangle, \ldots, \langle\text{value of SA}\rangle) = \textit{true}.$$

For simplicity, we omit the word '*true*', because it appears in any assertion. For example, the assertion "*pine-tree* and *cedar* can be found *frequently* in the meadow type of forest" will be expressed in the following way: (*meadow*, *pine-tree*, *cedar*, *frequently*). We also omit the sign of conjunction between values of different attributes and the sign of disjunction (separating disjunction) between values of the same attribute. For example, the assertion in the form (*meadow*, *pine-tree*, *cedar*, *frequently*) is equivalent to the following expression of formal logic: P((type of forest = *meadow*) & ((kind of trees = *pine-tree*) V (kind of trees = *cedar*)) & (SA = *frequently*)) = *true*.

Only one kind of requests to the KB is used: SEARCHING VALUE OF <name of attribute> [,<name of attribute>,...] IF (<name of value>, <name of value>, ...), where "name of value" is the known value of an attribute, "name of attribute" means that the value of this attribute is unknown. For example, the request "to find the type of forest for a region with plateau, without watercourse, with the prevalence of pine-tree" will be represented as follows: SEARCHING VALUE OF the type of forest IF (*plateau*, *without watercourse*, *pine-tree*).

Deductive Reasoning Rules of the Second Type

The following rules of the second type lie in the basis of the reasoning process for solving diagnostic or pattern recognition tasks. Let *x* be a collection of true values observed simultaneously.

Using implication. Let *r* be an implication, left(*r*) be the left part of *r* and right(*r*) be the right part of *r*. If left(*r*) $\subseteq$ *x*, then *x* can be extended by right(*r*): $x \leftarrow x \cup$ right(*r*).

For example, *x* = '*a*, *b*, *c*, *d* ', *r* = '*a*, *d* $\rightarrow$ k ', $x \leftarrow x \cup k$.

Using implication is based on modus ponens: if *A*, then *B*; *A*; hence *B*.

Using interdiction. Let *r* be an implication *y* $\rightarrow$ not *k*. If left(*r*) $\subseteq$ *x*, then k is a forbidden value for all the extensions of *x*.

Using interdiction is based on modus ponendo tollens:

either *A* or *B* (*A*, *B* – alternatives); *A*; hence not *B*;

either *A* or *B*; *B*; hence not *A*.

Using compatibility. Let *r* = '*a*, *b*, *c* $\rightarrow$ *k*, *rarely* (*frequently*)'.

If left(*r*) $\subseteq$ *x*, then *k* can be used for an extension of *x* with the value of SA equal to '*rarely*' ('*frequently*'). The application of several rules of compatibility leads to the appearance of several values 'rarely' and/or '*frequently*' in the extension of *x*. Computing the value of SA for the extension of *x* requires special consideration. In any case, the appearance of at least one value '*rarely*' means that the total result of the extension will have the value of SA equal to '*rarely*'. Two values equal to '*frequently*' lead to the result 'less *frequently*', three values equal to '*frequently*' lead to the result 'less less *frequently*' and hence the values '*rarely*' and '*frequently*' must have the ordering scale of measuring.

Using compatibility is based on modus ponens.

Using diagnostic rules. Let *r* be a diagnostic rule such as '*x*, *d* $\rightarrow$ *a*; *x*, *b* $\rightarrow$ not *a*', where '*x*' is true, and 'a', 'not a' are hypotheses or possible values of some attribute. There are several ways for refuting one of the hypotheses: to infer either *d* or *b* with the use of knowledge base (*AtB*, *AsB*); to involve new instances from the database and/or new assertions from the knowledge base for inferring new diagnostic rules distinguishing the hypotheses '*a*' and 'not *a*'; or, eventually, ask an expert which of the values *d* or *b* is true.

Our experience shows that generally the experts have in their disposal many diagnostic rules corresponding to the most difficult diagnostic situations in their problem domain.

Using a diagnostic rule is based on modus ponens and modus ponendo tollens.

Using rule of alternatives. Let '*a*', '*b*' be two alternative hypotheses about the values of some attribute. If one of these hypotheses is inferred with the help of reasoning operations, then the other one is rejected.

Using a rule of alternatives is based on modus tollendo ponens: either *A* or *B* (*A*, *B* – alternatives); not *A*; hence *B*; either *A* or *B*; not *B*; hence *A*.

The operations enumerated above can be named as "forward reasoning" rules. Experts also use implicative assertions in a different way. This way can be named as "backward reasoning".

Generating hypothesis. Let *r* be an implication $y \to k$. Then the following hypothesis is generated "if *k* is true, then it is possible that *y* is true".

Using modus tollence. Let r be an implication $y \to k$. If 'not *k* ' is inferred, then 'not *y*' is also inferred.

Natural diagnostic reasoning is not any method of proving the truth. It has another goal: to infer all possible hypotheses about the value of some target attribute. These hypotheses must not contradict with the expert's knowledge and the situation under consideration. The process of inferring hypotheses is reduced to extending maximally a collection *x* of attribute values such that none of the forbidden pairs of values would belong to the extension of *x*.

Inductive Reasoning Rules of the Second Type

Inductive steps of reasoning consist of using already known facts and statements, observations and experience for inferring new logical rules of the first type or correcting those that turn out to be false.

For this goal, inductive rules of reasoning are applied. The main forms of induction are the canons of induction that have been formulated by English logician Mill (1900). These canons are known as the five induction methods of reasoning: method of only similarity, method of only distinction, joint method of similarity-distinction, method of concomitant changes, and method of residuum.

The method of only similarity: This rule means that if the previous events (values) *A*, *B*, *C* lead to the events (values) *a*, *b*, *c* and the events (values) *A*, *D*, *E* lead to the events (values) *a*, *d*, *e*, then *A* is a reason of *a*.

The method of only distinction: This rule means that if the previous events (values) *A*, *B*, *C* give rise to the events (values) *a*, *b*, *c* and the events (values) *B*, *C* lead to the events (values) *b*, *c*, then *A* is a reason of *a*.

The joint method of similarity-distinction: This method consists of applying two previous methods simultaneously.

The method of concomitant changes: This rule means that if the change of a previous event (value) *A* is accompanied by the change of an event (value) *a*, and all the other previous events (values) do not change, then *A* is a reason of *a*.

The method of residuum: Let *abcd* be a complex phenomenon, *A* be the reason of *a*, *B* be the reason of *b*, and *C* be the reason of *c*. Then it is possible to suppose that there is an event *D* which is a reason of *d*.

An Example of the Reasoning Process

Let the content of the Knowledge Base be the following collection of assertions:

AtB:

1. (*meadow*, *bilberry wood*, *red bilberry wood* ....) $\to$ *types of woodland*;
2. (*pine-tree*, *spruce*, *cypress*, *cedars*, *birch*, *larch*, *asp*, *fir-tree*) $\to$ *dominating kinds of trees*;
3. (*plateau*, *without plateau*) $\to$ *presence of plateau*;
4. (*top of slope*, *middle part of slope*, ….) $\to$ *parts of slope*;
5. (*peak of hill*, *foot of hill*) $\to$ *parts of hill*;

6. (*height on plateau*, *without height on plateau*) $\rightarrow$ *presence of a height on plateau*;
7. (*head of watercourse*, *low part of watercourse*, ….) $\rightarrow$ *parts of water course*;
8. (*steepness* $\geq 4°$, *steepness* $\leq 3°$, *steepness* $< 3°$, …) $\rightarrow$ *features of slope*;
9. (*north*, *south*, *west*, *east*) $\rightarrow$ *the four cardinal points*;
10. (*watercourse*, *without watercourse*) $\rightarrow$ *presence of a watercourse*.

AsB:

11. (*meadow*, *pine-tree*, *larch*, *frequently*);
12. (*meadow*, *pine-tree*, *steepness* $\leq 4°$, *never*);
13. (*meadow*, *larch*, *steepness* $\geq 4°$, *never*);
14. (*meadow*, *north*, *west*, *south*, *frequently*);
15. (*meadow*, *east*, *rarely*);
16. (*meadow*, *fir-tree*, *birch*, *asp*, *rarely*);
17. (*meadow*, *plateau*, *middle part of slope*, *frequently*);
18. (*meadow*, *peak of hill*, *watercourse heads*, *rarely*);
19. (*plateau*, *steepness* $\leq 3°$, *always*);
20. (*plateau*, *watercourse*, *rarely*);
21. (*red bilberry wood*, *pine-tree*, *frequently*);
22. (*red bilberry wood*, *larch*, *rarely*);
23. (*red bilberry wood*, *peak of hill*, *frequently*);
24. (*red bilberry wood*, *height on plateau*, *rarely*);
25. (*meadow*, *steepness* $< 3°$, *frequently*).

Let x be a request to the KB equal to:

SEARCHING VALUE OF type of woodland IF (*plateau*, *without watercourse*, *pine-tree*).

The process of reasoning evolves according to the following sequence of steps:

Step 1. Take out all the assertions t in *AsB* containing at least one value from the request, i.e. $t \in AsB$, $t \cap x \neq \varnothing$, where x is the request. These are assertions 11, 12, 17, 19, 20, 21, and 24.

Step 2. Delete (from the set of selected assertions) all the assertions that contradict the request. Assertion 20 contradicts the request because it contains the value of attribute '*presence of water course*' which is different from the value of this attribute in the request. The remaining assertions are 11, 12, 17, 19, 21, and 24. This step uses the rule of alternatives.

Step 3. Take out the values of attribute '*type of woodland* ' appearing in assertions 11, 12, 17, 19, 21, and 24. We have two hypotheses: '*meadow* ' and '*red bilberry* '. This step uses implications for generating hypotheses. It is a step of "forward reasoning". As a result, we have two extensions of the request:

SEARCHING VALUE OF the type of woodland IF (*plateau*, *without watercourse*, *pine-tree*, *meadow*?).

SEARCHING VALUE OF the type of woodland IF (*plateau*, *without watercourse*, *pine-tree*, *red bilberry*?).

The sign ' ? 'means that the values of type of woodland are hypotheses.

Step 4. An attempt is made to refute one of the hypotheses (the application of a diagnostic rule). For this goal, it is necessary to find an assertion that has the value of SA equal to '*never* ' and contains one of the hypotheses, some subset of values from the request and does not contain any other value. There is only one assertion with the value of SA equal to '*never* '. This is assertion 12: (*meadow*, *pine-tree*, *steepness* $\leq 4°$, *never*). However, we cannot use this assertion because it contains the value '*steepness* $\leq 4°$' which is not in the request.

Step 5. An attempt is made to find a value of some attribute that is not in the request (in order to extend the request). For this goal, it is necessary to find an assertion with the value of SA equal to '*always*' that contains a subset of values from the request and one and only one value of some new attribute the values of which are not in the request. Only one assertion satisfies this condition. This is assertion 19: (*plateau*, *steepness* $\leq 3°$, *always*).

Step 6. Forming the extended requests:

SEARCHING VALUE OF the type of woodland IF (*plateau*, *without watercourse*, *pine-tree*, *steepness* ≤ 3°, *meadow*?).

SEARCHING VALUE OF the type of woodland IF (*plateau*, *without watercourse*, *pine-tree*, *steepness* ≤ 3°, *red bilberry*?).

It is easy to see that Step 5 and Step 6 involve the rule of using implication in order to extend the requests.

Steps 1, 2, and 3 are repeated. Assertion 25 is involved in the reasoning.

Step 4 is repeated. Now assertion 12 can be used because the value '*steepness* ≤ 4° is in accordance with the values of '*feature of slope*' in the request. We conclude now that the type of woodland cannot be '*meadow*'. The non-refuted hypothesis is "the type of woodland = *red bilberry*". This step uses the interdiction rule in order to delete one of the hypotheses.

The process of pattern recognition can require inferring new rules of the first type from data, when it is impossible to distinguish inferred hypotheses. In general, there exist two main cases to learn rules of the first type from examples in the process of pattern recognition: i) the result of reasoning contains several hypotheses and it is impossible to choose one and only one of them (uncertainty), and ii) there does not exist any hypothesis.

An approach to Inferring Rules of the First type

Our approach to learning implicative rules from data is based on the concept of a good diagnostic (classification) test. A good classification test can be understood as an approximation of a given classification on a given set of examples (Naidenova and Polegaeva, 1986; Naidenova, 1996).

A good diagnostic test is defined as follows. Let R be a set of examples and $S = \{1, 2, \ldots i, \ldots, n\}$ be the set of indices of examples, where n is the number of example of R. Let $R(+)$ and $S(+)$ be the set of positive examples and the set of indices of positive examples, respectively. By $R(-) = R/R(+)$ denote the set of negative examples. Let U be the set of attributes and T be the set of attributes values (values, for short) each of which appears at least in one of the examples of R.

Denote by $s(A)$, $A \in T$ the subset $\{i \in S$: A appears in t_i, $t_i \in R\}$, where $S = \{1, 2, .., n\}$.

Following (Cosmadakis, et al., 1986), we call $s(A)$ the interpretation of $A \in T$ in R. The definition of $s(A)$ can be extended to the definition of $s(t)$ for any collection t, $t \subseteq T$ of values as follows:

if $t = A_1 A_2 \ldots A_m$, then $s(t) = s(A_1) \cap s(A_2) \cap \ldots \cap s(A_m)$.

Definition 1. A collection $t \subseteq T$ ($s(t) \neq \emptyset$) of values is a diagnostic test for the set $R(+)$ of examples if and only if the following condition is satisfied: $t \not\subset t^*$, $\forall t^*$, $t^* \in R(-)$ (the equivalent condition is $s(t) \subseteq S(+)$).

Let k be the name of a set $R(k)$ of examples. To say that a collection t of values is a diagnostic test for $R(k)$ is equivalent to say that it does not cover any example t^*, $t^* \notin R(k)$. At the same time, the condition $s(t) \subseteq S(k)$ implies that the following implicative dependency is true: 'if t, then k'. Thus a diagnostic test, as a collection of values, makes up the left side of a rule of the first type.

It is clear that the set of all diagnostic tests for a given set $R(+)$ of examples (call it 'DT(+)') is the set of all the collections t of values for which the condition $s(t) \subseteq S(+)$ is true. For any pair of diagnostic tests t_i, t_j from DT(+), only one of the following relations is true: $s(t_i) \subseteq s(t_j)$, $s(t_i) \supseteq s(t_j)$, $s(t_i) \approx s(t_j)$, where the last relation means that $s(t_i)$ and $s(t_j)$ are incomparable, i.e. $s(t_i) \not\subset s(t_j)$ and $s(t_j) \not\subset s(t_i)$. This consideration leads to the concept of a good diagnostic test.

Definition 2. A collection $t \subseteq T$ ($s(t) \neq \varnothing$) of values is a good test for the set $R(+)$ of examples if and only if $s(t) \subseteq S(+)$ and simultaneously the condition $s(t) \subset s(t^*) \subseteq S(+)$ is not satisfied for any t^*, $t^* \subseteq T$, such that $t^* \neq t$.

Now we shall give the following definitions.

Definition 3. A collection t of values is irredundant if for any value $v \in t$ the following condition is satisfied: $s(t) \subset s(t/v)$.

If a collection t of values is a good test for $R(+)$ and, simultaneously, it is an irredundant collection of values, then any proper subset of t is not a test for $R(+)$.

Definition 4. A collection $t \subseteq T$ of values is maximally redundant if for any implicative dependency $X \to v$, which is satisfied in R, the fact that t contains X implies that t also contains v.

If t is a maximally redundant collection of values, then for any value $v \notin t$, $v \in T$ the following condition is satisfied: $s(t) \supset s(t \cup v)$. In other words, a maximally redundant collection t of values covers the number of examples greater than any collection ($t \cup v$) of values, where $v \notin t$.

If a diagnostic test t for a given set $R(+)$ of examples is a good one and it is a maximally redundant collection of values, then for any value $v \notin t$, $v \in T$ the following condition is satisfied: ($t \cup v$) is not a good test for $R(+)$.

Any example t in R is a maximally redundant collection of values because for any value $v \notin t$, $v \in T$ $s(t \cup v)$ is equal to $\varnothing$.

For example, in Table 1 the collection '*Blond Bleu*' is a good irredundant test for class 1 and simultaneously it is maximally redundant collection of values. The collection '*Blond Embrown*' is a test for class 2 but it is not good and simultaneously it is maximally redundant collection of values.

The collection '*Embrown*' is a good irredundant test for class 2. The collection '*Red*' is a good irredundant test for class 1. The collection '*Tall Red Bleu*' is a good maximally redundant test for class 1.

It is clear that the best tests for pattern recognition problems must be good irredundant tests. These tests allow constructing the shortest rules of the first type with the highest degree of generalization.

Table 1. Example 1of Data Classification (This example is adopted from (Ganascia, 1989))

Index of example	Height	Color of hair	Color of eyes	Class
1	Low	Blond	Blue	1
2	Low	Brown	Blue	2
3	Tall	Brown	Embrown	2
4	Tall	Blond	Embrown	2
5	Tall	Brown	Blue	2
6	Low	Blond	Embrown	2
7	Tall	Red	Blue	1
8	Tall	Blond	Blue	1

One of the possible ways for searching for good irredundant tests for a given class of positive examples is the following: first, find all good maximally redundant tests; second, for each good maximally redundant test, find all good irredundant tests contained in it. This is a convenient strategy as each good irredundant test belongs to one and only one good maximally redundant test with the same interpretation (Naidenova, 1999).

Inductive Rules of the Second Type

We use the lattice theory as the mathematical model for constructing good classification tests. We define a diagnostic test as a dual object (Naidenova, 2001), i. e. as an element of the concept lattice introduced in the Formal Concept Analysis (Wille, 1992).

The links between dual elements of concept lattice reflect both inclusion relations between concepts (structural knowledge) and implicative relations between concept descriptions (deductive knowledge).

Inferring the chains of lattice elements ordered by the inclusion relation lies in the foundation of generating all types of diagnostic tests. We use the following variants of inductive transition from one element of a chain to its nearest element in the lattice:

(i) from $s_q = (i_1, i_2, \ldots, i_q)$ to $s_{q+1} = (i_1, i_2, \ldots, i_{q+1})$;

(ii) from $t_q = (A_1, A_2, \ldots, A_q)$ to $t_{q+1} = (A_1, A_2, \ldots, A_{q+1})$;

(iii) from $s_q = (i_1, i_2, \ldots, i_q)$ to $s_{q-1} = (i_1, i_2, \ldots, i_{q-1})$;

(iv) from $t_q = (A_1, A_2, \ldots, A_q)$ to $t_{q-1} = (A_1, A_2, \ldots, A_{q-1})$.

We have constructed the special rules for realizing these inductive transitions: the generalization rule, the specification rule, the inductive diagnostic rule, and the dual inductive diagnostic rule.

The Generalization Rule

The generalization rule is used to get all the collections of indices $s_{q+1} = \{i_1, i_2, \ldots i_q, i_{q+1}\}$ from a collection $s_q = \{i_1, i_2, \ldots i_q\}$ such that $t(s_q)$ and $t(s_{q+1})$ are tests for a given class of positive examples.

The termination condition for constructing a chain of generalizations is: for all the extension s_{q+1} of s_q, $t(s_{q+1})$ is not a test for a given class of positive examples.

The Specification Rule

The specification rule is used to get all the collections of values $t_{q+1} = \{A_1, A_2, \ldots, A_{q+1}\}$ from a collection $t_q = \{A_1, A_2, \ldots, A_q\}$ such that t_q and t_{q+1} are irredundant collections of values and they are not tests for a given set of positive examples.

The termination condition for constructing a chain of specifications is: for all the extensions t_{q+1} of t_q, t_{q+1} is either a redundant collection of values or a test for a given set of positive examples.

The Inductive Diagnostic Rule

The inductive diagnostic rule is used to get a collection of values $t_{q+1} = \{A_1, A_2, \ldots, A_{q+1}\}$ from a collection $t_q = \{A_1, A_2, \ldots A_q\}$ such that t_q is not a test but t_{q+1} is a test for a given set of positive examples.

We extend t_q by choosing values which appear simultaneously with it in the examples of $R(+)$ and do not appear in any example of $R(-)$. These values are to be said essential ones.

The Dual Inductive DiagnostIc Rule

The dual inductive diagnostic rule is used to get a collection of indices $s_{q-1} = (i_1, i_2, \ldots, i_{q-1})$ from a collection $s_q = (i_1, i_2, \ldots, i_q)$ such that $t(s_{q-1})$ is a test but $t(s_q)$ is not a test for a given set of positive examples. This rule uses a method for choosing indices admissible for deleting from s_q. By analogy with an essential value, we define an essential example (Naidenova, 2005).

The rules for constructing diagnostic tests as elements of dual lattice generate logical rules of the first type, as shown in Table 2.

Table 2. Deductive Rules of the First Type Obtained with the Use of Inductive Rules for Inferring Diagnostic Tests

Inductive rules	Action	Inferring deductive rules of the first type
Generalization rule	Extending *s* (narrowing *t*)	Implications
Specification rule	Extending *t* (narrowing *s*)	Implications
Inductive diagnostic rule	Searching for essential values	Diagnostic rules
Dual inductive diagnostic rule	Searching for essential examples	Compatibility rules or (approximate implications)

The analysis of the inference for lattice construction allows demonstrating that this inference engages both inductive and deductive reasoning rules of the second type.

Both the generalization and specification rules realize the joint method of similarity-distinction. The extending of *s* results in obtaining the subsets of positive examples of more and more power with more and more generalized features (set of values). An algorithm NIAGaRa based on this variant of generalization rule is used in (Naidenova, 2006), for inferring good maximal redundant tests.

Both the inductive and dual inductive diagnostic rules are based on the inductive method of only distinction.

For example, a variant of the generalization rule involves the following deductive and inductive reasoning rules of the second type: the joint method of similarity-distinction, the rule of using forbidden rules of the first type, the method of only similarity, the rule of using implication, lattice operations.

It is important to note that the rules of the first type (implications, interdictions, rules of compatibility) generated during the lattice construction used immediately in this process.

Conclusion

This work is an attempt to transform a large class of machine learning tasks into a commonsense reasoning process based on using well-known deduction and induction reasoning rules. The key concept of our approach is the concept of a good diagnostic test. We have used the lattice theory as the mathematical model for constructing good diagnostic tests for learning implications from examples.

We have divided commonsense reasoning rules in two classes: rules of the first type and rules of the second type. The rules of the first type are represented with the use of implicative logical statements. The rules of the second type or reasoning rules (deductive and inductive) are rules with the help of which rules of the first type used, updated and inferred from data.

The analysis of the inference for lattice construction allows demonstrating that this inference engages both inductive and deductive reasoning rules of the second type.

Bibliography

[Cosmadakis et al., 1986] S. Cosmadakis, P. C. Kanellakis, N. Spyratos, “Partition Semantics for Relations”, *Journal of Computer and System Sciences*, Vol. 33, No. 2, pp.203-233, 1986.

[Ericson et al., 1992] H. Ericson, A. R. Puerta, and M. A. Musen, “Generation of Knowledge Acquisition Tools from Domain Ontologies”, International Journal of Human Computer Studies, Vol. 41, pp. 425-453, 1992.

[Ganascia, 1989] J.- Gabriel. Ganascia, “EKAW - 89 Tutorial Notes: Machine Learning”, *Third European Workshop on Knowledge Acquisition for Knowledge-Based Systems*, Paris, France, pp. 287-296, 1989.

[Gappa and Poeck, 1992.] U. Gappa, and K. Poeck, “Common Ground and Differences of the KADS and Strong Problem Solving Shell Approach”, EKAW – 92, Lecture Notes in Artificial Intelligence, No. 599, pp. 52-73, 1992.

[Lichko, 1983.] A A. E. Lichko, Psychopathies and Accentuations of Character of Teenagers, second edition, Leningrad, USSR, “Medicine”, 1983.

[Mill, 1900] J. S. Mill, *The System of Logic*, Russian Publishing Company “Book Affair”: Moscow, Russia, 1900.

[Naidenova, 1996] X. A. Naidenova, “Reducing Machine Learning Tasks to the Approximation of a Given Classification on a Given Set of Examples”, *Proceedings of the 5-th National Conference at Artificial Intelligence*, Kazan, Tatarstan, Vol. 1, pp. 275-279, 1996.

[Naidenova, 1999] X. A. Naidenova, “The Data-Knowledge Transformation”, in: “*Text Procesing and Cognitive Technologies”, Paper Collection*, editor Solovyev, V. D., - Pushchino, Russia, Vol. 3, pp. 130-151, 1999.

[Naidenova, 2001] X. A. Naidenova, “Inferring Good Diagnostic Tests as a Model of Common Sense Reasoning”, *Proceedings of the International Conference “Knowledge-Dialog-Solution” (KDS’2001)*, State North-West Technical University, Publishing House «Lan», Saint-Petersburg, Russia, Vol. II, pp. 501-506, 2001.

[Naidenova, 2005] X. A. Naidenova, “DIAGARA: an Incremental lgorithm for Inferring Implicative Rules from Examples”, International Journal “ Information Theories & Applications”, Vol. 12, pp. 171-186, 2005.

[Naidenova, 2006] X. A. Naidenova, “An Incremental Learning Algorithm for Inferring Logical Rules from Examples in the Framework of the Common Reasoning Process”, in: “*Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques”,* editors Evangelos Triantaphyllou and Giovanni Felici*,* Springer*, pp. 90-146, 2006.*

[Naidenova and Polegaeva, 1985a] X. A. Naidenova, J. G. Polegaeva, “A Model of Human Reasoning for Deciphering Forest’s Images and its Implementation on Computer”, Theses of Papers and Reports of School-seminar “Semiotic Aspects of the Formalization of Intellectual Activity”, Kutaisy, Georgia Soviet Socialist Republic, pp. 49-52, 1985a.

[Naidenova and Polegaeva, 1985b] X. A. Naidenova, J. G. Polegaeva, “The Project of Expert System GID KLARA – Geological Interpretation of Data Based on Classification and Pattern Recognition”, Report I-A VIII.2 10-3/35, “Testing and Mastering Experimental Patterns of Flying (Aircraft) and Surface Spectrometry Apparatus, Working out Methods of Automated Processing Multi-Spectral Information for Geological Goals”, All Union Scientific Research Institute of Remote Sensing Methods for Geology, 1985b.

[Naidenova and Syrbu, 1984] X. A. Naidenova, V. N. Syrbu, “Classification and Pattern Recognition Logic in Connection with the Problem of Forming and Using Knowledge in Expert Systems”, Theses of Papers of Republican Scientific-Technical Conference “Interactive Systems and Their Practical Application”, Mathematical Institute with Computer Center, Kishinev, Moldavia, pp. 10-13, 1984.

[Naidenova and Polegaeva, 1986] X. A. Naidenova, J. G. Polegaeva, “An Algorithm of Finding the Best Diagnostic Tests”, *The 4-th All Union Conference “Application of Mathematical Logic Methods”*, Theses of Papers, Mintz, G; E, Lorents, P. P. (Eds), Institute of Cybernetics, National Acad. of Sciences of Estonia, Tallinn, Estonia, pp. 63-67, 1986.

[Wille, 1992] R. Wille, “Concept Lattices and Conceptual Knowledge System”, *Computer Math. Appl*., Vol. 23, No. 6-9, pp. 493-515, 1992.

Author's Information

Naidenova Xenia Alexandrovna - Military medical academy, Saint-Petersburg, Stoikosty street, 26-1-248, naidenovaxen@gmail.com naidenova@mail.spbnit.ru.