ON THE INDUCTION OF "INTERESTING" RULES

YVES KODRATOFF

CNRS, LRI Bât. 490 Univ. Paris-Sud, F – 91405 Orsay Cedex
yk@lri.fr

This paper holds that one deep reason of the large success of a new scientific field, called Data Mining or Knowledge Discovery in Databases, is the admittance that computer programs can perform inductive thinking, induction being the process by which new models of the reality are built.

The beginning of this paper discusses the difference between numerical induction, where the model is built in terms of variables taking an infinity of numerical values, and symbolic induction where the model is built in terms of variables taking a finite (preferably, only a few) number of values, or even by Boolean variables.

The second part of the paper presents an instance of symbolic induction, based on an evaluation of the interestingness of the rules induced from examples using **Inductive** Text Mining (ITM). The better-known **deductive** text mining is called Information Extraction, and amounts to finding instances of a predefined pattern in a set of texts. ITM looks for unknown patterns or rules to discover inside a set of texts. We mainly discuss two of the problems of ITM: building ontologies of concepts, and extracting patterns.

1. INTRODUCTION: AUTOMATIC INDUCTION IS *VERY* DIFFICULT

We have to deal with two different problems, the first one of the difference between deduction and induction in Computer Science (CS) software, and the second one of the difference between numerical and symbolic software. This leads to four main different approaches to automated inference, the four of them have been explored.

CS has favored the production of deductive software, induction being the reserved domain of the humans. Moreover, computers are mostly used as "number crunchers." It follows that deductive numerical software constitutes the main bulk of CS products.

Deductive symbolic software receives a large amount of attention, mostly from academics. The theoretical part of CS calls this effort *Automated Deduction*, while the more applied side called it *Expert Systems* in the 80's, but left this wording to prefer nowadays either *Knowledge Management*, or *Deductive Data Bases*. The difference between numerical and symbolic deduction, somewhat roughly speaking, lies in the environment they use in order to perform the reasoning. Numerical deduction relies almost entirely on the semantics of real numbers, while symbolic deduction defines several semantics – different from the one of the real numbers – and performs only the inference steps compatible with the allowed semantic. The exact representation of knowledge is thus not significant, symbols can be numbers or letters, only the way to manipulate the symbols is significant.

Both numerical and symbolic induction generate new models as an output of their computations. At the beginning of their history (and still today for some people) they had tendency to differentiate each other in the same way as the two deductive paths did. The field known as Inductive Logic programming (ILP) is a good example of a field where non numerical data are handled following the rules of first order logic, up to the point that handling numbers is very cumbersome for ILP. On the contrary, and since the birth of Data Mining (DM), it has been acknowledged that both use, in fact, various numerical and logical techniques to perform or to validate their inductions. Thus the difference between numerical and symbolic induction does not lie in the way the computations are performed (as we said it is roughly the case for deduction), but rather in the form into which the induced model is delivered to the user.

Numerical inductive techniques build models expressed in terms of numerical infinite valued variables. As a consequence, the models they build are hardly comprehensible to the person non expert in these inductive numerical techniques. Three different fields have been developing these techniques. Outside CS, Statistics has developed the so-called *Exploratory Statistics*, also called *Data Analysis* when more linked to CS. Within CS, the people dealing with pattern recognition developed a very efficient tool, called the perceptron, which is able to treat linearly separable data. A major success was achieved in 1962, when Novikoff was able to evaluate the rate of convergence of the perceptron in terms of $(R/\rho)^2$, where R is the radius of the data, and ρ is of the order of the smallest distance between two data points belonging to different classes. As we shall see, this result is of large practical importance in the success of the modern systems. The treatment of non linearly separable data has been performed by a development of the perceptron known as *Neural Networks*. Around 1989, took place a convergence between the statistical approach and the neural networks ones which can be essentially attributed to Vapnik's vision of statistical learning (Vapnik, 1995). Since 1995, it has led to the development of many numerical learning discriminant systems, all gathered under the name of *Support Vector Machines* (SVM). Novikoff's theorem applies to the perception only, and the first theoretical result giving a good approximation of the real nature of learning for all learning discriminant machines is that, provided the space of hypotheses shows the property of finite Vapnik-Chervonenkis dimension d_{VC} (see, for instance, Vapnik, 1995; Christianini and Shawe-Taylor, 2000), provided some learning has been taking place on a first set of examples, it is possible to ensure that the learned discrimination ability will not degrade, within a given approximation, for all the examples. It points very clearly at the main difficulty of induction in this case: given N objects, the best way to recognize them with a 100% accuracy is to memorize them without learning. Induction can then take the form of a hidden "learning by heart" where as many parameters as necessary are introduced in order to simply "fit the data" without a new model being really built. If we limit the power of the recognition device by

limiting its d_{VC}, then we can ensure that a real amount of induction has been taking place. Notice however, that this result can be also interpreted as a negative result: if the space of hypotheses is weak enough to be unable to shatter more than d_{VC} points, and if we are lucky enough to have been learning something, then this learning is proved to be conserved on new examples. The *if-we-are-lucky-enough-to-have-been-learning-something* part is not so promising, especially since it exists another theorem, called familiarly the *no free lunch* theorem (Wolpert, 1996), stating that we have been indeed lucky to learn something. The no free lunch theorem states that, given a fixed induction device, the mean performance of this device on an infinity of problems is zero. Since one of the hypotheses of Vapnik-Chervonenkis' theorem is that we have been already learning something on some examples, the no free lunch theorem says that our chances to perform this feast are very feeble, except on very small subsets of problems.

Symbolic inductive techniques may use continuous variables, but through a process of discretization (*i.e.*, cutting into a finite number of intervals) build models in terms of discrete variables. As compared to a perceptron where the separating hyperplane can be moved at will, the only freedom we are left with is to move potentially separating hyperplanes perpendicular to the axis of the to-be-discretized variable. The case of ILP is even more rigid: the predicates have to be defined beforehand, which means that the discretization process does not take place during learning, but before learning. In that sense, we can only decrease the d_{VC} of our hypotheses spaces. Vapnik-Chervonenkis theorem then states that we need less many examples to learn correctly, if we did learn anything at all. It is quite striking to see that people in symbolic learning did not yet take the habit to evaluate the d_{VC} of their hypotheses spaces, the most probable reason of it will be given in Section 2.1, below. The main strength of symbolic induction is more of a social nature. If these discrete values make sense to a field expert (*i.e.*, an expert in the field described by the variables, as opposed to the scientist who is expert in inductive symbolic techniques) the model is at least a little understandable by the field expert. In many cases, even symbolic techniques fail to provide an understandable model, but this is then looked upon as a failure of the inductive technique. As a research field, this gives an altogether completely different way towards improvement: accuracy of the results is interesting but looked upon more as a validation technique than as a goal in itself: a grossly inexact model cannot be worth studying, but a slightly less precise one is much more interesting, if more understandable by the field user. We agree that symbolization of the data produces a severe loss of discrimination power, and when the results of symbolic induction are not understandable, they are thus vastly inferior to numerical techniques. Inversely, if they are understandable, and if the problem requires comprehensibility, then the price to pay is heavy but only symbolic methods can be used.

It is somewhat unexpected that a fifth option is available: it is in fact a mixed approach, combining symbolic and numeric knowledge. This is achieved by Bayesian networks that combine a symbolic structure expressing the dependencies between the variables, and numeric calculations of probability distributions. It is noticeable that their "deductive" power is very special since they implement both deduction (reasoning from the premises to the conclusions) and abduction, *i.e.*, reasoning from the consequences to the premises. Inductive learning of structures and probability tables from data is an inductive problem we shall speak of in Section 3.6.

Section 2 will discuss some properties of symbolic induction, Section 3 will present the problems of inductive text mining, Section 4 shows how the texts have to be preprocessed in order to transform them into datasets from which models of the texts can be built, and Section 5 will show various measures of "interest" for the rules induced from datasets, Bayesian networks being one of the very interesting options.

2. PROBLEMS WITH THE SEMANTICS OF THE RULES INDUCED FROM DATA

2.1. SYMBOLIC AND NUMERIC GENERALITY

Both symbolic and numeric induction evaluate the variation in generality, in a sense, this is the "generality distance" between the data and the model built. The definition of generality is however very different in the two communities.

Numeric induction sees generalization as a measure of the rate of convergence of the learning process. It is intuitively obvious that the less many data points are needed to build a satisfying model, the larger the generalization abilities. We pointed out the contradiction that spaces of hypotheses with a small d_{VC} will certainly have high generalization ability, but they will also solve a very small number of problems. Besides, the technique used by the SVM to solve non-linear problems increases very rapidly the dimension of the data space. In fact, it could be believed that in non-linear cases, the good solution is to leave apart separation by hyperplanes, and to use higher degrees separating surfaces. It turns out that the SVM approach took the opposite solution. It kept the hyperplane separation, but it modified the representation space of the data by applying transformations to the coordinates of this space. For instance, the gravitation law in $m_1 * m_2 / r^2$ is not linear and could never be discovered by a perceptron. Nevertheless, by applying the coordinate transformation

$(m_1, m_2, r) \rightarrow (\log(m_1), \log(m_2), \log(r)) = (x, y, z)$

5

Newton's law becomes linear: x + y – 2z. These transformations have the property (not reflected by our example) that they tend to raise very rapidly the dimension of the space of the data. In a data space of dimension n, the separation by hyperplanes has $d_{VC} = n+1$, hence these transformations, leading to very high d_{VC} values could be seen as more harmful than useful. This is why Vapnik introduces the effective Vapnik-Chervonenkis dimension, d_{VCeff}, which is such that, for a hyperplane in a space of dimension n,

$$d_{VCeff} \leq \min (n, (R/\rho)^2) + 1.$$

It follows that if the data are non linearly separable, n grows very rapidly due to the coordinate transformations performed, and if the "clouds" of data points are very near to each other (*i.e.*, ρ is small), then the learning problem is not actually solvable. In opposition, even in the case of very high dimension representation spaces (in some cases, their dimension can be infinite), as long as the data are clearly separated in this space, the term $(R/\rho)^2$ insures that learning can take place with a good rate of convergence. In other words, this approach defines the learning problem as a function of the data, and this explains the practical success of the SVM. To speak crudely, either the problem is relatively easy, n is quite small, and the generalization power is measured by n + 1, or the problem is relatively hard, n grows very rapidly, and the generalization power is measured by $(R/\rho)^2 + 1$.

Generality is defined in a very different way within the symbolic settings. In fact, it uses definitions that can hardly lead to a distance measure. For instance, replacing a constant by a variable is a generalization, increasing the domain of a variable is a generalization, and the inverse operation are specializations. The inductive steps do not take place by modifying the orientation and distance to origin of a hyperplane, they take place by increasing or decreasing the degree of generality of a formula. This is the famous paradigm of "learning as search" developed by Tom Mitchell (1983) under the name of *version spaces*. Most symbolic learning systems use a version of this paradigm. The data are split into what is called positive examples (that illustrate the class to be learned) and negative examples that do not belong to the class to be learned. The positive examples are looked upon as the most particular expression, and learning proceeds by generating formulas of which the examples are instances, and of which the negative examples are not instances. This paradigm received very little critique until recently when it was shown (Giordana and Saitta, 2001) that this version space is very large, and there is only a little zone of which useful inductions can be drawn. This result tells us one must be weary of the place where the induction starts in the version space, but it does not destroy Mitchell's principle of "learning as a search."

As a first conclusion, statistical induction learns coordinates of separating hyperplanes in a transformation of the original data descriptions, and symbolic induction learns formulas expressed in terms of the original data descriptions. Both

use accuracy as a leading path to successful inductions, and both validate their induction by their accuracy on a test set. Symbolic learning introduces also the notion of complexity of the induced formula, and its comprehensibility for a field expert. My main point here is as follows: if symbolic learning attempts to compete with the statistical learning on the terrain of accuracy, the SVM approach will always be better. Inversely, understandability is the strong point of symbolic learning, and the field should pay more attention in judging the results in accordance with this criterion.

As a second conclusion, let us stress out that the amount of generalization performed by a symbolic system is always somewhat arbitrary to evaluate numerically since it depends on the grain of each allowed generalization step. It should be nevertheless a part of the evaluation process. Of two systems that learn with the same precision, the most general one is the most promising for application to new unknown data, and to "explain" them, the less general one explains the actual data with more details. Both have their merit from the explanatory point of view, but it should be known if the induction algorithm provides the one or the other.

As a third conclusion, it is relatively easy to analyze the cases where statistical learning is able to show some understandability. The result of a SVM is a hyperplane that discriminates the data with the largest possible ρ, in order to increase the speed of convergence of the induction process, as shown by the Novikoff-Vapnik formula given above, since it decreases the value of d_{VCeff}. It may happen that this hyperplane is perpendicular to one of the coordinate axes, and if this axis describes a comprehensible variation, then it can be understood by the field user. But this situation describes a case where symbolic learning will work very well, in other words, a case where all the refinements of numerical learning are not needed. When these refinements are needed, then the separating hyperplanes are not perpendicular to any axis, and their "meaning" is the one of a linear combination of many variables that usually highly obscure the field expert who presented his/her data as defined by the values of the non transformed, non combined variables.

Let us now consider some problems linked to the generation of rules from data, and especially from data generated by a text analysis.

2.2. HEMPEL'S PARADOX AND THE THEORY OF CONFIRMATION

Hempel underlines the existence of the contraposition associated to each theorem as shown below:

$$A \Rightarrow B \sim \neg A \vee B \sim \neg A \vee \neg \neg B \sim \neg \neg B \vee \neg A \sim \neg B \Rightarrow \neg A$$

The existence of a contraposition proves that any theorem is confirmed by the

7

simultaneous observation of both premise and conclusion, as well as by the simultaneous observation of both negation of premise and negation of conclusion. For example:

$\forall$ x (crow (x) $\Rightarrow$ black(x))	$\sim$	$\forall$ x ($\neg$ black(x) $\Rightarrow$ $\neg$ crow (x))
confirmed by the observation of crow(A), black(A)		confirmed by the observation of $\neg$crow(B), $\neg$black(B) example: white(B), shoe(B)

Induction generates hypotheses that have to be confirmed by observation of the reality, but Hempel's paradox tells us that so many things confirm any crazy hypothesis that confirmation by counting of instances is simply impossible, thus automatization of induction (which has to rely on some sort of counting) is absurd.

In order to show how automatic induction has been nevertheless possible, it is necessary to consider that induction contains complex chains of reasoning steps (Kodratoff & Bisson, 1992, Kodratoff, 1994), and an analysis of this complexity leads to a better understanding of the conditions into which safe confirmation can be performed. Let me now summarize this argument. The first remark is that a specific semantic (or meaning) is associated to an implication, and Hempel's paradox holds in a different way depending on the semantics. If the implication is relative to the description of the properties of an object, such as the black crow above, then there is little to discuss: the "*descriptive theorem*" $\forall$x (crow (x) $\Rightarrow$ black(x)) is indeed a theorem from the deductive point of view (the contraposition of such a theorem is valid: for instance, if anything is not black, obviously it is not a crow) but it is not a real theorem from the inductive point of view since it is not confirmed by instances of its contraposition. This is why, when dealing with induction, we have to make the difference between the descriptive theorems, and what we call *causal theorems*, where the implication carries a causal meaning. Due to the fact that Science has been concerned until now with the automation of deduction, the difference between descriptive and causal theorems is not acknowledged.

2.3. IMPLICATIONS THAT CARRY THE MEANING OF CAUSALITY

Consider the implications that represent a causal relationship, such as

$\forall$x (smokes(x) $\Rightarrow$ cancer(x)) with probability p. There is no point in calling on Hempel's paradox here, since indeed, observing ($\neg$smokes(A)) & ($\neg$cancer(A)) confirms also this theorem, as it should. It must be however noticed that spurious causes can introduce again a paradox. For instance the theorem:

$\forall x$ (smokes(x) & French(x) $\Rightarrow$ cancer(x)) is absurdly confirmed by observing ($\neg$smokes(A) $\vee$ $\neg$French(A)) & ($\neg$cancer (A)) meaning that, say, a German who has no cancer confirms this theorem. A simple analysis of the correlations (see the definition of a spurious dependency, in Section 5.5, below)

8

will show easily that nationality has nothing to do with the link between smoking and cancer. A striking example of this problem has been given recently by the so-called "French paradox" stating that Frenchmen had a higher cholesterol count than Americans and they would nevertheless die less of heart attack. It was called very aptly a "paradox" because the fact of being French has obviously no causal role, and the real cause has been found in some typical French habits.

Another disputable argument is that the conjunction of causes is dangerous, for instance:

$\forall x$ (smokes(x) & drinks-alcohol(x) $\Rightarrow$ cancer(x)) is confirmed by (($\neg$smokes(A) $\vee$ $\neg$drinks-alcohol (A)) & $\neg$cancer (A)) which is confirmed by any person who does not drink and has no cancer. The counter-argument here is that the "medical *and*" is not really a logical conjunct. Actually, here, drinking increases the unhealthy effect of smoking and we have to confirm two independent theorems, one stating that

$\forall x$ (smokes(x) $\Rightarrow$ cancer(x)), and the other one that

$\forall x$ (drinks-alcohol(x) $\Rightarrow$ aggravates (cancer, cause_is_smoking, x)).

More generally, the paradox originates here from a false knowledge representation, and it will indeed lead to absurdities,[1] but this is not especially linked to the theory of confirmation. In other words, the implication using a logical *and* is false, and it is confirmed by almost anything, as it should be. Inversely, when two conditions are simultaneously necessary to cause a third one, say, as in stress & age>45 $\Rightarrow$ heart condition (where we suppose that both stress and aging are causal), then the disjunction in the contraposition is no longer paradoxical.

In short, in the case of causal implications, absurd confirmations are avoided by a careful examination of the meaning of the implication. Simple counting might be dangerous, but there are enough statistical methods to avoid easily the trap of Hempel's paradox.

2.4. PRACTICAL CONSEQUENCES

[1] There is another famous argument against induction, namely that observing that 1. drinking vodka with water makes you drunk, 2. drinking gin with water makes you drunk, leads to the generalization that drinking water makes you drunk. Some claim: "This generalization is obviously absurd, hence generalization is absurd." I leave to the reader to show that, again, this argument is due to a bad knowledge representation where the word *with* is represented by a logical *and*- which is indeed absurd!.

All that shows how Science has been able to build theories using confirmation, in spite of Hempel's paradox. Unfortunately it also shows how unreliable some confirmation measurements that are automatically performed might be.

Suppose you are looking for implications A ⇒ B (*i.e.*, associations) coming

9

from a taxonomy of generality, *i.e.*, with inherited property semantics, such as, for example, dog ⇒ canine. Then only the couples (A,B) confirm this hypothesis, and the couples (A,¬B) disconfirm it. Thus, the probability of confirmation for A ⇒ B should be estimated by counting the number of corresponding items (let # be the counting function, and N be the total number of items) and approximated by the value: (# (A,B) – # (A,¬B)) / N.

Inversely, suppose you are looking for implications A ⇒ B with a causal semantics. They are confirmed by all couples (A,B) and (¬A,¬B), and they are disconfirmed by all couples (A,¬B). The probability of the confirmation of A ⇒ B should then be approximated by (# (A,B) + # (¬A,¬B) – 2*(# (A,¬B)) / N. These remarks will explain the changes I propose to the classical definitions of coverage and confidence in section 5.3.2. and 5.3.3.

3. INDUCTIVE TEXT MINING

The discovery of interesting rules is an especially difficult case for Inductive Text Mining (ITM). This is due to the fact that texts tend to be speak of a specific topic, and the number of topics they do not speak of is almost infinite. Therefore, the amount of patterns that express relationships among topics that are NOT dealt with in the texts is normally overwhelming (with a very high cover and a very high precision). This results in a flood of patterns. One way to save and extract the interesting (and useful) patterns is by using a measure of interest. Another difficulty comes from the fact that natural language uses thousands of different terms even in relatively rigid settings. At present pattern discovery systems are unable to deal with this level of variety. One solution is to define classes of terms, and then to look for patterns among these less numerous classes of terms. It is even possible that these classes have meaning for the expert reading these texts. These classes are then called concepts. It is obviously interesting to discover unknown patterns among concepts that are meaningful to an expert. It follows that the building of ontologies makes ITM possible and makes its results more interesting – when the defined classes are indeed interesting for someone.

The Inductive Text Mining (ITM) approach we propose in this paper puts the Data Mining (DM) phase after a learning phase.

The learning phase is as follows. Gather a large quantity of the kind of texts you want to mine. Perform a syntactic analysis on these texts. Built taxonomies of concepts by an interaction between a field expert and the results of the syntactic analysis. In each of the texts, note the apparition of each concept, and build up a table that gives the probability of appearance of each concept in each text. The probability of appearance of a concept is approximated by the number of times the concept appears in the text, divided by the total number of concepts appearing in

10

the text. The text is now reduced to one record in a table, the fields of which are labeled by the concepts, and the items of which are the probabilities of appearance of each concept in this text (= in this record).

Clearly, the only new step during the learning phase is the building of a taxonomy of concepts. The second section of this paper explains how we perform this step. The DM then takes place in a straightforward way. Apply DM techniques to the table obtained during the learning phase. The kind of DM we consider here is the detection of patterns in the data, most often called association detection. It looks for the dependencies among values of the fields. In Section 5, we study various methods for detecting such dependencies, including the building of a Bayesian network from data, and new dependency measures.

4. BUILDING ONTOLOGIES FROM TEXTS

In our case, the ontology is reduced to a special subform: a graph, or a taxonomy, of generality relations among concepts. The generality relationships are most often in the form of a graph because of the polysemy of the words. The word “problem”, for instance, can be an instance of a concept referring to ‘activity’ (technical problems to solve), an instance of a concept referring to ‘relations’ (problems with other people), or an instance of a concept referring to the ’individual’ (personal problems).

Faure and Nédellec (1999) have developed a system, ASIUM, that is able to build such generality graphs. The text is submitted to a syntactic analysis, and all nouns found in the same syntactic position (say, all the nouns that are direct object of verb ‘to move’, or all nouns associated with adjective ‘movable’) are put into the same base classes. ASIUM is provided with a distance measure between classes and can offer its user the opportunity to merge the nearest base classes into a concept class. When the user accepts a new concept, then the base classes are validated as significant, and the new concept is created. This process can be iterated several times, thus creating concepts of higher and higher levels of generality. The process is interactive, and the user can accept or reject new

proposed classes, and he or she can also reject a chosen noun as not belonging to the class. ASIUM is able to point out useful classes that actually exist in the texts, which is especially useful as the field expert is not always able to provide them all.

To go further in this process, we are presently producing a system, called Rowan, that generates taxonomies, in place of generality graphs. However, the cost for this is quite high due to the amount of work involved. Rowan collects each syntactic position into which a noun belongs to a given concept. The number of such positions is very large as soon as the style of the texts is complex enough. Rowan is still under development. It has been until now applied to only one corpus of 6 megabytes of text. These texts belong to the company Performanse,

11

specialized in human resources. This company uses batteries of psychological tests, but their specificity lies in their concern for showing the results of the test to the testees, under the form of a text that has been especially tailored so as to be seen as neutral by the testee. Discussion of this text with the psychologist then takes place, and the final result is a new text upon which testee and psychologist agree.

To better understand the complexity of the work involved in Rowan, consider the following example that gives some results relative to the word ‘environment’. The word ‘environment’ is recognized in a total of 2367 different syntactic situations. It is used as an agent of a verb 35 times, as the (direct) object of a verb 1084 times, 190 times as the subject of a verb, and 1058 times in a noun-noun or a noun-adjective link. There are 53 noun-adjective links, among them it is used 9 times in ‘understanding environment’, 6 times in ‘able environment’, etc., until it is used 30 times in a link appearing only once in the texts, such as ‘relaxed environment’. It appears thus in 1058 – 53 = 1005 noun-noun links. Among those, 170 are used only once. Verifying that all these links refer to a concept is a very tedious task.

As the word environment appears in the texts, it characterizes no less than 8 different concepts. This is specific to the style of the writer, and it does not intend to give general laws about the way the word environment should be used by someone else. It is a leaf of

- concept *belonging* when used as ‘supporting environment’;
- concept *communication* when used as ‘attitude of the environment’, ‘to react in favor of the environment’, etc.;
- concept *expansion* when used as ‘communication with the environment’, etc.;
- concept *hierarchy* when used as ‘authority in the environment’, dynamization of the environment’, etc.;
- concept *independence* when used as ‘personal environment’;
- concept *influence* when used as ’to dynamize the environment’ etc.;

– concept *relational* when used as ‘action of the environment’, ‘cohesion with the environment’, etc.;
– concept *stress* when used as ‘conflict with the environment’, ‘crisis in the environment’, ’tension in the environment’, etc.

The influence of the author’s style is particularly clear here when he uses ‘dynamization’ in the context of hierarchy, and ‘to dynamize’ in the context of influence (within a hierarchical relation or not). The care taken by Performanse’s psychologist to write sentences addressing individual situations reflects in the following figures: the total number of syntactic relationships in the text is 90630, and the number of syntactic relationships appearing only once in the corpus is 50793. In other words, more than half of the sentences use a unique grammatical form.

12

The final taxonomy contains the thousands of relations characterizing the 12 concepts of interest to Performanse. As a matter of fact, and in its present state, Rowan is nothing but a friendly interface helping the user to check these thousands of relations against the actual sentences given in the texts. This large amount of work is the price to pay to be able to extract information from texts: there is no cheap way to handle natural language.

These results open the way to more research, as follows.

We thus succeeded in building a taxonomy of concepts, and we will be able, on this specific corpus, to find the occurrences of concepts in the texts. This is an obviously interesting goal, but we do not intend to stay at this stage. Each taxonomy building, for each corpus, will ask for the same amount of effort. It is therefore important to study what uses these existing taxonomies can have. There is a large amount of work already done in the use of ontologies. Nevertheless, usual ontologies do not mix up terms and syntactical positions, and a complete re-examination of the use of our taxonomies is to be undertaken. Another obvious task to perform is building subcategorization frames out of the taxonomy. Our frames will not hinge around the verbs only. For instance, a set of adjectives can hinge around a noun, the combination noun-{set-of-adjectives} all belonging to the same concept. The next step is the building of semantic schemes by properly generalizing the “hinging” sets among themselves. For instance, from the frames noun1-{set-of-adjectives1} and noun2-{set-of-adjectives2}, it is possible to find the scheme noun1-{set-of-adjectives1-2}-noun2, where {set-of-adjectives1-2} is a generalization of the two sets of adjectives. Once this learning has taken place, it is possible to work the other way round on a new corpus: check if the frames and schemes found are still valid. Old schemes can be accepted as such, rejected, or modified, in any case reducing the amount of work needed to build a new taxonomy for a next corpus.

5. "INTERESTING" ASSOCIATIONS

5.1. INTERESTING = RELEVANT (SOCIAL PROPERTIES) + RARITY (STATISTICAL PROPERTIES)

First, it seems necessary to focus somewhat on the definition of interesting. This word contains so many personal or social connotations that it can hardly be used in a scientific context. We nevertheless said above that 12 concepts were of "interest" for Performanse. The field expert, in this case the Performanse's chief psychologist who conceived the method for generating the texts, usually has a very precise idea of what is relevant or not. The field expert thus provides the relevance, and finding relations among relevant (or non-relevant fields) of the database is the responsibility of this expert.

13

Now consider a property Π of the data, and suppose that this property is relevant. This property obviously takes a mean value on the data, M_Π, and its distribution shows a standard deviation σ_Π. Now the patterns that show the value M_Π for Π are less statistically surprising that the patterns showing a value, say, greater than $M_\Pi + \sigma_\Pi$. This is why I suggest following a simple path: call *interesting* the patterns showing a value of property Π **greater** than $M_\Pi + \boldsymbol{p}*\sigma_\Pi$, where ***p*** is a coefficient indicating the 'size' of the surprise asked from the data to be deemed 'interesting'. Note that several authors have already been using this approach. However, they usually try to find a general value for σ_Π, one that is valid for all sorts of data, which is obviously impossible.

For instance, on the example set known as "mushrooms", a classical database available on line, the most surprising associations (see our definition of 'surprise' below) shows a value, for Π = surprise, equal to $M_\Pi + 4.168 * \sigma_\Pi$.

Inversely, it might be that we are interested in finding the most standard instances of a property that show the most often. In that case, call a *normal* pattern those showing a value of property Π **less** than $M_\Pi + \boldsymbol{p}*\sigma_\Pi$, where ***p*** should be less than 1. An instance of a measure for which normality is desirable will be described below in Section 5.5. It uses concepts describing a Bayesian network.

5.2. A REVIEW OF THE MOST USUAL PROPERTIES Π FOUND IN THE LITERATURE (see (Brin *et al*., 1997; Lavrac *et al*., 1999)

All measures relative to discrete or boolean valued descriptors are summarized by the following diagram.

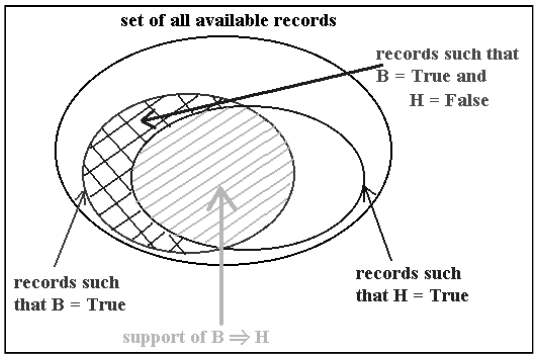

Fig. 1. – Diagram showing P(B,H) and P(B, ¬ H).

Consider that we are trying to "discover" a pattern of the form Body ⇒ Head (also called 'association between two items'). The **support** of [B ⇒ H] is defined as **Supp(B ⇒ H)** = P(B,H). This expresses the probability that B and H will be True together. Notice that when P(B) and P(H) are both very large (*i.e.*, they determine a field containing almost only 1s in the DB), it is somewhat meaningless

to study their relationship: their support is always very large.

The **confidence** in [B ⇒ H] is defined as P(B,H) / P(B) = P(H | B). This expresses the conditional probability that H will be True, knowing B is True.

The **conviction** associated to [B ⇒ H] is defined as P(B) * P(¬H) / P(B,¬H). When B is True and H is False, then [B ⇒ H] is disconfirmed. This is why the larger P(B,¬H) is, the lower the conviction. Suppose that P(H) is tiny (hence P(¬ H) is large), then a large P(B, H) can hardly be due to simple chance, and we have more conviction in favor of [B ⇒ H]. Thus, the larger the support of P(¬ H), the larger the conviction.

The **'interest'** of [B ⇒ H] is defined as P(B,H) / [P(B) * P(H)] = P(H | B) / P(H). The more probable P(B) and P(H) are, the less interesting is their intersection since it is expected to be always large, as we already pointed out in defining the support.

The **dependency-1** (the classical dependency measure) of H upon B is defined as Abs(P(H | B) – P(H)) if P(B) ≠ 0, where Abs is the function absolute value. If B is not absurd, *i.e.*, its probability of occurrence is not zero, which this means that P(H | B) = P(B,H) / P(B) is computable, then the probability of meeting H given B should increase when B ⇒ H. The greater the difference between P(H | B) and P(H) the greater the strength of the dependency. It is interesting to discuss a few properties of dependency in order to understand exactly what it evaluates.

By definition, dependency is small when P(H) is large.

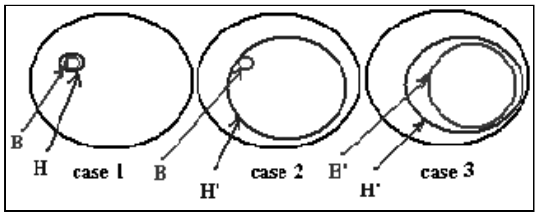

Fig. 2. – Three cases asking for different measures of dependency.

Three cases are particularly interesting. In all three cases, the body is supposed to be almost perfectly included in the head, thus P(H |B) = 1.

In case 1, P(H) is small, thus case 1 is the one of a large dependency.

In case 2, B is "drowned" in H' and is intuitively normal that H' depends much less from B than H does. The dependency, 1 – P(H'), is low, as it should.

In case 3, P(H') and P(B') are both very large and their dependency is more "uninteresting" than small. In this case, it follows that the dependency-1 measure has no intuitive meaning.

It is often said that "dependency is the discrete equivalent to correlation." This statement is to be used carefully since numerical correlation of two variables x and y is symetrical, while dependency is not. In fact, dependency (B, H) =

15

dependency(H, B) * (P(H) / P(B)).

Moreover, dependency is measured between values of variables, not between variables. For instance, weight and size are correlated, but this correlation is very clear for high and low values of these variables. It might well be that the values of the discretized variables, height = medium, size = medium, are totally independent.

The **dependency-2** (also called **novelty**) of H upon B is defined as P(B,H) – (P(B) * P(H)). If B and H are independent, the expected value of P(B,H) is P(B) * P(H), hence the difference between these two values expresses how far from independence are B and H. This is indeed a measure of dependency. Note however that dependency-2 = P(B) * dependency-1. It follows that dependency-2 will favor implications [B ⇒ H] such that P(B) is large, hence less rare than the ones where P(B) can also be small. Dependency-2 thus compensates the defaults of dependency-1 when P(B) is large as in case 3, (and this is exactly the case where large dependencies are not interesting), but will spoil the good behavior of dependency-1 in case 1 and 2.

The **dependency-3** (also called **satisfaction**) of H upon B is defined by (P(¬H) – P(¬H | B)) / P(¬H) which the same as (P(H | B) – P(H)) / (1 - P(H)). Instead of multiplying dependency-1 by P(B) as in dependency-2, we now divide by (1 – P(H)). Thus, the so-called satisfaction is again a dependency. It behaves quite well in cases 2 and 3, because it takes into account the fact that P(H) is large (and we divide by the small quantity 1 – P(H)). Inversely, in case 1, dependency-3 will be very small, even when B and H are indeed very dependent of each other. In other words, dependency-3 favors the implications such that P(H) is large.

The interest of detecting implications with large support follows from the fact that if an implication has too little support, it might very well happen that it is then confirmed by a very small number of instances, that come from noise only. Besides, this measure finds a fast generalization to implications between several items (of the form

[A & B & … ⇒ A' & B' & …]) which leads to the well-known APRIORI algorithm (Agrawal *et al*., 1993).

A large amount of research has been done around this algorithm, based on the assumption that only associations with a support larger than a fixed value are interesting.

This approach can be criticized from two directions. Firstly, these systems can detect spurious associations that are pure errors due to this approach. Secondly, associations with large support can be not very surprising, since they are relative to a large part of the population. For instance, on the example set "mushrooms", already cited, the most surprising association (see our definition of surprise below) has a support of 0.000985, and it is thus impossible to detect with a support-based technique. Inversely, a high value of the surprise does not automatically mean low

16

support. For instance, a rule found by all systems, 'odor_foul → poisonous,' shows a surprise of M + 1.941*σ, and a support of 0.266. Among the most surprising rules of surprise value = M +3.698*σ, there is even a rule of support 0.694.

Notice that this criticism is invalid for the use of APRIORI-like algorithms in the detection of sequences. In the representation promoted by Agrawal (Agrawal and Sikrant, 1995; Sikrant and Agrawal, 1996), it is obvious that only the instances that are often repeated can be part of a sequence, hence their support must be large, otherwise they are isolated facts, not part of a sequence.

5.3. DESCRIPTION OF SOME LESS USUAL PROPERTIES Π

5.3.1. Statistical surprise

A less classical property Πis what we call 'surprise'. The **statistical surprise** of [B ⇒ H] is defined as (P(B, H) – P(B,¬H)) / P(H). As in the case of conviction, it is noticed that the larger P(B, H), the more [B ⇒ H] is confirmed, while the larger P(B,¬H), the more [B ⇒ H] is disconfirmed. The exact value of the confirmation is thus given by (P(B, H) – P(B,¬H). Further, the larger P(H), the more probable it can contain small subsets that will imply it trivially. It is therefore more surprising to find an implication with a large confirmation when P(H) is small.

Many oppose to the use of P(B, H) - P(B,¬H) because P(B, H) + P(B,¬H) = P(B), hence P(B, H) – P(B,¬H) = 2 * P(B, H) – P(B). Clearly, this last expression has no interesting semantics, but each formula above could be made meaningless by a similar manipulation.

Making the difference between the confirmation and the disconfirmation can give the additional advantage that the noise distribution on P(B, H) and on P(B,¬H) should be very similar, hence their difference should be more stable

against the noise than the individual values. A large drawback of this measure is that it generalizes very poorly to implications among several items. This is why we need to consider the building of Bayesian networks, giving an exact form of the possible combinations among multiple items.

5.3.2. Including disconfirmation

All measurements can include the idea that disconfirmation decreases the value of the measurement, as we did in defining the statistical surprise. It is quite easy to define an effective support by P(B,H) – P(B,¬H). However, extending this notion to other measures is difficult since we do not know how to treat the extra term P(B,¬H). This is why we rather chose to introduce a coefficient ***k*** defining the

17

amount of disconfirmation we can stand without changing the definitions.

Hence, the **effective support** is defined by: **SuppEff(B ⇒ H)** = IF P(B,H) < ***k****P(B, ¬H) THEN 0 ELSE P(B,H). The (confirming) support has to be at least ***k*** times larger than (disconfirming) P(B, ¬H) to be looked upon as different from 0. Notice that this definition does not keep the "good" properties that all combinations of n items have the same support (for instance: SuppEff(B, C ⇒ H) is possibly different from SuppEff(B ⇒ H, C)). Nevertheless, is it still true that all combinations of n items have a support less or equal to a combination of n-1 items. For instance, SuppEff(B ⇒ H, C) is ≤ SuppEff(B ⇒ H), since P(B, ¬C) can only decrease the effective support of B ⇒ H, C. This can be trivially proved by induction on n.

The **effective confidence** is defined as **ConfEff(B;H)** = IF SuppEff(B ⇒ ⇒H) = 0 THEN 0 ELSE P(H|B).

The **effective dependance-1** is defined as **DepEff(B;H)** = IF SuppEff(B ⇒ ⇒H) = 0 THEN 0 ELSE abs(P(H|B) – P(H).

5.3.3. Including contraposition

We include now the values of the contraposition. We explained in section 1 that it amounts to giving the semantics of causality to the confirmation measures, this is why we shall say these measures are causal.

The **causal effective support** is defined by IF P(B,H) + P(¬B, ¬H) < < 2***k****P(B, ¬H) THEN 0 ELSE P(B,H) + P(¬B, ¬H).

The **causal effective confidence** is defined by P(B,H)/P(B) + P(¬B, ¬H)/P(¬H) if the causal effective support is not 0.

The **causal effective dependance** is defined by abs(P(H|B) – P(H)) + + abs(P(¬B|¬H) – P(¬B)) if the causal effective support is not 0.

5.4. ANALYSIS OF SOME RESULTS RELATIVE TO THE 'MUSHROOM' DATABASE

It should be clear that our first condition for obtaining interesting relations from this DB is not fulfilled: except mycologists nobody has any interest in the concepts contained in this DB. Thus, this section aims simply at exemplifying the properties of the various measures given in Section 5.3.

The measurements were done with $\boldsymbol{k} = 3$, *i.e.*, a rule such that its confirmation is not higher than three times its disconfirmation will have its support and other values put to 0, as explained in Section 5.2. There are 195 rules of the form $B \Rightarrow H$, and such that the statistical surprise is greater than mean + standard deviation. We ordered them by value of their statistical surprise. Most of them show a value near 1 for most measures except for those that are proportional to P(B) or P(H).

[18]

Tables 1 and 2 present some of the rules we obtained:

19

Table 2

Rule	Rule #
veil-color = yellow ⇒ stalk-color-above-ring = yellow	1
ring-number = one ⇒ veil-type = partial	2
odor = none ⇒ edible	3
gill-attachment = free ⇒ gill-spacing = close	4
edible ⇒ odor = none	5
odor = foul ⇒ spore-print-color = chocolate	6
odor = foul ⇒ poisonous	7
gill-size = broad ⇒ edible	8
poisonous ⇒ odor = foul	9

The most surprising rule is a typical "nugget" of knowledge, meaning that it expresses a knowledge relative to a very small part of the data, but one which is never disconfirmed. The rule, *IF veil-color = yellow ⇒stalk-color-above-ring = yellow*, expresses a property that seems to be very specific of yellow veils, since there are many rules relative to white veils, none relating its colour to the one of the stalk. The value of P(B) is around 0.001, thus the so-called novelty is of the order of 0.001 since dependency-1 is near 1. Its support is also around 0.001, which means that only 8 samples of the 8000 contained in the DB show this property. This can be seen as tiny, but if we think in terms of large DB, say of

8 millions records, then it defines a population of 8 thousand records showing a quasi deterministic property. A very predictable population of 8000 individuals can be very important from a financial point of view.

Let us now consider the rules linking edibility and odor. The rules *odor = none* $\Rightarrow$ *edible*, and *edible* $\Rightarrow$ *odor = none* are very near to each other, with a quite large support of 0.419. The first one seems to be somewhat less causal (effective causal dependency-1 = 0.708) that the second one (effective causal dependency-1= = 0.863). Inversely, the rules *odor = foul* $\Rightarrow$ *poisonous*, and *poisonous* $\Rightarrow$ *odor = foul* are very different. The first one shows a surprise of 1.941, and an effective causal dependency-1 of 0.547. The second one shows a very little surprise of 0,129 and an effective causal dependency-1 of 0.000. This means that odor = foul is quite causal for the property of being poisonous. We do know that this causality is fortuitous but, still, it makes sense that foul odour is more causal for edibility than absence of odour.

The rule *ring-number = one* $\Rightarrow$ *veil-type = partial* illustrates the default of dependency-1 we have underlined. The value of P(H) is high (= 1.0), thus even with a P(H | B). of 1, dependency-1 is zero. In this case, that P(B)/ P(H) is equal to 0.922 indicates that B is not a relatively small subset of H, and the dependency

between the two is actually high. Neverthless, both are true for almost all mushrooms, and this dependency is trivial. Notice alos that the causal effective confidence is not computable, since P(¬H) = 0.

The rule *gill-attachment = free* $\Rightarrow$ *gill-spacing = close* shows a very high P(B) of 0.974, that is, almost all mushrooms have a free gill attachment. As a consequence, P(B, ¬H) is almost the same as P(¬H), and the limit value of 2****k**** P(B, ¬H) is often reached, leading to a zero causal support, as happens in this rule where P(¬H) is around 0.16. Since we deal with probabilistic causalities, there is nothing wrong with the cause being more probable than its effect. There no real good reason for eliminating these causal effects, except that they are indeed very hard to detect from the data only: they are typical of the causes that need experiences to be detected. For instance, the probability of presence of oxygen, nitrogen, etc. in the atmosphere is always, one. Only experiments can tell what causes what, by creating data where, exactly my point here, these probabilities become low.

Rule *odor = foul* $\Rightarrow$ *spore-print-color = chocolate* illustrates the necessity to deal with effective supports. As you can see, all it classical indicators are quite high, but P(B, ¬H) happens to be quite high, and the confirmation is not more than three times the disconfirmation, hence the effective supports drops down to zero. Notice also that adding the contraposition changes completely these figures, hinting at the possibility of a causal relationship.

Rule *gill-size = broad* $\Rightarrow$ *edible* illustrates the possibility that including the disconfirmation values eliminates both causal and non causal dependencies.

5.5. WHY BUILD BAYESIAN NETWORKS?

Bayesian networks deal with a phenomenon that is not considered in the systems performing the detection of associations, the so-called spurious dependencies.

Consider the case where the computations lead to accepting three implications as B $\Rightarrow$ H, B $\Rightarrow$ C, and C $\Rightarrow$ H. Since B implies both H and C, H will be true when C is true, even if C is not causal to H, that is, even if the implication C $\Rightarrow$ H does not hold in reality. If C is really causal for H, then H must be more frequent when B and C hold than when B alone holds, that is, P(H | B,C) must be higher than P(H | B). Hence the classical definition of a spurious dependency between C and H is: P(H | B,C) = P(H | B). In the vocabulary of Bayesian networks, it is said that, when P(H | B,C) = P(H | B) holds, H is independent of variable C for B given. In other words, H and C are conditionally independent, see for instance Jensen (1998).

A Bayesian network is built in order to take into account all kinds of conditional dependency. For instance, in a Bayesian network such that P(H | B,C) = P(H | B), the 'arrows' B$\rightarrow$ C and B $\rightarrow$ H will be indicated in accordance with the

21

implications B $\Rightarrow$ C, and B $\Rightarrow$ H, but the implication C $\Rightarrow$ H, the spurious dependency, will not be marked by an arrow. Inversely, spuriousness is not checked by the systems that perform association detection.

We are now able to provide an instance of a measure for which normality (as defined in Section 5.1 is desirable. Suppose we are given a Bayesian network and a set of data supposed to fit this network. Suppose also it shows the 'arrows' B$\rightarrow$ C and B $\rightarrow$ H, but no link between C and H. It is very important to be able to measure how much the absence of link between C and H is justified since this is the very reason of superiority of Bayesian networks over association detection. We must thus evaluate how much P(H | B,C) = P(H | B) holds. Instead of defining an arbitrary value under which the property (cf. 5.1) Π = P(H | B,C) – P(H | B) is considered to be zero, we suggest to follow the procedure recommended in 5.1. Obviously, there is a coefficient here also, but this coefficient makes statistical sense since it represents the probability that a new item will fall within the defined range. In other words, it varies with the way the experimental data are distributed.

Recently, Munteanu (Jouffe and Munteanu, 2000; Munteanu and Cau, 2000; see references therein to find the classical results in this field) has been showing that it is possible to directly build equivalence classes of Bayesian networks in a reasonable time. This opens the way for an alternate method for detecting

associations among multiple items: only those compatible with the Bayesian network supported by the data are evaluated. The main limitation of this approach is that it is strongly sensitive to the number of different fields it can deal with. A maximum of some 500 fields (or descriptors) seems already asking for a very long computing time. Another feature of this approach is that it demands a very large number of records, but this is easily met in most modern applications.

6. CONCLUSION

This paper proposes two refinements to the classical approaches to ITM. On the one hand, we claim that the problem of polysemy in ontology building must not be solved by using generality graphs to describe the concepts, as the 'grain of the theory' is then too coarse. We have to refine the theory in order to transform the graphs into taxonomies, and pay the price of the theory's finer granularity which is a large amount of work, as done when using our software Rowan. It seems that the association of a noun with its syntactical context is enough to build taxonomies. This is, however, our working hypothesis and it has to be confirmed on more data. The next data we are going to analyze with Rowan are a set of introductions of KDD papers, and the Grimm tales (in English). These two sets should cover a quite large spectrum of possibilities of naturally produced texts. We will then be able to evaluate the amount of noise in the definition of the concepts that is still left after this work is done.

22

On the other hand, the present day techniques for association detection seem largely inadequate since they detect relationships that have a support larger than a given threshold. Experience has shown that association with a tiny support can be very 'interesting' in some sense. The global strategy we recommend is the following: use a 'Bayesian networks synthesis from data' technique in order to obtain an overall idea of the network of relationships characterizing the texts. Then, evaluate the interest of each link with your favorite interestingness function. Our surprise function seems to show the important feature that it resists well to noise. Our preliminary measurements (complete results will be published elsewhere) show that the other measures tend to be sensitive to very low noise levels of the order of 1 to 2% of noise. Since there is absolutely no hope for detecting the presence of a concept in a text with such a low noise, this emphasizes even more the need for new techniques of association detection, already discussed for other reasons earlier in this paper.

REFERENCES

Agrawal R., R. Srikant R., *Mining Sequential Patterns*, in Proceedings of the 11th International Conference on Data Engineering (ICDE'95), Tapei, Taiwan, March 1995.

Agrawal R., Imielinski T., Swami A., *Mining Association Rules between Sets of Items in Large Database*, in Proc. SIGMOD'93, pp. 207–216, Washington DC, USA, May 1993.

Brin S., Motwani R., Ullman J. D., Tsur S., *Dynamic Itemset Counting and Implication Rules for Market Basket Data*, in Proc. SIGMOD'97, pp. 255–264, Tucson, Arizona, May 1997.

Christianini N., Shawe-Taylor J., *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.

Faure D., Nédellec C., *Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: the System ASIUM*, in Fenser & Studer (Eds.), 11th European Workshop EKAW 99, 329–334, Springer-Verlag, 1999.

Giordana A., Lorenza Saitta L., *Phase Transitions in Relational Learning*, Machine Learning Journal (to appear).

Jensen F. V., *An Introduction to Bayesian Networks*, UCL Press, 1998.

Jouffe L., Munteanu P., *Smart-Greedy+*: *Apprentissage hybride de réseaux bayésiens*, Colloque francophone sur l'apprentissage (CAP), St. Etienne, juin 2000.

Lavrac N., Flach P., and Zupan B., *Rule Evaluation Measures: A Unifying View*, in Ninth International Workshop on Inductive Logic Programming (ILP'99), Vol. 1634 of Lecture Notes in Artificial Intelligence, pp. 174–185. Springer-Verlag, June 1999.

Mitchell T., *Learning and Problem Solving*, Proc. IJCAI'83, pp. 1139–1151, 1983.

Munteanu P., Cau D., *Efficient Learning of Equivalence Classes of Bayesian Networks*, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), pp. 96–105, Lyon, Sept. 2000.

Srikant R., Agrawal R., *Mining Sequential Patterns: Generalizations and Improvements*, Proc. of the Fifth Int'l Conference on Extending Database Technology (EDBT), Avignon, France, March 1996.

Vapnik V., *The Nature of Statistical learning Theory*, Springer Verlag, 1995

Wolpert D., *The lack of a priori distinctions between learning algorithms*, *Neural Computation*, 8(7):1341–1390, 1996.