Prototyping Structural Descriptions: An Inductive Learning Approach

L.P. Cordella, P. Foggia, R. Genna, M. Vento[1]

Dipartimento di Informatica e Sistemistica
Via Claudio 21, I-80125 Napoli, Italy
E-mail: {cordel, foggiapa, vento}@unina.it
WWW Home Page: http://amalfi.dis.unina.it

Abstract. In structural pattern recognition, a common problem is the difficulty of constructing classification models or rules from a set of examples. This paper illustrates a method for generating class descriptions from a training set by using inductive logic programming, and in particular an inductive learning algorithm based on the FOIL system. The goal of the method is to find descriptions which are general, i.e. are successfully applicable to recognize objects different from the ones in the training set, while preserving their discrimination ability. The application of the method to a difficult real-world problem (handprinted character recognition) is presented, proposing a hierarchical description and classification scheme in order to reduce the complexity of the task to a level which can be profitably handled by this kind of learning methodologies.

Keywords. Structural Description, Prototyping, Inductive Learning, FOIL.

1 Introduction

Structural pattern recognition methods [1] are typically based on the decomposition of an initial representation of the objects of interest into simple parts, so that a shape can then be described in terms of these parts, their attributes and of more or less complex relations among them. It is worth noting that the obtained descriptions have not fixed length and frequently it is not possible to establish an *a priori* order for arranging the extracted primitives in the description; thus the objects are often represented by means of complex structures such as the *Attributed Relational Graphs* (ARG) [2].

A common problem with this kind of representation is the difficulty of constructing, from a suitably chosen collection of examples, the models or the rules that are needed to perform the classification task. In fact, the impressive collection of

[1] Contact author.

effective and theoretically sound learning methodologies that are available when the patterns are represented by means of vectors, like the Statistical Learning theory or the Artificial Neural Networks, cannot be applied to the more complex structures which encode the structural descriptions.

Several authors [3] overcome this drawback assuming that an expert of the application domain can define by hand the prototypes of the classes to be recognized, exploiting the fact that structural descriptions are easily understandable and manageable by humans. This approach, however, could be not feasible if the patterns of a same class show a large variability: in this case the manually defined prototypes may not be able to capture the patterns that differ significantly from the ideal model reflected in the prototype construction, or may at least require a considerable effort in the classification system to absorb these differences. Moreover, in some domains might not be available sufficient expertise to determine which prototypes are to be considered so representative of a class as to deal with all of its possible variations.

The approach we have followed in this paper is to consider the determination of the class prototypes as a symbolic machine learning problem [4]: given a suitably chosen training set, the goal of the system is to devise, by means of an inductive process, a description of each class which is more general than the bare enumeration of the training examples, in the sense that it also covers instances of the class which are not present in the training set, but still preserves the ability of discriminating the objects belonging to other classes. Furthermore, these descriptions must be explicit and easily interpretable by humans (in contrast, for instance, with the ones produced by neural networks), allowing an expert to validate or to improve them, or to understand what has gone wrong in case of errors. First-order logic predicates constitute a powerful representation means for this kind of knowledge, since they are expressive enough to encode both structural descriptions and complex classification rules, and can be directly employed (under some restrictions) to build a classification system by means of a logic programming language such as Prolog. For this reason the learning task has been performed using an Inductive Logic Programming method [5], based on the FOIL algorithm [6], which produces for each class a classification rule expressed as a Prolog program.

Literature about our application domain, which is that of hand-printed character recognition, do not report many attempts to discover prototypes by using ILP approaches; this is due to high variability of the shapes belonging to a same class, thus large training sets are required to encompass all the possible variants of a class, and very detailed descriptions are needed to discriminate between similar classes. This fact puts a heavy burden on the learning system, since symbolic learning algorithms are better suited to work with small and not very noisy training sets. This problem can be, at least partially, overcome by employing a hierarchical description and classification scheme [7]: in a first stage each sample is assigned to a pool-class, which may contain one or more actual classes, using a less detailed description; then in a successive step the actual class is determined, using more specific features, possibly dependent on the selected pool-class. In this way at each level the learning algorithm has to deal with a reduced number of classes and of features, making the derivation of the classification rules a simpler task. In this paper we will focus our attention on the first level of the system, presenting a description method which is tailored to partitioning the input samples into pool-classes, neglecting some features that would be important to recognize the actual class, but that at the first level reveal to be too specialized, discriminating among samples that should instead belong to the same pool-class.

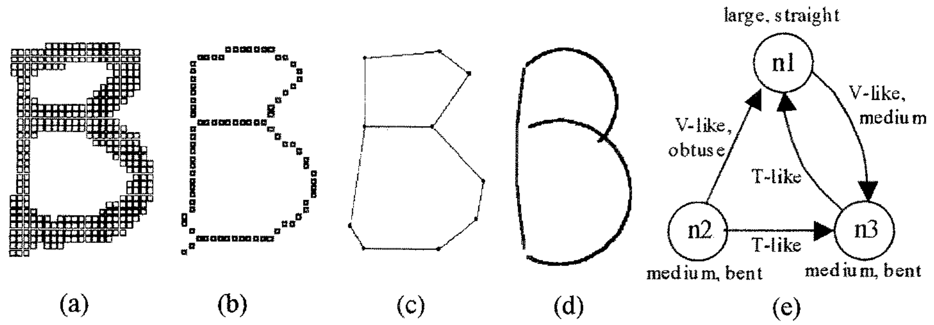
In the following a more detailed illustration of the description method and of the learning algorithm is provided; then some preliminary experimental results are examined, discussing some possible improvements.

2 Preprocessing and Description Phases

Characters can be considered as ribbon-like shapes whose thickness is not significant, but in a few special cases. Thus, an appealing preprocessing technique for obtaining a synthetic representation of them is thinning (see fig. 1b). However, to be effective for recognition purposes, the obtained thin lines should preserve the shape of the original ribbons. Unfortunately, this is one of the main problems for almost all the presently available thinning algorithms. In fact, they introduce shape distortions particularly at the join and crossing of strokes. We use an original method [8] that allows to substitute to the ribbon a polygonal line centered within it and faithfully reflecting its shape (see fig. 1c).

The polygonal line is then decomposed into pieces with the constraint that each piece can be simply described and at the same time represents a perceptively significant feature of the character. The shape of unconstrained handprinted characters, produced by a large number of writers, shows an extreme variability. Strokes having the same perceptive significance can have quite different shape and size, some strokes may be missed or added with respect to the hypothetical ideal representative of each class, and relative position of strokes may be partially altered. Nevertheless it has to be assumed that such variations are not so dramatic as to destroy the essential features making characters recognizable by a human observer. In order to cope with character variability and to single out the features most characteristic and invariant for members of a shape class, the polygonal line representing a character is decomposed into circular arcs (see fig. 1d) by a fitting algorithm [9]. The circular arc has been chosen since it provides the simplest, but still good approximation of the typical strokes used to form characters. Since this description scheme is tailored to a first level of classification, the fitting algorithm has been tuned to smooth as much as possible differences between similar shapes.

Character components, i.e. circular arcs, are described by using two features: the size (normalized with respect to the size of the character) and the angle spanned by the arc. The relations between arcs are described in terms of the contact points between them. Three types of contact are described and, in one case, the angle between arcs at the contact point is also measured. The attribute values of both arcs and their relations are roughly quantized as shown in fig. 1f. It is worth noting that all the attributes are independent of the absolute orientation of the arcs, resulting in a description which is not affected by a rotation of the character. Of course the orientation has to be taken into account in the subsequent levels of the classification, in order to discriminate the actual classes. The arcs and their relations form the structural description of the character, represented by means of an Attributed Relational Graph (ARG) whose nodes correspond to the circular arcs, and whose edges correspond to the relations between contacting arcs (see fig. 1e).



ARCS		JUNCTIONS	
Attributes	**Values**	**Attributes**	**Values**
Size	large medium small	**Type**	T-like V-like X-like
Angular Span	straight bent loop	**Angle**	acute medium obtuse

(f)

```
has3nodes(g1).
has(g1, n1).
has(g1, n2).
has(g1, n3).
large(n1).
straight(n1).
medium(n2).
bent(n2).
medium(n3).
bent(n3).
tlike(n2, n3).
tlike(n3, n1).
vlike(n2, n1).
obtuse(n2, n1).
vlike(n1, n3).
medium(n1, n3).
```

(g)

Fig. 1. The phases of the structural decomposition: (a) the initial image, (b) the thinned image, (c) the polygonal approximation, (d) the resulting circular arcs, (e) the ARG of the character, (f) the features used for shape description with their attributes and values (the angle values are assigned only to junctions of the V-like type), (g) the description in terms of Prolog predicates.

In order to describe an ARG in terms of simple logical relations we have defined, for each possible attribute of a node or an edge, a set of first-order predicates corresponding to the values of the attribute; in addition, another predicate indicates that a node belongs to a given graph. In fig. 1g an example of description of a character in terms of logical relations is shown.

3 Description of the Learning Algorithm

The method we have employed to learn the descriptions of the classes is based on Quinlan's FOIL algorithm [5] for Inductive Logic Programming (ILP). Inductive Logic Programming is a machine learning methodology aimed at learning new concepts from a set of examples, where both the concepts and the examples are described by means of a suitably chosen subset of first order predicate calculus (usually Horn clauses); in our application, a concept is a description of a class. More formally, an ILP system is given a logic program B, which encodes the eventual background knowledge about the problem, and a set T of examples, represented by logic facts, partitioned into positive and negative examples (subsets T^+ and T^-), which in our case correspond to characters of the class being defined and of the other classes respectively. The system attempts to find a new logic program P (a class description expressed as a set of rules) such that:

$$\forall t \in T^+, B \wedge P \Rightarrow t \qquad \text{and} \qquad \forall t \in T^-, B \wedge P \not\Rightarrow t \,. \tag{1}$$

If the samples in T are sufficiently representative of the concept being learned, an ILP algorithm is expected to find a concept definition which is more general than the supplied examples; only in this case we can speak of inductive learning.

The algorithm we have employed is able to learn only one predicate at a time, requiring different learning phases to learn a logic program P composed of more than one predicate; this is not a severe restriction for our application. Hence, the program being learned is formed by one or more *clauses* defining a Prolog predicate p; each clause can be expressed as:

$$p(X_1, \ldots, X_n) :- l_1,\ l_2,\ \ldots\ l_m .$$

where the l_i are the preconditions, or *literals*, which can have one of the following forms:

$$q(Y_1, \ldots, Y_k),\quad Z_i = Z_j,\quad \mathtt{not}(q(Y_1, \ldots, Y_k)),\quad \mathtt{not}(Z_i = Z_j).$$

The algorithm performs the search in the concept space by successive specializations. The initial hypothesis is the maximally general definition of the concept being learned, that is the trivial definition that encloses all the examples in the training set, positive and negative; it corresponds to a clause with no preconditions.

At each step, the definition is refined by adding a new literal to the clause; this way, the definition becomes more specialized, covering a smaller number of examples. The aim of the algorithm is to find a set of preconditions that cover all the elements in T^+, but no element in T^-. The former condition is not always satisfied, so if the algorithm has reached a clause which covers no element in T^-, this clause is added to the definition of the concept being learned, and the construction of a new clause is started to cover the remaining positive examples. Eventually, the algorithm will either arrive to cover the whole T^+, thus succeeding in learning the desired concept, or it will be unable to find a clause which covers the remaining positive examples, and will then terminate without finding a complete definition. The latter situation may happen if the information provided by the background knowledge B and by the training set T is not sufficient.

The most delicate phase of this algorithm is the choice of the literals to be added for specializing the current clause, which obviously cannot be based on an exhaustive search, since the computational cost would grow exponentially with the number of literals. Hence, the adopted algorithm chooses the literals in a *greedy* fashion, i. e. at each step the most promising literal is added to the clause, considering only the effect of that single literal on the clause, without any form of lookahead or backtracking; this fact may lead to a suboptimal set of literals. The choice is driven by a preference criterion, which in our case is based on Information Theory entropy [5].

4 Application to OCR: Discussion and Perspectives

The experimental phase has been carried out using the ETL-1 database [10] of handprinted characters. A set of about 500 characters, randomly selected, has been used for training the system. In fig. 2, the structural decomposition of characters used for the training are reported. Note that, starting from the 500 characters we have obtained 238 distinct character descriptions. The class descriptions produced by the learning algorithm (see fig. 3) have been verified against a test set of about 100 specimens for each class (totally less than 2600 ARGs).

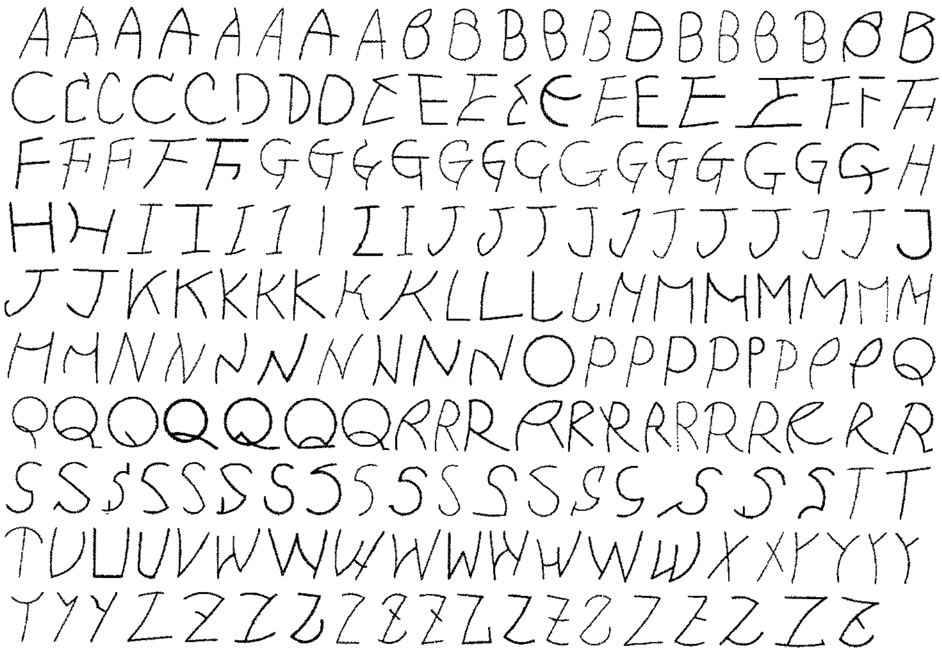


Fig. 2. The training set chosen in our experiments (after applying structural decomposition).

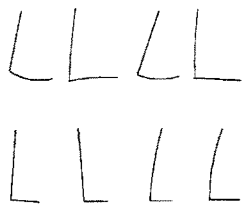
Table 1 summarizes the obtained classification results. The table should be read as a preliminary analysis aimed at verifying if inductive learning is mature enough to be applied to a real handprinted character recognition application. Nevertheless, an overall analysis of this table seems to suggest that our convincement is reasonable, because the table is sparse (the higher the number of 0-cells, the lower the misclassification ratio) and most elements of the main diagonal have high values.

However, as we expected, these results shows that the classification rules obtained at this first level are not always sufficient to recognize the actual class. The most evident case is that of class D, whose samples in a large number of cases are assigned to class O, class U or class L. Other examples of classes that often give rise to misclassification are the class B, confused with R, and the class U, confused with C, L or S. This depends on the description employed, which tends to absorb the difference among these classes in order to favor the grouping into pool-classes. Hence, the obtained rules must be suitably merged to form descriptions of the pool-classes, which can be reliably used, at this first level, to limit the decision of the classifier to a small number of actual classes.

```
is_L(A):-
    has2nodes(A),
    has(A, B),
    medium(B, C),
    medium(C),
    straight(B).
```

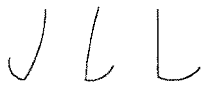
(a)

```
has2nodes(g1).
has(g1, g1n0).
large(g1n0).
straight(g1n0).
has(g1, g1n1).
medium(g1n1).
straight(g1n1).
vlike(g1n0, g1n1).
medium(g1n0, g1n1).
```



(b)

```
has2nodes(g2).
has(g2, g2n0).
large(g2n0).
straight(g2n0).
has(g2, g2n1).
medium(g2n1).
bent(g2n1).
vlike(g2n0, g2n1).
medium(g2n0, g2n1).
```



(c)

Fig. 3. One of the clauses found out by our system for the class L: (a) the rule discovered, (b)-(c) some of the samples in the test set matching the rule and their representations.

A successive step is needed to assign a sample to one of the classes which are contained in the selected pool-class; the definition of classification rules for this step is less critical for the ILP system, since it has to deal with a considerably smaller number of training samples and of classes. Furthermore, this step could be performed using other features, more effective with regard to the considered bitmaps (those belonging to the pool-class); these specialized features can be extracted only from the samples which are ascribed to the pool-class, if they are needed by the system in order to separate the single classes.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Rej	Tot
A	89	0	0	0	0	2	1	0	0	0	1	0	1	3	0	0	2	1	0	1	0	0	1	0	0	2	2	99
B	9	58	1	3	5	5	9	0	0	1	0	0	1	1	0	0	12	34	5	0	0	0	0	0	0	4	3	98
C	0	0	81	1	0	0	0	0	0	1	0	1	0	0	9	0	0	0	2	0	1	0	0	0	0	0	4	99
D	1	2	1	13	0	1	0	0	0	0	0	19	0	0	33	3	4	1	20	2	21	0	0	0	2	0	0	94
E	0	1	0	0	43	2	6	0	0	0	3	0	2	0	0	0	0	1	0	2	0	0	1	0	13	8	23	100
F	2	0	0	0	9	54	2	0	5	0	6	0	0	0	0	0	0	0	5	2	0	0	0	0	16	1	9	98
G	4	1	0	0	2	2	61	1	5	1	3	1	1	1	0	1	8	5	3	0	1	0	1	0	1	2	14	97
H	0	0	0	0	0	0	0	84	6	0	5	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	4	99
I	0	0	1	0	0	2	0	1	80	7	0	0	1	0	0	0	0	0	0	5	0	0	0	0	0	0	2	97
J	0	0	3	0	0	0	0	0	4	74	0	3	0	0	0	0	2	0	2	0	0	0	0	0	0	0	9	92
K	0	0	0	0	1	8	0	1	2	0	74	0	3	0	1	0	0	0	0	0	0	0	2	6	0	1	4	100
L	0	0	3	0	0	0	0	0	0	1	0	83	0	1	0	0	0	0	2	4	3	3	0	0	1	0	1	100
M	1	1	0	0	1	2	2	1	1	2	6	0	70	3	0	1	0	0	6	0	0	0	11	0	0	10	4	97
N	0	0	0	0	0	1	0	2	1	0	6	0	3	74	0	1	0	1	3	0	0	2	0	0	0	3	5	98
O	0	1	0	0	0	0	1	0	0	0	0	5	1	0	84	0	2	0	4	0	6	0	0	0	0	0	1	99
P	0	0	0	1	1	0	0	0	0	4	0	1	0	0	0	89	5	5	1	1	2	0	0	0	4	0	0	99
Q	7	14	0	7	0	1	1	0	0	0	0	0	0	0	2	3	65	5	4	0	1	0	0	0	4	0	1	93
R	0	1	1	0	0	6	4	0	1	0	6	0	5	1	0	1	2	70	2	0	1	0	1	0	0	4	16	98
S	0	3	1	1	1	3	3	0	0	1	0	0	10	1	0	1	3	1	85	0	1	0	0	0	6	6	3	99
T	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	93	0	1	0	0	2	0	0	97
U	0	2	26	3	0	0	0	0	0	0	2	21	0	2	0	0	0	0	22	0	28	0	2	0	1	1	6	93
V	0	0	0	0	1	1	0	0	0	0	0	12	0	4	0	0	0	0	2	3	3	74	0	0	1	5	0	99
W	0	0	0	0	2	3	1	0	0	0	8	0	3	0	0	0	0	0	1	0	0	0	82	1	0	5	2	97
X	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	94	1	0	2	99
Y	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	94	0	0	98
Z	0	0	0	0	4	5	4	0	2	2	3	0	3	10	0	0	0	1	1	0	2	1	3	0	0	67	5	95

Table 1. Classification results on the test set. Cell (i,j) reports the number of samples of the class 'i' attributed to the class 'j'. The Rej column reports the number of unrecognized samples.

Moreover, by examining our system at work, we have noted that, because of its greedy behavior, it often chooses an inadequate literal in an early step which leads to a weak clause. Since, as already pointed out, our algorithm doesn't include any kind of backtracking, there is no way for replacing the chosen literal by another one. In order to overcome the illustrated problem, future versions of our system will be based on other kinds of search techniques, such as beam search [11] which maintains at each step a limited number of promising clauses instead of just the best one.

5 Conclusions

This work has collected several aspects of ILP and OCR (with particular regard to handprinted character recognition), trying to sketch a guideline in order to realize a robust system inspired to ILP methods, able to find out prototypes from a suitably chosen training set of handprinted characters. A preliminary test has been performed on a few samples extracted from ETL-1 database to confirm our expectation about applicability of the method. The obtained results seem to suggest to cope with the unconstrained variability of the handprinted characters by means of aggregating more classes in a same pool-class, through a hierarchical description and classification scheme [7].

This kind of choice is particularly well-suited to be used with ILP methods, because it can decrease significantly the number of target relations at each level of the hierarchy, resulting in a better performance of the learning phase.

References

1. Pavlidis, T: Structural Pattern Recognition. Springer-Verlag (1977)
2. Eshera, M. A., Fu, K.S.: An image understanding system using attributed symbolic representation and inexact graph matching. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI–8, n. 5 (1986) 604–617
3. Rocha, J., Pavlidis, T.: A shape analysis model with applications to a character recognition system. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. PAMI–16, n. 4 (1994) 393–404
4. Hsu, J., Hwang, S.: A machine learning approach for acquiring descriptive classification rules of shape contours. Pattern Recognition, Vol. 30, n. 2, pp. 245–252, (1997)
5. Muggleton, S.: Inductive Logic Programming. New Generation Computing, Vol. 8, n. 4 (1991) 295–318
6. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning, Vol. 5, n. 3 (1990) 239–266
7. Cordella, L.P., Foggia, P., Sansone, C., Vento, M.: Hierarchical description and recognition of linear shapes. Proc. 3rd International Workshop on Visual Forms (1997)
8. Boccignone, G., Chianese, A., Cordella, L.P., Marcelli, A.: Using skeletons for OCR. In: Progress in Image Analysis and Processing, World Scientific Publ. (1990)

9. Chianese, A., Cordella, L.P., De Santo, M., Vento, M.: Decomposition of ribbon-like shapes. Proc. 6th Scandinavian Conference on Image Analysis. Oulu, Finland (1989) 416–423
10. ETL Database distributed by Electrotechnical Laboratory – Japanese Technical Committee for OCR, during the 2nd ICDAR (1993)
11. Lavrac, N., Dzeroski, S.: Inductive Logic Programming: Techniques and Applications. Ellis Horwood (1994)