Inductive Logic Programming for Symbol Recognition

Santosh K.C., Bart Lamiroy, Jean - Philippe Ropers

HAL Id: inria-00430927
https://inria.hal.science/inria-00430927

Submitted on 10 Nov 2009

Inductive Logic Programming for Symbol Recognition

Santosh K.C.
INRIA Nancy - Grand Est – LORIA
615, rue du Jardin Botanique
54600 Villers-lès-Nancy – France
e-mail: Santosh.Kc@loria.fr

Bart Lamiroy
Nancy-Université – INPL – LORIA
Campus Scientifique - BP 239
54506 Vandoeuvre-lès-Nancy CEDEX – France
e-mail: Bart.Lamiroy@loria.fr

Jean-Philippe Ropers
Nancy-Université – INPL – LORIA
Campus Scientifique - BP 239
54506 Vandoeuvre-lès-Nancy CEDEX – France

Abstract

In this paper, we make an attempt to use Inductive Logic Programming *(ILP) to automatically learn non trivial descriptions of symbols, based on a formal description. This work is a first step in this direction and is rather a proof of concept, rather than a fully operational and robust framework.*

The overall goal of our approach is to express graphic symbols by a number of primitives that may be of any complexity (i.e. not necessarily just lines or points) and connecting relationships that can be deduced from straightforward state-of-the art image treatment and analysis tools. This representation is then used as an input to an ILP solver, in order to deduce non obvious characteristics that may lead to a more semantic related recognition process.

1 Introduction

The main and primary goal of image analysis is to eventually find the means of bridging the semantic gap between low level descriptions of an image and the concepts of what is presented within it. The global state of the art assumption is that this may be obtained through an adequately conceived interaction between shape descriptions and the comparison or distance measurements between them on the one hand, and classification or grouping techniques to associate these descriptions to higher level concepts on the other hand.

Trying to express visual information using "natural" descriptions has actually been the original underpinning drive behind structural pattern analysis. Most often, this is done by first extracting low level visual cues that form the basic lexical data, and by proceeding by some grouping algorithm in order to express relationships or properties that are then translated into more and more complex "vocabulary" that triggers higher level rules, eventually expressing terminal concepts [14].

Our approach is slightly different in the way that we don't try to construct a chain of syntactic rule triggering, but rather build our vocabulary on direct extraction of (more or less) complex structures in the images. This vocabulary is based on currently unpublished work [8] that characterises symbols by a set of very robust, local structures. These structures need not necessarily be extracted by a structural or syntactic methods. In our case, for instance, we have developed specific and specialised detectors for circles [11], oriented corners, loose endpoints, rectangles, *etc*. Once a symbol is expressed as a set of elementary items, we use a reduced version of the force histogram based approach [15] to position all items relatively one to another by using a quantitative assessment of directional spatial relationships (such as "to the right of", "above", "south of"...) between two items in a way that corresponds quite closely to natural language and perceptual coherent relative positioning.

This allows us to express symbol descriptions with first-order logic predicates, expressing their type as per the vocabulary previously described and expressing the relative positioning one to another. Examples of the exact representation is given in section 4 and subsequent ones, where the experiments are described.

This framework gives us a straightforward way of describing the image that combines both expressiveness and very high flexibility. On the one hand, one can reduce or extend the size of the vocabulary in function of what robust descriptors are available. They may even be obtained

IEEE
computer
society

using statistical or signal based extractions [17]. Furthermore the relations that express the relative positioning need not only by as simple as those represented, and can even include more quantitative information (*e.g.* [4, 2, 9]).

The remaining problem is how to explore what this new representation can offer in terms of recognition, classification of learning of concepts. We are going to do this in the next sections, by feeding these data to a Inductive Logic Programming process. We first give a very brief introduction to Inductive Logic Programming and how it can contribute to learning visual classes for symbol recognition in section 2. We then show how this behaves on real data in section 4.

2 Inductive Logic Programming

Inductive Logic Programming (ILP) [16] combines automatic learning and first order logic programming. It requires three main sets of information, the automatic solving and deduction theory set aside:

1. a set of known vocabulary, rules, axioms or predicates, describing the domain knowledge base $\mathcal{K}$;

2. a set of positive examples $\mathcal{E}^+$ the system is supposed to describe or characterise with the set of predicates of $\mathcal{K}$;

3. a set of negative examples $\mathcal{E}^-$ that should be excluded from the deducted description or characterisation.

Given these data, the ILP solver is then able to find the set of properties $\mathcal{P}$, expressed with the predicates and terminal vocabulary of $\mathcal{K}$ such that the largest possible subset of $\mathcal{E}^+$ verifies $\mathcal{P}$, and such that the largest possible subset of $\mathcal{E}^-$ does not verify $\mathcal{P}$. This approach has already been successfully used for extraction of semantics from written text [6] or in document and image analysis structures [1, 5].

This approach allows for learning common properties within classes of symbols such as to express non-trivial knowledge of visual representation of more semantic concepts. We illustrate this in the next section. This work currently only concentrates on the Image Analysis part of the problem, and uses ILP as a black-box framework. The Inductive Learning Programming solver used in our experiments – Aleph – is freely available from the Oxford University Computing Lab[1].

In the following sections we will be using datasets of electrical graphical symbols, expressed using our first order logic vocabulary, and coming from an electric wiring component database [8][2].

---

3 Data Representation

In order to show what kind of data we actually manipulate, we have selected symbols 225_2 and 226_2 from Figure 1 as positive examples. All others as considered as counter-examples. Expressed in our visual vocabulary, symbol 225_2, for instance, is described as:

```
%    225_2*****************************
type(prim_170,cornerne). type(prim_171,cornernw).
type(prim_172,cornerse). type(prim_173,extremity).
has_element(img_225_2,prim_170).
has_element(img_225_2,prim_171).
has_element(img_225_2,prim_172).
has_element(img_225_2,prim_173).
nw(prim_170,prim_171). n(prim_170,prim_172).
nw(prim_170,prim_173). se(prim_171,prim_170).
ne(prim_171,prim_172). n(prim_171,prim_173).
s(prim_172,prim_170). sw(prim_172,prim_171).
w(prim_172,prim_173). se(prim_173,prim_170).
s(prim_173,prim_171). e(prim_173,prim_172).
%fin 225_2*****************************
```

The output of the ILP solver consists of à `[theory]` section, containing the rules that define the positive example set. For each rule of the theory, the solver gives matching statistics, indicating the precision of the rules (how many positive examples covered, and how many negative examples). For a perfect match, the theory section should consist of one single rule covering all positive examples and no negative examples. Further experiments will show that this is not always attainable. Sometimes the theory is composed of multiple rules, each of which covering a subset of the positive examples. Sometimes negative examples are covered by the theory as well. In our example, this gives:

```
[theory]
[Rule 1] [Pos cover = 2 Neg cover = 0]
symbol(A):-
    has_element(A,B), type(B,cornerne),
    has_element(A,C), n(B,C), type(C,cornerse).

[positive examples covered]
    symbol(img_225_2).
    symbol(img_226_2).
[negative examples covered]

test
[covered]
symbol(img_225_2):-
    has_element(img_225_2,prim_170),
    type(prim_170,cornerne),
    has_element(img_225_2,prim_172),
    n(prim_170,prim_172),type(prim_172,cornerse).
```

Following the `[theory]` section come two sections giving the examples covered by the theory: `[positive examples covered]` and `[negative examples covered]`. These two

sections simply explicit the occurrences of examples covered by the ruleset.

The last part [covered] is simply an example of one of the covered occurrences, as to allow "visual" verification.

The full interpretation of the output of our solver is that symbols 225_2 and 226_2 can be formally and completely distinguished from the other symbols by the fact that the dispose of two vertically aligned corners like this ⌐ . A visual inspection of Figure 1 does not allow to find any counter-examples. A more formal analysis of the image descriptions confirms this
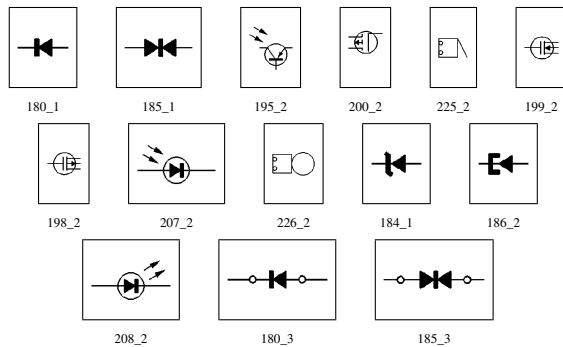
4 Global Behaviour

Figure 1. First image set for ILP experimentation.

Let images {195_2, 198_2, 199_2, 200_2, 207_2, 208_2} from Figure 1 be positive examples, representing the symbol of which the representation is to be learnt, and all others be counter examples. The ILP solver gives the following result:

```
symbol(A):-
    has_element(A,B), type(B,circle),
    has_element(A,C), inside(C,B),
    type(C,cornernw).
```

This experiment, translated into natural language, means that the chosen examples all have circles containing a northwest corner element (⌐). With this rule, all positive examples are perfectly classified with respect to the negative ones.

However, when selecting another set of symbols, like {180_1, 180_3, 184_1, 185_1, 185_3, 186_2}, the system is not able to reduce the set to a single predicate:

```
[Rule 1] [Pos cover = 1 Neg cover = 0]
symbol(img_180_1).
```

```
[Rule 2] [Pos cover = 2 Neg cover = 0]
symbol(A):-
    has_element(A,B), type(B,circle),
    has_element(A,C), e(B,C),
    type(C,cornernw).
```

```
[Rule 3] [Pos cover = 3 Neg cover = 0]
symbol(A):-
    has_element(A,B), type(B,blackthick),
    has_element(A,C), type(C,cornersw),
    has_element(A,D), ne(C,D).
```

Image 180_1 is not covered by the predicates, and the remaining, positive examples, are split up into two distinct sub-classes, each of them covered by a separate rule. The first one, describing the symbols as containing a circle ○ and a corner ⌐, placed to the east of it. The second one is far more interesting, and gives an outstanding reason of using ILP solving. [Rule 3] indeed mentions that the corresponding symbols contain a black thick component ◄ and corner ⌐, but, more interestingly, that it contains a third, *unspecified – any –*, primitive at a north-east position of the corner. The fact that it is possible to express these very generic relationships (regardless of the underlying shape !) is something numeric learning or classification techniques [13, 10] cannot achieve.
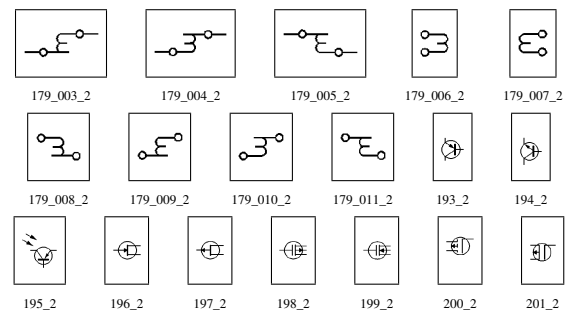
5 Choosing a Side

Figure 2. Second image set for ILP experimentation.

Let us now consider images {179_006_2, 179_007_2, 179_008_2, 179_009_2, 179_010_2, 179_011_2} from Figure 2 as positive examples, while remaining symbols are the counter examples. The generated theory is hardly able to find any common rules between them. Only two positive examples are covered whithin one rule, while the others are generated independently. This, actually, is quite nor-

mal, since vocabulary used to represent the symboles (circles, thick components, corners ...) is very badly suited for distinguishing between them. The main point is, however, that, if we invert the positive and negative examples (*i.e.* we try to learn a common characterisation of the set of counter examples: {193_2, 194_2, 195_2, 196_2, 197_2, 198_2, 199_2, 200_2, 201_1}, the ILP solver generates one single rule.

```
symbol(A):-
    has_element(A,B), type(B,cornernw).
```

The developed rule covers all the examples in a single predicate and only a primitive ⌐ is needed to describe the set. It is thus very important to try and characterise both positive and negative example sets.

Another point of interest with this experiment is that the generated rule might seem "visually" weird with respect to human interpretation. Indeed, one would naturally describe the example set as "*circles containing stuff*". Although this rule is perfectly acceptable in our framework, giving something like:

```
symbol(A):- has_element(A,B), type(B,circle),
    has_element(A,C), inside(B,C).
```

it is, unfortunately, more complex than the rule the automatic solver found. This is due to the fact that the ILP solver works in a "closed world" of predicates, vocabulary and examples and cannot infer that a given solution might be "more generic" than another with respect to human interpretation standards.

6 Learning Set Induced Limits

In the prevous sections we addressed the question on how the predefined vocabulary affects the learning process. In this section we address the influence of the learning samples. Let us consider the case of a specific semantic concept: a diode. The positive and negative learning examples are taken from Figure 3.

First, let the set of positive samples be {180_1, 180_3, 184_1, 185_1, 185_3,186_2}. The negative ones are {194_2, 198_2, 210_2}. The produced theory is:

```
symbol(A):-
    has_element(A,B), type(B,blackthick).
```

This theory thus implies that every symbol which contains a black thick object is a diode. This rule, of course, is too "simple", due to the fact that the learning set was too limited (it is, however, quite correct in the context of the closed learning set world). Let's add a negative example which contains black thick objects: the negative sample set becoming {194_2, 198_2, 210_2, 195_2}.
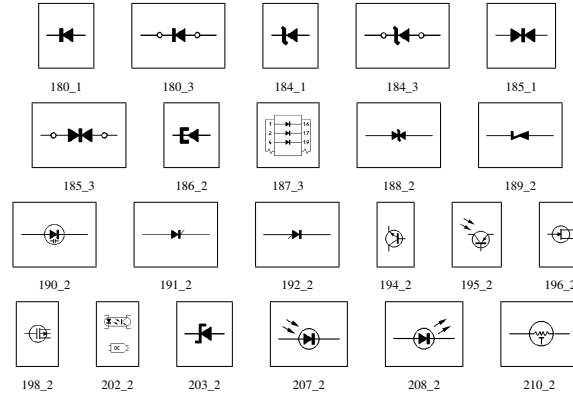

Figure 3. Third image set for ILP experimentation.

```
symbol(A):-
    has_element(A,B), type(B,blackthick),
    has_element(A,C), type(C,cornerse).
```

As expected, the rule evolved to take into account the new counter-examples, but still only relies on the presence of two kinds of primitive: "blackthick" and "cornerse". We can now add negative examples which contain these primitives. The set of negative examples become {194_2, 198_2, 210_2, 195_2, 196_2, 202_2} and the theory is now:

```
symbol(A):-
    has_element(A,B), type(B,blackthick),
    has_element(A,C), type(C,cornerse),
    has_element(A,D), e(D,C), inside(D,B).
```

At this stage, the spatial relations become the important criterion. The interpretation of the rule is "*having a SouthEast-corner which is on the right of another object which contains the black thick object*". This is not exactly the rule that we might be expecting, but we have to be aware that it is a rule based on only 6 negative examples and that the primitive detection is not perfect. By extending to the full set of 19 positive examples and the 6 negative examples previously used, the system obtains:

```
symbol(A):-
    has_element(A,B), type(B,blackthick),
    has_element(A,C), type(C,cornernw),
    has_element(A,D), e(C,D), type(D,cornerse).
```

What is interesting to note here, is that, compared to statistical learning models, the system adapts the complexity of the classification with respect to the learning data, without need for any parametrization of any sorts. On the other side, however, it is also quite straightforward to see that, if the learning set is contradictory with respect to the available vocabulary, it cannot deduce any classification rule.

7 Conclusion and Further Work

This paper presents the first step towards another approach of symbol recognition and representation, by combining robust elementary form detectors that compose a predefined, but extensible vocabulary. This vocabulary is combined with relative positioning in order to obtain a first order logic based description of the symbols, on which ILP can be used to extract "semantic" contexts or concepts. The interesting part of this is that the description of the symbols can now be easily mixed with other, more context related information. The main advantage of this approach is, that information need not necessarily be visually represented (for example, from surrounding text), and it thus opens a new scope of possible combined text/image concept characterisation and learning. It is even possible to expand the framework to generate symbols from the obtained descriptions (either for visual validation of classification results or for automatic illustration generation), as shown in [12].

However, the method, as it currently stands, is limited by the expressive power of the used vocabulary. Further work is therefore consisting of extending the initial vocabulary, by introducing the notion of connexity, refining the inclusion predicate and using relative distance and size (close, far, large, small ...). These are all straightforward extensions that are readily available from an image analysis standpoint. Further, less straightforward, work will concern inclusion of more numerical or statistical form descriptions that might be able to better quantify differences between the shapes and will need to rely on Markov Logic [18, 7] to handle the numerical part. More prospective work will be to connect this to Formal Concept Analysis [3] and Galois Lattices to achieve unsupervised learning of visual concepts.

Acknowledgements

References

[1] A. Amin, C. Sammut, and K. C. Sum. Learning to recognize hand-printed Chinese characters using Inductive Logic Programming. *International Journal of Pattern Recognition and Artificial Intelligence*, 10(7):829–847, 1996.

[2] B. Bennett and P. Agarwal. Semantic categories underlying the meaning of 'place'. In *Spatial Information Theory: Proceedings of the 8th International Conference (COSIT-07)*, volume 4746 of *Lecture Notes in Computer Science*. Springer-Verlag, 2007.

[3] G. S. Bernhard Ganter and R. W. (eds.), editors. *Formal Concept Analysis: Foundations and Applications*. Number 3626 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 2005.

[4] I. Bloch. Fuzzy Spatial Relationships for Image Processing and Interpretation: a Review. *Image and Vision Computing*, (23):99–110, 2005.

[5] M. Ceci, M. Berardi, and D. Malerba. Relational data mining and ilp for document image processing. *Applied Artificial Intelligence*, 21(8):317–342, 2007.

[6] V. Claveau and P. Sébillot. From Efficiency to Portability: Acquisition of Semantic Relations by Semi-Supervised Machine Learning. In *Proceedings of 20th International Conference on Computational Linguistics, COLING'04*, pages 261–267, Geneva, Switzerland, 2004.

[7] P. Domingos, S. Kok, D. Lowd, H. Poon, M. Richardson, and P. Singla. Markov logic. In *Probabilistic Inductive Logic Programming*, pages 92–117, 2008.

[8] FRESH. Final report on symbol recognition with evaluation of performances. Deliverable 2.4.2. - FP6-516059, January 2007.

[9] S. K.C., L. Wendling, and B. Lamiroy. Dynamic angle based theory in learning relative directional spatial relationship on components of raster symbols. In *Proceedings of 8th IAPR International Workshop on Graphics Recognition, La Rochelle, France*, July 2009.

[10] L. I. Kuncheva. Diversity in multiple classifier systems. *Information Fusion*, 6(1):3–4, 2005.

[11] B. Lamiroy, O. Gaucher, and L. Fritz. Robust Circle Detection. In Flavio Bortolozzi and Robert Sabourin, editors, *9th International Conference on Document Analysis and Recognition - ICDAR'07*, pages 526–530, Curitiba Brasil, 2007.

[12] B. Lamiroy, K. Langa, and B. Leoutre. Assessing classification quality by image synthesis. In *Proceedings of 8th IAPR International Workshop on Graphics Recognition, La Rochelle, France*, July 2009.

[13] C. Malon, S. Uchida, and M. Suzuki. Mathematical symbol recognition with support vector machines. *Pattern Recognition Letters*, 29(9):1326 – 1332, 2008.

[14] J. Mas Romeu, G. Sanchez, J. Llados, and B. Lamiroy. An Incremental On-line Parsing Algorithm for Recognizing Sketching Diagrams. In Flavio Bortolozzi and Robert Sabourin, editors, *9th International Conference on Document Analysis and Recognition - ICDAR'07*, pages 452–456, Curitiba Brasil, 2007.

[15] P. Matsakis, J. M. Keller, O. Sjahputera, and J. Marjamaa. The Use of Force Histograms for Affine-Invariant Relative Position Description. *IEEE Transactions on PAMI*, 26(1):1–18, Jan. 2004.

[16] S. Muggleton, Luc, and D. Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19:629–679, 1994.

[17] T. O. Nguyen, S. Tabbone, and O. Ramos Terrades. Symbol descriptor based on shape context and vector model of information retrieval. In *The 8th IAPR International Workshop on Document Analysis Systems*, pages 191–197, Nara Japan, 09 2008.

[18] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62:107 – 136, 2006.