MACHINE LEARNING TECHNIQUES FOR DETERMINING PARAMETERS OF CARTOGRAPHIC GENERALISATION ALGORITHMS

François LAGRANGE[*], Bruce LANDRAS[*], Sébastien MUSTIERE[**]
[*] Ecole Navale, Lanvéoc-Poulmic, France
[**] IGN and University Paris VI, France
[**] Sebastien.Mustiere@ign.fr

Working Group IV/5

KEY WORDS: Cartographic Generalisation, Machine Learning, Neural Networks, Parameters Setting.

ABSTRACT

This paper reports on research performed in the field of automated map generalization. We address the issue of determining how to set parameters of transformation algorithms. Empirical and theoretical studies have shown that, even given fixed map scale and purpose, these parameter values vary from one object to the other according to different characteristics such as the shape, size or environment of the object. Because of the complexity of cartographic rules and generalisation algorithms, we address this problem with techniques developed in the field of Machine Learning from examples. Specifically, we automatically learn, with neural networks, how to determine an algorithm parameters according to a set of measures describing the object to be transformed. We present the main issues to be addressed to use neural networks. We show that our approach is useful and that its main limit stands in the lack of good measures to describe an object. As a case study, this paper presents results of learning how to set the strength of a smoothing algorithm on a line from the French BDCarto database to represent a road on a 1/250,000 scale road map.

RÉSUMÉ

Cet article présente les résultats de recherches effectuées dans le domaine de la généralisation cartographique. Plus précisément, le problème traité est celui du paramétrage automatique des algorithmes de généralisation. Même pour produire une carte donnée, les algorithmes doivent être paramétrés différemment selon les caractéristiques des objets traités, comme leur forme, leur taille ou leur environnement graphique. Les difficultés de ce paramétrage reposent dans la complexité des règles cartographiques, leur manque de formalisation, et la complexité des algorithmes. Nous abordons ce problème avec des techniques développées dans le domaine de l'apprentissage automatique à partir d'exemples en Intelligence Artificielle, et plus particulièrement les réseaux de neurones. Nous apprenons à déterminer les paramètres d'un algorithme en fonction de mesures décrivant l'objet à traiter. Nous présentons les points clés pour pouvoir utiliser les réseaux de neurones. Nous montrons l'utilité de notre approche et mettons en valeur que ses principales limites reposent dans le manque de mesures efficaces de description des objets géographiques. Cette étude pratique est illustrée par la détermination de la force d'un algorithme de lissage à appliquer sur certaines routes de la base de données BDCarto pour effectuer une carte routière au 1/250.000.

1 INTRODUCTION

This paper reports on research performed in the field of automated map generalization, and more generally, in the field of automation of cartographic processes. One of the key issues in automating cartographic processes is determining *where*, *when* and *how* to use transformation operators [McMaster and Shea, 92]. We concentrate here on the question of *how* to use them, and more specifically, the question : given an object and an algorithm which is to be used on it, how to set the parameter(s) of this algorithm for this object ?

Empirical and theoretical studies have shown that, even given fixed map scale and purpose, these parameter values vary from one object to the other according to different characteristics such as the shape, size or environment of the object [Weibel, Keller and Reichenbacher 95; Ruas, 98]. Our objective is to find relationships (hereafter called 'hypotheses' to use the terminology of the Machine Learning field) linking, on the one side, a set of measures describing an object and, on the other side, possible parameter values of a given algorithm for use with this object for a specific generalization. Because of the complexity of cartographic rules and the difficulty of formalizing them, we address this problem with techniques developed in the field of Machine Learning from examples (see [Mitchell, 97] for an overview of Machine Learning). Specifically, we want to automatically learn hypotheses from a set of examples given by a cartographer.

This work closely follows a framework presented in [Reichenbacher, 95] and [Weibel, Keller and Reichenbacher, 95] and has even been performed from the same test data set. We first present (part 2) this work and its limitations; some of them enhanced by these authors themselves. We then define our learning task (part 3), how examples are collected (part 4), how learning has been performed (part 5) and the evaluation of results (part 6).

2 PREVIOUS WORK: KNOWLEDGE ACQUISITION BY PROCESS TRACING

2.1 Process description

We shortly describe here a study presented in [Reichenbacher, 95] and [Weibel, Keller and Reichenbacher, 95]. This study aimed at automatically learning how to set the parameter of the Lang algorithm [Lang, 69] to perform generalisation from the BDCarto (a French database with approximately 10 meters resolution) to a road map at 1/250,000 scale. Weibel, Keller and Reichenbacher claim that this can be done by taking advantage of generalisation process tracing and define the following process:

- The source data are, one the one hand, an extract of the BDCarto and, on the other hand, the same region manually generalized to make a 1/250,000 scale road map.
- They display on a GIS window the BDCarto as a foreground and the generalized data as a background.
- They apply on 52 roads from BDCarto the Lang algorithm with different parameters until the result is close to the manually generalized result. They store the parameter finally used for each road.
- Meanwhile, they compute a set of measures describing each road. Each measure is discretised into three classes (like LOW, MEDIUM and HIGH).
- Production rules are learnt from the set of 52 couples of “set of measures – parameter value” with symbolic machine learning tool like AQ15 [Hong et al. 86].

2.2 Limitations

As Weibel, Keller and Reichenbacher quote, their study has been done to show the technical feasibility of such a framework for process tracing, and this work needs to be continued. Particularly they quote that:

- The measures they used to describe a road are not expressive enough, they propose to add measures developped in [Plazanet, 96]
- The study should include more algorithms than the Lang algorithm, as it is not enough to perform generalisation.
- Validation of the learnt rules has not been performed.
- More sophisticated Machine Learning tools should be used.

We add to the limitations of this study that:

- The Lang algorithm is a filtering algorithm (removing points from a line) and is more concerned with data compression than cartographic generalisation. It is so difficult to closely follow the generalized roads with roads treated by the Lang algorithm. The knowledge acquisition is so difficult.
- Measures describing a road as well as Lang parameter are numerical data. The study used a completely symbolic Machine Learning algorithm (which use symbolic inputs and outputs, and produce symbolic rules) which is not the most adapted to the problem.
- Roads were considered as a whole, and treated in once with the same algorithm and parameter. [Ruas, 98] shows that using existing algorithms to generalize a road ask to split it first in homogeneous parts and to treat each different part with a different algorithm or with different parameters.
- Few examples where provided, compared to the complexity of the learning problem.

The following of the paper describes another test of automatic hypothesis learning to determine the parameter of a given algorithm and intend to overcome the previously described limitations.

3 LEARNING TASK

3.1 Choosing smoothing as a case study

First, we choose, as a case study, to learn the parameter of a smoothing algorithm. We choose smoothing because:

- Smoothing is a generalisation operation, performed by hand when necessary (contrary to filtering).
- Most of smoothing algorithms contain a single parameter (the strength of the smoothing).
- Smoothing strength is relatively independent of the environment of the objects to be smoothed. We will not need to describe the environment of the object to expect efficient learnt results (contrary to caricature or displacement)

- The need of adapting the smoothing strength to the objects has been enhanced during interactive evaluation of automatic generalisation results made at IGN-France by traditional cartographers (personal report).

Then, we choose to learn to smooth roads from BDCarto to a 1/250,000 scale road map, because this operation ask for a lot of smoothing. We choose to use the algorithm "gaussian smoothing", because of the known efficiency of this algorithm for linear objects. This algorithm contains one parameter σ, which is the strength of the smoothing.

Finally, as it is impossible to determine exactly *one* best smoothing strength to apply, we will not learn one value of parameter but one range of possible values.

3.2 Context of use

We first need to determine the context of use of the smoothing algorithm. Determining the parameters of an algorithm is not the first problem to address to perform automatic generalization: it is first necessary to determine on which object this algorithm can be used. Indeed it is useless to learn the parameter of a smoothing algorithm on a set of sharp bend series as we will not use smoothing on bend series, but rather caricature algorithms. Even, if we learn a smoothing parameter from examples containing roads to be smoothed as well as bend series, the useless examples can decrease the quality of the learnt hypothesis for the roads to be actually smoothed. The main reason of this is that the learning algorithm will try to solve the unsolvable problem of smoothing bend series and not focus on the interesting problem.

Like in the automated generalisation process "GALBE" [Mustière, 98], we do define the context of use of smoothing as: "all the parts of a road which are not in graphic coalescence conflict before generalisation". Figure 1 illustrates this: (a) is a road of the BDCarto, (b) is the same road after symbolization at 1/250.000 scale, (c) is the result of a coalescence detection, non coalesced parts are parts where the smoothing has to be done, (d) shows the result of such an automatic approach.

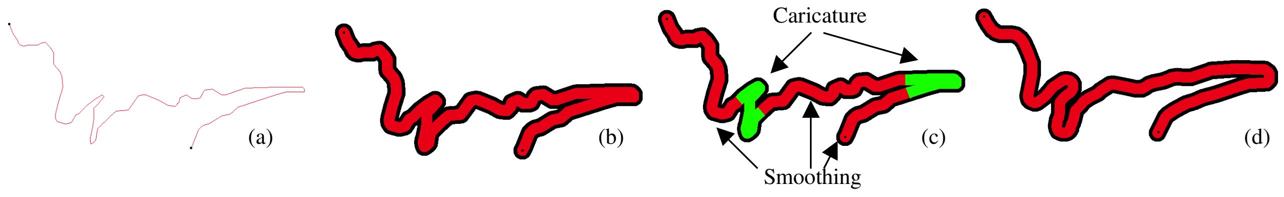

Figure 1. Context of use of the smoothing operation

3.3 Objects description

The choice of the smoothing strength depends on the shape of the treated part of the road. We so describe each part of the road by a set of measures found in the litterature [McMaster, 86], [Plazanet, 96] :

- M1: Symbol_width (in mm)
- M2: Length of the line (in mm)
- M3: Length / Number of points of the line (in mm)
- M4: Base / Length (Base is the length of the segment joining the two extremities of the line)
- M5: Length / Number of bends (in mm)
- M6: Length / Number of large bends (in mm; the number of large bends is considered as the number of bends after a given smoothing)
- M7: Height of the highest bend (in mm)
- M8: Average of the bends heights (in mm)
- M9: Length of the line / Length of the line simplified by [Douglas and Peucker, 73] filtering algorithm (no unit)
- M10: surface of the symbolized line / length of the line / symbol width (no unit)

3.4 Task definition

To sum up the previous considerations we define our learning task as *to learn form examples provided by a cartographer how to link, on the one hand, a set of measures describing the shape of non-coalesced parts of a road from the BDCarto to, one the other hand, the possible values of the strength of the gaussian smoothing algorithm on these parts to generalize them at 1/250,000 scale.*

4 EXAMPLES COLLECTION

Our source data are two data sets : an extract of the BDCarto (10 meter resolution) and the same area manually generalized that we consider as a reference.

The BDCarto data set is first split in homogeneous parts regarding coalescence, and only non coalesced parts are considered, as explained above. We then simultaneously display the reference and the non-generalized BDCarto and interactively determine between which and which strength the smoothing shall be done on the BDCarto road to obtain a visually road similar to the manually generalized road.

Figure 2 shows how suitable parameters are collected. Upper figures represent the BDCarto line smoothed with different parameters and the manually generalized line (dash-line); lower figure represent the symbolized BDCarto line for a 1/250,000 scale road map. From left to right we see the BDCarto line without smoothing (σ=0), not enough smoothed (σ=0.1), at the minimum of smoothing to be acceptable (σ=0.2), at the maximum of smoothing to be acceptable (σ=0.3) and too much smoothed (σ=0.4). For this example we so consider that good parameters for the smoothing are between 0.2 and 0.3 .

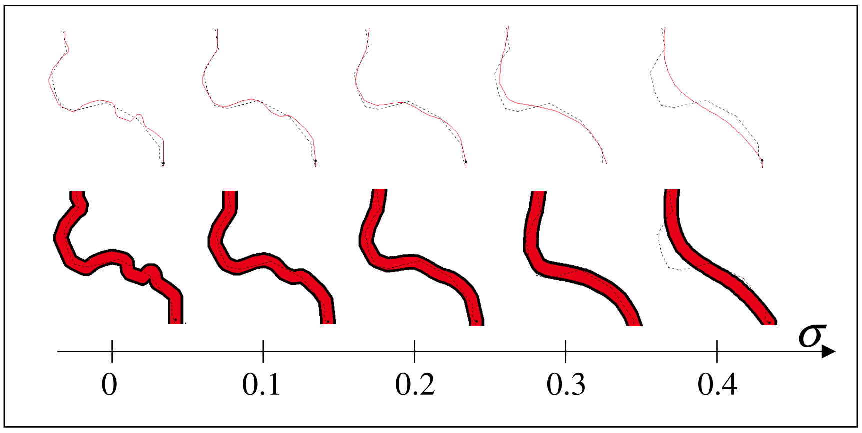

Figure 2. Collecting parameters for smoothing

Following the same method, we obtain a set of 170 examples. An extract of these examples is shown in Table 1. In term of Machine Learning our problem is characterized by a relatively small number of examples provided for learning and by the high quantity of fuzziness in the examples, due to the very subjective aspect of cartography and the fuzziness of our spatial analysis measures.

M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	**σ min**	**σ max**
0.5	12.0	0.30	0.97	1.3	1.7	0.10	0.05	1.9	1.3	0.20	0.30
0.3	40.8	0.36	0.77	1.6	1.6	0.90	0.14	1.7	1.5	0.32	0.64

Table 1. Two of the 170 collected examples

5 PARAMETER LEARNING AND LEARNING CONFIGURATION

Our objective is not only to determine procedural knowledge for our specific case, but also to analyze the adequacy of Machine Learning for the purpose of map generalization. To undertake this analysis, we have used neural networks with many different configurations. Neural Networks are known to be efficient in Machine Learning. They are particularly adapted to our case, as our examples contain only numerical data (input measures and output smoothing parameters). Neural networks can be seen as a way to approximate any function f from a set of examples (x,f(x)) – x being here our objects description and f(x) our parameter values.

The main difficulty for using neural networks stands in defining:

- the representation of inputs and outputs of the network,
- the model of the network,
- the architecture of the network,
- the learning protocol.

Some of these network parameters can be determined from well known results in literature, some others can only be determined by testing and evaluating results.

5.1 Representation of inputs and outputs

Because our task is to learn parameter values from measures describing an object, the inputs of our neural networks are obviously the set of 10 measures describing each object. In our study we want to determine a range of values for the smoothing strength. Several options can be foreseen and have been tried (Figure 3); we can either learn:

- one the one hand the minimum and, on the other hand, the maximum of σ
- on the one hand the average of the min and the max (medium value) and, on the other hand, the difference between the min and the max (the range).

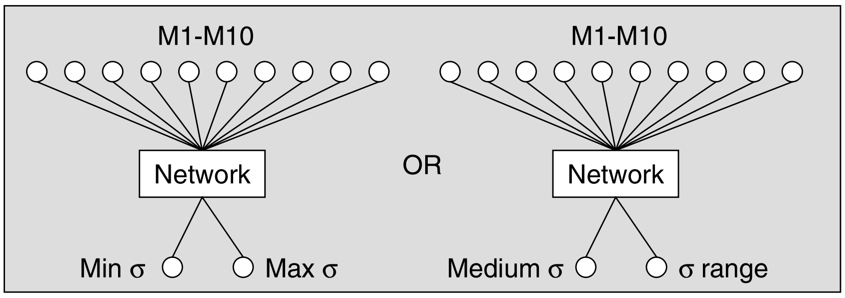

Figure 3. Which outputs ?

5.2 Model of network

The most known model of neural networks is the multilayer network with sigmoid units (see Figure 4) [Rumelhart, Hinton and Williams 1986]. This kind of network has been a lot of studied, theoretically as well as practically. Its ability to well approximate any functions is very well known, we will so use this kind of network.

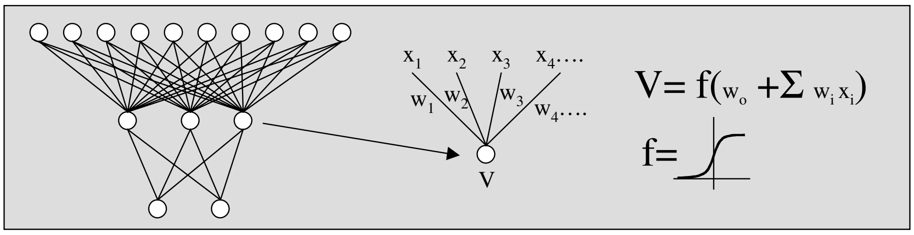

Figure 4. Multilayer networks with sigmoid units (here one hidden layer with 3 neurons)

Theoretical works show that it is useless to use more than two hidden layer. It can also be shown that, for optimally using this kind of network, inputs should be normalized between –1 and 1, and outputs should be normalized between 0 and 1.

5.3 Architecture of the network

Once the model has been chosen we must specify the architecture: i.e. the number of hidden layers and the number of neurons per layer. It is well known that if the neural network contains not enough neurons it will not be able to approximate complex functions and so will not be efficient to well determine the good outputs for new examples. It is also known that if it contains too much neurons it will "learn by heart" the provided examples and will not interpolate between then to well approximate the expected function. Some techniques exist to automatically find the best architecture of the system, but when we tested them we saw we do not have enough examples for these techniques to be efficient. Then, the only way to choose the best architecture is to try several one, and keep the one providing the best results. We tried 10 different architecture, from the very simple containing no hidden neurons, to a very complex one containing 100 hidden neurons. Figure 5 shows three different architecture with respectively no, one and two hidden layers.

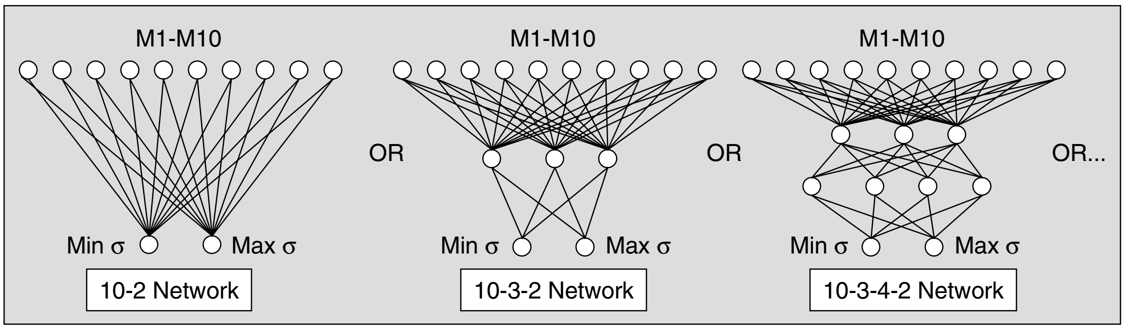

Figure 5. Choosing the architecture of the network

An other choice must be done: we want to learn two values (the min and the max, or the medium and the range). We can either use two independent neural networks, each of them with one output, or use one single neural network with two outputs (see Figure 6). Both options have been be tried.

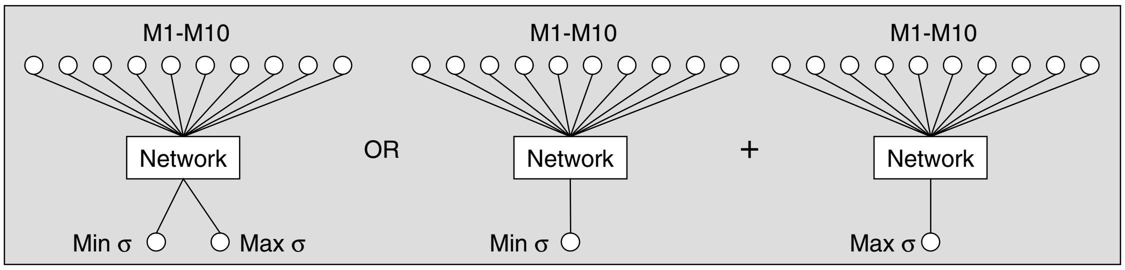

Figure 6. One or two neural networks ?

5.4 Learning protocol

Training a neural network means determining the weights (all the *w* in Figure 4), from the set of provided examples of inputs and outputs. This weights determination can be made by several techniques, all more or less derived from the backpropagation technique [Rumelhart, Hinton and Williams, 1986]. We tried four of these techniques (all of them are step by step techniques).

A difficulty encountered in all these techniques is to know when stopping the weights determination. If we do not perform enough steps the network do not well classify new examples because it did not learn to approximate the good function. If we perform too much steps the network is also bad because it learns “by heart” the provided examples. To determine when stopping the weights determination we use the “early stopping” method which consist on splitting the learning set in two sets (containing respectively 2/3 and 1/3 of the learning examples): one to train the network (defining step by step the weights) and one to stop it (when the network provides minimal mistakes on this set).

5.5 Test performed

Neural networks are well adapted to our problem. The difficulty to use them stands in the number of different configurations of networks that can be defined. For our study we fixed some network parameters (choice of multilayer networks, normalization of the inputs and outputs, the 10 measures are the inputs, use of “early stopping” technique). We then performed many tests to estimate the influence of the other parameters (representation of the outputs, number of network used, weights setting technique, architecture of the network).

6 HYPOTHESIS EVALUATION

6.1 Evaluation process

First of all, our set of 170 examples is randomly split into two data sets: one learning data set for learning hypothesis containing 130 examples, and one test data set for evaluating results containing 40 examples (putting ¾ of the examples in the learning data set and ¼ in the test data set is a usual proportion in Machine Learning). The learning data set is used to learn weights of neural networks with different configurations and the test data set is used to evaluate the quality of what is learnt.
The evaluation protocol used is a “train and test” protocol. That means:

- we use the learning data set to learn a neural network, with one given configuration of network parameters,

- we feed the network with the measures describing objects of the test data set and store the output provided by the network: these are the “learnt values”.
- we compare the learnt values to the expected value determined interactively while collecting the examples.

6.2 Evaluation results

Detailed results of all the performed experiments can be found in [Lagrange and Landras, 99] we only quote here the most important results.

6.2.1 In term of quality of the results. The minimal error found during all the experiment on the maximum and minimum of σ was approximately 0.15mm (in our test σ was defined between 0 and 1.25mm).
This result must be compared to the standard deviations of minimum and maximum of σ which are approximately 0.22mm (0.22mm would so be the error if we use a fixed value for the maximum and minimum of σ instead of using a learnt function to determine it). This means that machine learning is useful but that we did not succeed to obtain perfect results. As we performed many different tests (the presented one and others changing the examples data set) we think that the only way to obtain better results is to improve measures describing the roads.

Learning the maximum of σ has been more efficient than learning the minimum. It seems so easier to learn how far we can smooth than the minimum required smoothing.

6.2.2 In term of methodology. The most important parameter influencing the quality of the results is the architecture of the network. Other parameters also influence the results but in a lower extent. The best results where found for a “10-12-4-2” architecture (that means 10 inputs, 12 neurons in the first hidden layer, 4 neurons in the second hidden layer, 2 outputs).
Another result is that networks did not learn well **extreme values** (like learning when the minimum of σ is near 0). This is certainly due to the relatively small number of examples.

6.2.3 In term of measure relevance. We studied the influence of each input measure to the parameter determination on neural network (i.e. we empirically study the derivative of the outputs relatively to each input). This is a way to evaluate the relevance of each measure to our problem:
- the measures “Base/Length”, “ Length / Nb of large bends” and “Height of the biggest bend” are the most relevant measures. As proposed by [Plazanet, 96] measures starting from bends analysis seem to be relevant.
- the measures “Length” and “medium height of the bends” are also influencing parameter determination but in a lower extent.
- other measures were useless for our study.

6.3 Comparison to symbolic learning

[Weibel, Keller and Reichenbacher, 95] used symbolic learning techniques. These techniques have the advantage over neural networks to provide understandable hypothesis (like a set of rules or a decision tree). We also used symbolic learning to compare the results in term of quality to the one obtained by neural networks. We used C4.5 algorithm [Quinaln, 92] because it does not ask for a preliminary discretisation of examples: the discretisation is dynamically built during the learning. We clearly saw that neural networks are more efficient to learn numerical data from numerical inputs than symbolic learning techniques.

Nevertheless we quoted that symbolic techniques were more efficient than neural networks to learn extreme values. As a typical example we can notice that symbolic tools learnt the rule “if the road is a straight line than no smoothing is necessary” (more precisely the rule was “if Base/Length=1 then minimum of $\sigma = 0$). This is an obvious rule but that our neural networks did not exactly recognize.

We so proposed an hybrid approach, mixing the use of neural networks to learn intermediate values and symbolic tools to learn extreme values. This approach provided better results than each separated approach.

7 CONCLUSION

Our study shows the first interest of machine learning for our problem, that is learning from examples knowledge which are difficult to formalize. Then we have explained that machine learning can be used to determine the relevant descriptive measures for our problem (here mainly measures based on bends analysis). We think this ability of machine learning can be very useful.

We have shown that the parameters of the learning phase (e.g. architecture of the network) must be chosen with care as well as the problem representation (learning what ?). Then, evaluating the result is of first importance.

Efficiency of neural networks is well known and we performed a lot of experiments with a lot of different network configurations. Nevertheless we never succeeded to perfectly determine the smoothing parameter for roads. This allow us to conclude that our measures were not relevant enough, and that fully taking advantage of machine learning ask for the development of more relevant measures to describe geographic objects.

More globally we think that the development of more expressive spatial analysis tools describing geographic objects is a crucial point to guide cartographic generalisation processes and so to improve cartographic results.

ACKNOWLEDGEMENTS

All empirical experiments on neural networks described in this paper have been performed with SNNS© v4.1 software (Stuttgart Neural Network Simulator) that provides an easy access to many different neural network training options. This software can be downloaded from http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns .

REFERENCES

Douglas D.H. & Peucker T.K. 1973. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or it's Caricature. The Canadian Cartographer Vol 10(2) pp 112-122

Hong J., Michalski R.S. and Mozetic I. 1986. AQ15: Incremental Learning of Attributes-Based Description from Examples, the Method and User's Guide. Reports of the Intelligent System Group, Department of Computer Science, University of Illinois at Urbana-Champaign, ISG-86-5.

Lang T. 1969 Rules for the Robot Droughtsmen. The Geographical Magazine Vol. 42(1) pp 50-51

Lagrange F., Landras B., 1999. Application des réseaux de neurones à l'apprentissage des valeurs paramétriques des algorithmes de généralisation cartographique. Rapport de stage, Ecole Navale, Brest, France

McMaster, R. B. 1986. A Statistical Analysis Of Mathematical Measures for Linear Simplification The American Cartographer 13(2) p. 103-117

McMaster, R.B. & Shea, K.S. 1992. Generalisation in Digital Cartography. Ed. Association of American Geographers

Mitchell T.M. 1997. Machine Learning. McGraw-Hill International Editions.

Mustière S. 1998. GALBE: Adaptive Generalisation - The need for an Adaptive Process for Automated Generalisation, an Example on Roads. GIS'Planet 1 proceedings, Lisbon, September 98.

Mustière S., Zucker J.-D., Saitta L., 2000 (under print). An abstraction-based Machine Learning Approach to Cartographic Generalisation. Proceedings of SDH conference-Beijing-August 2000.

Plazanet C. 1996 Enrichissement des bases de données géographiques : analyse de la géométrie des objets linéaires pour la généralisation cartographique (application au routes). PhD Thesis, University Marne-la-Vallée, France

Quinlan J.R. 1992. C4.5 : Programs for Machine Learning. Morgan Kaufmann Series in Machine Learning.

Reichenbacher T., 1995. Knowledge acquisition in map generalization using interactive systems and machine learning. ACI'95 proceedings, Vol 2, pp 2221-2230

Ruas A. 1998. First results on the OEEPE test on generalisation. OEEPE Newsletter (1), pp. 5-10.

Rumelhart D.E., Hinton G.E., Williams R.J. 1986. Learning internal representations by error propagation. In D.E. Rumelhart et J.L. McClelland (Eds) Parallel Distributed Processing (Vol. 1). Cambridge, MA: MIT Press.

Weibel R., Keller S. & Reichenbacher T. 1995 Overcoming the Knowledge Acquisition Bottleneck in Map Generalization : the Role of Interactive Systems and Computational Intelligence. COSIT'95 pp. 139-156