To appear in *Proceedings of the Eighteenth International Conference on Machine Learning ICML-2001*.

Pairwise Comparison of Hypotheses in Evolutionary Learning

Krzysztof Krawiec KRAWIEC@CS.PUT.POZNAN.PL
Institute of Computing Science, Poznan University of Technology, Piotrowo 3A, 60965 Poznan, Poland

Abstract

This paper investigates the use of evolutionary algorithms for the search of hypothesis space in machine learning tasks. As opposed to the common scalar evaluation function imposing a *complete* order onto the hypothesis space, we propose genetic search incorporating pairwise comparison of hypotheses. Particularly, we allow incomparability of hypotheses, what implies a *partial* order in the hypothesis space. We claim that such an extension protects the 'interesting' hypotheses from being discarded in the search process, and thus increases the diversity of the population, allowing better exploration of the solution space. As a result it is more probable to reach hypotheses with good predictive accuracy. This supposition has been positively verified in an extensive comparative experiment of evolutionary visual learning concerning the recognition of handwritten characters.

1. Introduction

Evolutionary computation (De Jong, 1975; Holland, 1975) has been used in machine learning for quite a long time (Mitchell, 1996). Now it is recognized as a useful approach or even as one of its paradigms (Langley, 1996; Mitchell, 1997, Chapter 9). It is highly appreciated due to its ability to perform global parallel search of the solution space with low probability of getting stuck in local minima. Its most renowned applications include feature selection (Yang & Honavar; 1998), feature construction, and concept induction (DeJong, Spears, & Gordon; 1993; Goldberg, 1989). In this paper, we focus on the last of the aforementioned tasks, with solutions corresponding to hypotheses; from now on, these terms will be used interchangeably.

Like all metaheuristics, evolutionary algorithm needs an evaluation (fitness) function to guide the search. In the machine learning framework, this evaluation is commonly based on an estimate of predictive accuracy of the hypothesis (Langley, 1996), eventually including the hypothesis-size factor to prevent overfitting, introduced explicitly or in a more sophisticated way (as it is, for instance, in the minimum description length by Rissanen (1983)).

Such a scalar evaluation function imposes a *complete order* onto the hypothesis space. As a result, it is assumed that all hypotheses are *comparable* with respect to their predictive ability and, given a pair of them, it is always possible to point the better one, unless they have the same evaluation.

In this contribution we argue that the above statement is in general not valid. The following section of this paper presents some negative consequences of forcing the hypotheses to be always comparable. Then, in Section 3 we propose the replacement the scalar evaluation by the pairwise comparison of hypotheses. Section 4 describes how it is possible to embed this idea into learning based on evolutionary search of the hypothesis space. Section 5 contains the description of an experimental evaluation of the approach on the visual learning task of handwritten character recognition, followed by conclusions.

2. The Need for Incomparability of Hypotheses

The main claim of this paper is that scalar evaluation of hypotheses implies the complete order of solutions, which does not reflect well the structure of the hypothesis space. A numerical evaluation function, like for instance the accuracy of classification, reflects well the utility of particular hypothesis, however, it reveals some shortcomings when used for hypothesis comparison. This is mainly due to the fact that such measures have by definition aggregating and compensatory character (Vincke, 1992). They may yield similar or even equal values for very different hypotheses.

We suggest that when the considered hypotheses 'behave' in a significantly different way, for instance they produce different outcomes on particular parts of the decision space, we should allow them to be *incomparable*. The need of such incomparability grows with the dissimilarity between the compared hypotheses and becomes especially important when their scalar evaluations are relatively close.

Let us illustrate this problem with the following example. For a hypothesis h, let $C(h)$ denote the subset of examples from the training set T that are classified correctly by h ($C(h) \subseteq T$). Then, let the hypotheses be evaluated by means

of scalar evaluation function f, computing the accuracy of classification of h on T ($f(h) = |C(h)| / |T|$). Let us consider three hypotheses, a, b, and c, for which $|C(a)| > |C(b)| = |C(c)|$. Thus, with respect to f, hypotheses b and c are of the same quality and are worse than a. This evaluation method cannot differentiate the pairs of hypotheses (a,b) and (a,c).

The above reasoning ignores the mutual relations between $C(a)$, $C(b)$ and $C(c)$. If, for instance, $C(b) \subset C(a)$, we probably would not doubt the superiority of a over b. But what about the relation between a and c, assumed that $C(c) \not\subset C(a)$ and $|C(c) \cap C(a)| << |C(a)|$? In such a case, although a classifies correctly more examples than c, there is a remarkable subset of examples ($C(c) \setminus C(a)$), which it does not cope with, while they are successfully classified by c. Thus, superiority of a over c is rather questionable. Moreover, if also $C(a) \not\subset C(c)$, the question concerning mutual relation between a and c should probably remain without answer, leading us to the concept of hypothesis incomparability. ■

Scalar evaluation ignores the issue illustrated in the above example and forces the hypotheses to be always comparable. As a result, some novel and 'interesting' hypotheses may be discarded in the search due to their minor evaluation. The loss of such solutions may influence significantly the effectiveness of the search. In the further processing, they could explore some new, hitherto unrevealed parts of the hypothesis space and attain better evaluation than the solutions that won the scalar competition.

Note that the above observation is valid for any machine learning algorithm (or other heuristics) that explicitly evaluates and compares hypotheses. However, it is of special importance in genetic programming, where the offspring solutions have often low fitness due to the destructive nature of mutation and recombination operators.

3. Pairwise Comparison of Hypotheses

3.1 From Incomparability to Outranking

In this part we will present an alternative approach, devoid of the shortcomings discussed in the previous section. We showed that it is reasonable in some cases to allow hypotheses to remain incomparable. This suggests us that we should move on from the *functional* method of hypothesis evaluation to the *relational* one. The resulting structure of complete order is very popular in, for instance, relational approaches to multiple criteria decision aid, where it is often being described by means of a binary *outranking relation*[1], denoted thereafter by '≥' (see, for instance, Chapter 5 of (Vincke, 1992)). According to the definition, for a pair a,b of solutions, $a \geq b$ should express the fact that a is *at least as good* as b. Then, one of the following cases is possible:

- a is indiscernible with b ($a \geq b$ and $b \geq a$), or
- a is strictly better than b ($a \geq b$ and not $b \geq a$), or
- b is strictly better than a ($b \geq a$ and not $a \geq b$), or
- a and b are incomparable (neither $a \geq b$ nor $b \geq a$).

Partial order has a natural graphical representation of a directed graph. The nodes of outranking graph correspond to hypotheses, whereas arcs express the outranking. Particularly, the 'best' solutions match the initial (predecessor-free) nodes. Note also that outranking is in general reflexive and non-symmetric.

3.2 Hypothesis Outranking

3.2.1 Preliminary Assumptions

The outranking relation ≥ may be defined in many different ways. Generally, we could consider here the definitions based on the *representation* (*form*) of the hypothesis (like in the well-known Candidate-Elimination by Mitchell (1997)) or the definitions based on the *functioning* (behavior) of the hypothesis on the training set. Obviously, the partial orders imposed on the hypothesis space by both these types of outranking are different. The approach presented here implements the latter case, which has an advantage of not making any assumption about knowledge representation used by the induction algorithm.

Particularly, we focus on the paradigm of supervised learning from examples, the one used most often in the real-world applications. For the sake of simplicity, however without loss of generality, we also limit our considerations to the *binary* (two-class) *classification problems* (the positive and negative decision classes).

3.2.2 Definition of Hypothesis Outranking Based on Training Set Classification

The presence of the training set T in learning from examples paradigm allows us to define the outranking in terms of sets. However, instead of using an aggregating measure like accuracy of classification, we go more into detail and analyze the behavior of the hypotheses on particular instances from the training set.

The example presented in Section 2 shows that it seems to be useful to refer here to the set difference of the sets of instances properly classified by the considered hypotheses a and b ($C(a)$ and $C(b)$, respectively). In particular, the more examples belong to $C(b) \setminus C(a)$, the less likely is the outranking $a \geq b$.

An outranking relation that follows this intuition may be reasonably defined in several different ways. An elegant idea could be to refer here to the notion of set inclusion

[1] Formally, an outranking relation induces partial *pre*order, as it permits indiscernibility.

grade (Dubois & Prade, 1980), or fuzzy inclusion relation (Dubois & Prade, 2000, Section 2.4). However, as the goal of this research was to investigate the issues of partial order of hypotheses and incomparability, it was undesirable at that point to apply sophisticated and parameterized relations. Thus, we employ here the crisp inclusion of sets and define the outranking of a over b as follows:

$$a \geq b \Leftrightarrow C(b) \subseteq C(a). \quad (1)$$

This definition states that a hypothesis a is at least as good as a hypothesis b iff a classifiers correctly at least all the examples which are classified correctly by b. Note that the outranking of a over b may be disabled by just a single training example x $(C(b) \setminus C(a) = \{x\})$. This sensitivity is surely a weak point, we decided however to pay such a price for keeping this study simple and non-parametric. For real-world implementations a more sophisticated definition should be engaged. On the other hand, outranking relation as defined in (1) is transitive; this property, in general not required, may be advantageous when computing some of the entities introduced further in the paper.

4. GPPO - Embedding Pairwise Hypothesis Comparison into Evolutionary Learning

Genetic Programming using Partial Order of solutions, referred hereafter to as GPPO, requires redefinition of some parts of the evolutionary search procedure (Goldberg, 1989). This applies to the selection process, to the maintenance of the set of best solutions found so far, and to the interpretation of the final result. The following subsections describe these changes.

4.1 Outranking-based Selection of Hypotheses

Selection is the central step of any evolutionary algorithm procedure and consists in choosing the set of parent solutions P^* (often referred to as mating pool) from the population P evolved in considered generation of evolutionary search. In the outranking-based selection process we have to take into account the potential presence of hypotheses incomparability. In the preliminary research (Krawiec, 2001), we tried to extend for this purpose the popular tournament selection scheme (Goldberg, Deb, & Korb, 1991). Unfortunately, that approach did not yield satisfactory results in experimental evaluation, probably due to the fact, that, as tournaments for incomparable hypotheses remain unsettled, the selection pressure decreases.

Thus, in GPPO we take an alternate way and start with computing the subset $N(P)$ of non-outranked solutions from P, i.e.

$$N(P) = \{h \in P: \neg\exists\, h' \in P: h' \geq h\}. \quad (2)$$

This definition is straightforward, but troublesome in the sense that we cannot directly control the cardinality of $N(P)$. In practice $N(P)$ usually contains a small fraction of P, nevertheless in extreme cases it can be empty may or encompass all the individuals from P. This is contradictory to a reasonable assumption that we should preserve constant size of the population (at least approximately).

Thus, the approach described in this paper combines the standard tournament selection with the outranking-based selection in the following steps:

1. $P^* \leftarrow N(P)$
2. If $|P^*|$ is smaller than a predefined fraction of the population size $\alpha|P|$, $\alpha \in (0,1\rangle$, the solutions in P^* are 'cloned' to reach that size.
3. The missing part of the mating pool $(P \setminus P^*)$ is filled with solutions obtained by means of the standard tournament selection on P.

The α parameter controls the penetration of the mating pool by the non-outranked solutions P^* and ensures that this influence is relatively constant, no matter what the actual size of P^* is.

4.2 Outranking-based Maintenance of Best Solutions

The presence of solution incomparability implies also some changes in the way we should keep track of the best solutions found in the evolution process. We have to be prepared to face many 'leaders' in the population and maintain the set of all non-outranked solutions found during the search, denoted further by N^*. Starting with $N^* = \emptyset$, the update of N^* for consecutive generations requires the following operation in GPPO:

$$N^* \leftarrow N(P \cup N^*). \quad (3)$$

4.3 Utilization of the Best Solutions

The set N^* of non-outranked solutions resulting from the completed evolution process may be used in a usual way, i.e. one can select from it the best solution with respect to the scalar evaluation function and treat it as the final outcome of the learning process. This was the method applied in the forthcoming case study (Section 5), as it ensures the comparability of results with the standard genetic programming.

However, the mutual incomparability of solutions from N^* suggests that they are significantly different in terms of particular definition of outranking. In the case of training set performance-based outranking (1), that means different performance in various parts of the decision space. Therefore, it seems reasonable to benefit from the knowledge acquired by (potentially all) solutions from N^*. A natural approach here is to refer to the methodology of meta-classifiers, which offers a broad scope of methods for combining classifiers, usually aiming at boosting the accuracy of classification (see, for instance, (Chan &

Stolfo, 1993)). We plan to devote a part of our future research on GPPO to this topic.

4.4 Remarks and Related Research

Methods of improving the exploration of the solution space (or maintenance of diversity) appear in evolutionary computation under the name of *niching* and *multimodal* genetic search. Some of those methods operate on the solution level and base the selection on a random, usually small sample of the population (e.g. tournament selection by Goldberg, Deb, and Korb (1991), or restricted tournament selection by Harik (1995)). Others use a more careful pairing of selected parents (Mitchell, 1997, p. 259). Yet another approaches rely on a more intermediate influence and modify the evaluation scheme, penalizing the solutions for 'crowding' in the same parts of the solution space, as in the popular fitness sharing by Goldberg and Richardson (1987) or sequential niche technique by Beasley, Bull and Martin (1993). In particular, niches may be maintained during the entire evolution process (parallelly) or only temporarily (sequentially); Mahfoud (1995) provided an interesting comparison of these groups of methods.

The specificity of GPPO method in comparison to the aforementioned approaches consists in the following features:

- GPPO supports niching in an explicit way, by means of the concept of outranking. In particular, GPPO does not require any extra distance metric in the search space (whereas, for instance, many fitness sharing methods do).
- GPPO carries out the search without making any reference to the scalar evaluation function, which, as pointed out in Section 2, has some drawbacks due to its aggregative character in machine learning tasks. Thus, GPPO is more than a mere niching method; it is rather a variety of evolutionary search procedure that maintains the set of mutually non-outranking solutions during the search process.
- GPPO makes direct use of the detailed and very basic information on performance of the solution on particular training examples. Thus, the comparisons of individuals in the genetic GPPO search are tied very closely to the mutual relationships of hypotheses in the hypothesis space.

A reader familiar with the topic may notice some analogies between GPPO and multiobjective genetic search and optimization (Schaffer, 1985; Van Veldhuizen, 1989). However, the multiobjective approach refers to the *dominance* relation, which assumes the existence of a multidimensional space spanned over a finite number of ordered objectives. The concept of outranking presented in Section 3 and, in particular, the outranking definition (1) used in this paper, do not assume an existence of such a space. The incomparability of solutions in dominance-based methodology is a consequence of the presence of multiple dimensions (objectives) and the tradeoffs between them, whereas in our case of outranking we do not explicitly define such dimensions.

5. Genetic Programming using Partial Order of Hypotheses in Visual Learning

5.1 Genetic Programming for Visual Learning

The proposed idea has been adopted in genetic programming-based *visual learning*, which was the subject of our previous research (Krawiec & Slowinski, 1997; Krawiec, 2000; Krawiec, 2001). The goal of the learning process is here to induce the complete pattern analysis and recognition program, without explicit division into stages of feature extraction and reasoning.

As the experimental test bed for the approach, we chose the problem of off-line handwritten character recognition. This task is often referred to in the literature due to its wide scope of real-world applications. The methods proposed in literature incorporate statistics, structural and syntactic methodology, sophisticated neural networks, or ad hoc feature extraction procedures, to mention only a few (for review, see (LeCun, Jackel, Bottou, Brunot, et al. 1995)). The genetic programming approach presented in this paper cannot be univocally classified into any of these categories, combining the elasticity and learning capability of adaptive systems with comprehensibility of symbolic knowledge representation.

5.2 Related Research on GP-based Learning

Genetic programming (GP) proposed by Koza (1994) is reported to be very effective in solving a broad scope of learning and optimization problems. The major difference in comparison with standard genetic algorithms is here the more sophisticated solution representation (usually LISP-like expressions or programs), which gives more elasticity, but requires also more sophisticated recombination operators (see Section 5.4.3).

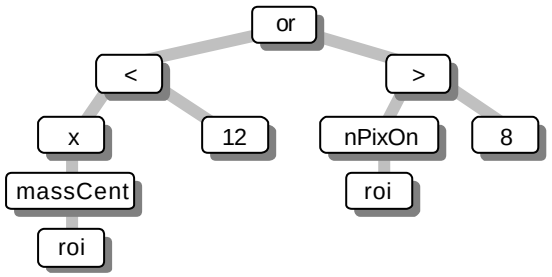
A remarkable part of research on evolutionary algorithms concerns machine learning (see (Mitchell, 1996; Mitchell 1997) for review). There are also some results in application of evolutionary algorithms for image processing and analysis (e.g. (Bala, De Jong, & Pachowicz, 1994). However, there are relatively few, which try to combine both these aspects and refer to the visual learning, understood as the search for pattern recognition programs (Johnson, 1995; Teller & Veloso, 1995; Poli, 1996; Krawiec, 1997; Krawiec, 2000). Only a small fraction of research aims at inducing the complete image analysis program based on training examples, which direction is in our opinion very promising and universal.

5.3 Representation of Image Analysis Programs

In conventional approaches to image analysis and interpretation, the processing is usually split into the feature extraction and reasoning (Gonzalez & Woods, 1992). The reasoning is based on the feature vector provided by image analysis methods and usually employs (statistical or machine learning) classifier. Such a separation of the reasoning process from the image analysis and feature extraction suffers from several drawbacks (Krawiec, 2000). The use of GP for image analysis and interpretation was motivated by this shortcoming and is aimed at expressing the complete program of image analysis and interpretation without the need for an external classifier. The major advantage of such setting is that the training process is no more limited to the decision space predefined by a human expert, but encompasses also the search for an appropriate image representation.

In this case study, the search takes place in the space of hypotheses being pattern recognition procedures formulated in a specialized language called GPVIS (Krawiec 2000). GPVIS is an image analysis-oriented language encompassing a set of operators responsible for simple feature extraction, region-of-interest selection, and arithmetic and logic operators. The programs performing image analysis and recognition are tree-like GPVIS expressions composed of such operations.

To give the reader a general idea what GPVIS is like, Figure 1 presents a simple example of image analysis program formulated in that language. Its interpretation is as follows: *if* the *x* coordinate of the mass center of the contents of rectangular region of interest (roi) limited by upper left corner (19,8) and lower right corner (2,4) is less than 12 *or* there are more than 8 pixels in the 'on' state in another region of interest (the right branch of the tree), *then* the return value is *true*. This returned value could be then further processed to yield binary class assignment, as in the case study described in this section.




```
(or (< (x (massCent (roi 19 8 2 4))) 12)
    (> (nPixOn (roi 7 23 10 18)) 8))
```

Figure 1. Tree-like and LISP-like representations of an exemplary solution formulated in GPVIS language (numerical values omitted in the tree).

5.4 The Experiment

5.4.1 The Goal of the Experiment

The goal of the computational experiment was to compare the performance of the proposed genetic programming with partial order of solutions (GPPO) with the 'plain' genetic programming (GP). The main subject of comparison was the accuracy of classification of the best evolved solutions (hypotheses) on the training and test set.

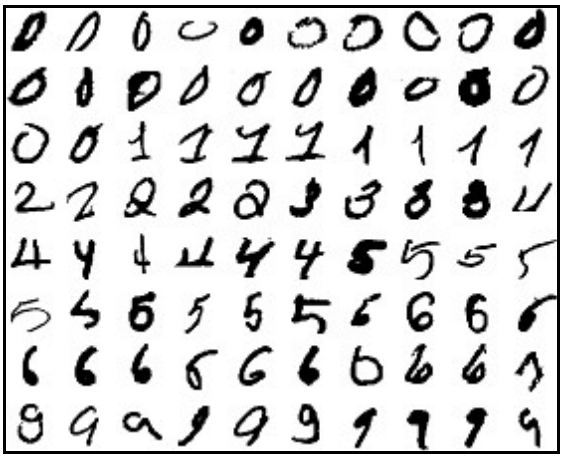


Figure 2. Selected difficult examples from the MNIST database.

5.4.2 Image Data

The source of images was the MNIST database of handwritten digits made available by LeCun et al. (1995). The database contains 70,000 digits written by approx. 250 persons (students and clerks), each represented by a 28×28 matrix of gray level pixels (Fig. 2). Characters are centered and scaled with respect to their horizontal and vertical dimensions, however, not 'deskewed'.

5.4.3 Experiment Design

To ensure a statistically strong support for the results, a extensive computational experiment with different training and test data has been carried out. First of all, instead of considering the complete ten-class digit recognition problem we ran a separate series of experiments for each pair of ten digit classes; there were $10 \times 9/2 = 45$ of them. Each such series consisted of three simulations. Corresponding GP and GPPO runs started from the same initial population. Thus, the results presented hereafter summarize 135 pairs of genetic runs.

The experiments have been also carefully prepared and carried out so as to ensure credible comparability of results. The particular GP and GPPO runs were 'paired' in the sense that they started from the same initial population and used the same training and test sets as well

as the values of parameters. The most important of them were set as follows: population size: 200; probability of mutation: .05; maximal number of generations: 100; training set size: 100 instances (50 images per class, randomly selected from the training subset of the MNIST database); tournament size: 5 (Goldberg et al., 1991), and $\alpha = .5$ (see Section 4.1).

In each generation, half of the population was retained unchanged, whereas the other fifty percent underwent modifications. The GP runs used the standard tournament selection based on scalar fitness function, whereas GPPO runs followed the selection procedure described in Section 4.1. Then, the offspring were created by means of the crossover operator, which randomly selects subexpressions (corresponding to subtrees in the graphical representation shown in Fig. 1) in the two parent solutions and exchanges them. The mutation operator applied to a solution randomly selects a subexpression and replaces it by other subexpression generated at random. In these operations the so-called *strong typing* principle must be obeyed (Koza, 1994).

Special precautions have been taken to prevent overfitting of hypotheses to the training data. In the GP case, the scalar fitness function was extended by additional penalty term implementing parsimony pressure. Particularly, solutions growing over 100 terms were linearly penalized with the evaluation decreasing to 0 when the threshold of 200 terms is reached. In the GPPO approach, a solution composed of 100 or more terms was always outranked, no matter how it performed on the training data.

5.4.4 PRESENTATION OF RESULTS

Table 1 presents the comparison of the best solutions (see Section 4.3) obtained in GP and GPPO runs. Table rows reflect consecutive stages of the evolution process (selected generations). Each row summarizes the comparison of 135 paired GP and GPPO runs (see Section 5.4.3). The description includes:

- the number of pairs of GP and GPPO runs (per total of 135) for which the best solution[2] evolved in GPPO yielded strictly better accuracy of classification on the training set than the best one obtained from 'plain' GP ('#GPPO BETTER'),
- the average increase of accuracy of classification of GPPO in comparison to GP ('AVERAGE INCREASE'),
- the false reject probability of Wilcoxon matched pairs signed rank test ('FALSE POSITIVE PROBABILITY'); the test takes into account the relative magnitude of differences in GP and GPPO accuracy.

[2] For both GP and GPPO, the term 'best' in this context refers to the best solution found in the evolution process, with respect to the *scalar* evaluation function, i.e. the accuracy of classification (see Section 4.3).

Table 2 presents the summary of the performance of the same solutions as in Table 1 when evaluated on an independent test set. The test set for each task contains 1600 objects, i.e. 800 images for both positive and negative classes, selected randomly from the testing part of the MNIST database. Note that the training (fitness) set and testing set are independent in a strong sense, i.e. contain digits written by another people (LeCun et al., 1995).

The tables do not refer directly to the (average) accuracy of classification, as it would not make much sense due to the heterogeneity of particular experiments (different pairs of decision classes). However, to give the reader an idea about the absolute performances of hypotheses elaborated by both algorithms, we provide the average accuracy of classification at the end of evolutionary runs (training and testing set, respectively): 90.3±6.0% and 85.2±10.2% for GP, 92.2±4.9% and 87.7±7.4% for GPPO (standard deviations included).

Table 1. Comparison of the best solutions evolved in GP and GPPO runs with respect to the accuracy of classification on the training set.

GENERATION	#GPPO BETTER	AVERAGE INCRASE [%]	FALSE POSITIVE PROBABILITY
20	74/135	0.55	.8681
40	76/135	0.93	.2880
60	89/135	1.67	.0085
80	88/135	1.46	.0096
100	105/135	1.97	.0002

Table 2. Comparison of the best solutions evolved in GP and GPPO runs with respect to the accuracy of classification on the test set.

GENERATION	#GPPO BETTER	AVERAGE INCRASE [%]	FALSE POSITIVE PROBABILITY
20	69/135	-0.02	.6234
40	82/135	1.68	.1093
60	86/135	1.92	.0461
80	89/135	1.69	.0325
100	92/135	2.63	.0061

6. Conclusions and Future Research Directions

The main qualitative result obtained in the experiment is that evolutionary search taking into account the partial order of solutions and allowing hypothesis incomparability (GPPO) outperforms the 'plain' genetic programming (GP) on average. The longer the time devoted to the search, the more best solutions obtained by means of GPPO outperform that obtained by GP. Tables 1 and 2 show that, as both algorithms proceed, the increase (difference) of accuracy of classification of best GPPO solutions grows in comparison to the best GP solutions. Starting from generation 60, this difference becomes statistically significant at the .05 level; at the end of the runs the probability of false reject error is lower than .01. Importantly, this applies to the training set as well as to the test set. The GPPO hypotheses (classifiers) are not only superior on the training set, but also reveal better predictive ability. The average 2.6% gain on accuracy of classification seems to be attractive, remembering the complexity of the visual learning task and the fact, that the accuracies provided by both the methods at the end of runs are close to 100%.

The result not shown in the tables is that the obtained GPPO solutions have similar size to those reached by GP (we define the solution size as the total number of GPVIS subexpressions; see Section 5.3). As far as time factor is concerned, although pairwise comparison of solutions introduces obviously an extra overhead, that additional cost does not exceed on average 10% of the total computing time.

The general conclusion of this work is that it is worthwhile to control the search of the hypothesis space by means of an incomparability-allowing, pairwise comparison relation. Such evaluation method protects the novel solutions from being discarded in the search process, even if they exhibit minor fitness in scalar terms. The more abstract conclusion could be formulated as follows: in the presence of an order, we do not have to look for an intermediation of numbers.

It seems also that such an observation is not limited to evolutionary search and could be generalized to other machine learning inducers, especially those, which explicitly evaluate and compare the hypotheses in the context of training data.

The proposed approach to evolutionary learning has the advantage of being independent of the knowledge representation. From the viewpoint of information theory, the method makes use of the information concerning th performance of the hypothesis on training sense in a much more extent than the scalar evaluation.

Further work on this approach may concern different aspects, some of them are however of special importance. In particular, it seems to be interesting to consider the more sophisticated definitions of hypothesis outranking, mentioned in Section 3.2.2, which should be less sensitive to the classification of particular examples. Then, as suggested in Section 4.3, a useful extension of the approach could be to combine the non-outranked hypotheses to form a meta-classifier, for instance by simple or weighted voting. To improve further the results and to speed up the learning we plan also to introduce the incremental growth of the training set (called *incremental evaluation* by Langley (1996), p. 60). And, last but not least, there is a need for a more extensive computational experiment concerning various (not necessarily visual) tasks to evaluate the usefulness of the GPPO method in a broader context.

Acknowledgements

The author would like to thank Jerzy Stefanowski for valuable comments and remarks and Yann LeCun for making the MNIST database of handwritten digits available to the public. This work was supported from the KBN research grant no. 8T11F 006 19.

References

Bala, J.W., De Jong, K.A., Pachowicz, P.W. (1994) Multistrategy learning from engineering data by integrating inductive generalization and genetic algorithms. In R.S. Michalski, G. Tecuci, *Machine learning. A multistrategy approach. Volume IV*. San Francisco: Morgan Kaufmann, 471-487.

Beasley, D., Bull, D.R., & Martin, R.R. (1993). A Sequential Niche Technique forMultimodal Function Optimization. *Evolutionary Computation* 1 (2), 101-125.

Chan, P.K., & Stolfo, S.J. (1993). Experiments on multistrategy learning by meta-learning. *Proceedings of the Second International Conference on Information and Knowledge Management*.

De Jong, K.A. (1975). An analysis of the behavior of a class of genetic adaptive systems. Doctoral dissertation, University of Michigan, Ann Arbor.

De Jong, K.A., Spears, W.M., & Gordon, D.F. (1993). Using genetic algorithms for concept learning. *Machine Learning*, *13*, 161-188.

Dubois, D., & Prade, H. (1980). *Fuzzy sets and systems. theory and applications.* New York: Academic Press.

Dubois, D., & Prade, H. (2000). *Fundamentals of fuzzy sets*. Boston: Kluwer Academic.

Goldberg, D. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading: Addison-Wesley.

Goldberg, D., Deb, K., & Korb, B., (1991). Do not worry, be messy. *Proceedings of the Fourth International Conference on Genetic Algorithms* (pp. 24-30). San Mateo: Morgan Kaufmann.

Goldberg, D., & Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, 41-49.

Gonzalez, R.C., Woods, R.E. (1992). *Digital image processing*. Reading: Addison-Wesley.

Harik, G. (1995). Finding multimodal solutions using restricted tournament selection. In L. J. Eshelman (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms* (pp. 24-31). San Francisco: Morgan Kaufmann.

Holland, J.H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Johnson, M.P. (1995). *Evolving visual routines*. Master's Thesis, Massachusetts Institute of Technology.

Koza, J.R. (1994). *Genetic programming - 2*. Cambridge: MIT Press.

Krawiec, K., & Slowinski, R. (1997). Learning discriminating descriptions from images. *Proceedings of the Sixth International Symposium 'Intelligent Information Systems'* (pp. 118-127), Warsaw: IPIPAN Press.

Krawiec, K. (2000). *Constructive induction in picture-based decision support*. Doctoral dissertation, Institute of Computing Science, Poznan University of Technology, Poznan.

Krawiec, K. (2001). *Genetic programming using partial order of solutions for pattern recognition tasks*. Unpublished manuscript. Institute of Computing Science, Poznan University of Technology, Poznan.

Langley, P. (1996). *Elements of machine learning*. San Francisco: Morgan Kaufmann.

LeCun, Y., Jackel, L. D., Bottou, L., Brunot, A., et al. (1995). Comparison of learning algorithms for handwritten digit recognition. *International Conference on Artificial Neural Networks* (pp. 53-60).

Mahfoud, S.W. (1995). A Comparison of Parallel and Sequential Niching Methods. In L.J. Eshelman (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms* (pp. 136-143). San Mateo: Morgan Kaufmann.

Mitchell, T.M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.

Mitchell, T.M. (1997). *Machine learning*. New York: McGraw-Hill.

Poli, R. (1996). *Genetic programming for image analysis*, (Technical Report CSRP-96-1). The University of Birmingham.

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *The Annals of Statistics, 11*, 416-431.

Schaffer, J.D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. *Proceedings of the First International Conference on Genetic Algorithms and their Applications*. Hillsdale: Lawrence Erlbaum Associates.

Teller, A., & Veloso, M. (1995). A controlled experiment: evolution for learning difficult image classification. *Lecture Notes in Computer Science, Vol. 990*, 165-185.

Vafaie, H., & Imam, I.F. (1994). Feature selection methods: genetic algorithms vs. greedy-like search. *Proceedings of International Conference on Fuzzy and Intelligent Control Systems*.

Van Veldhuizen, D.A. (1999). *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. Doctoral dissertation, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

Vincke, P. (1992). *Multicriteria decision-aid*. New York: John Wiley & Sons.

Yang, J., & Honavar, V. (1998). Feature subset selection using a genetic algorithm. In H. Motoda, & H. Liu (Eds.), *Feature extraction, construction, and subset selection: A data mining perspective*. New York: Kluwer Academic.