Universität Hamburg
Department Informatik
Knowledge Technology, WTM

Bio-inspired algorithm for feature selection in classification

Seminar Paper

Bio-inspired Artificial Intelligence

Paul Anton
Matr.Nr. 6765670
5anton@informatik.uni-hamburg.de

01.02.2017

Abstract

This paper is focusing on the suitability of genetic algorithms for implementing feature selection in the context of classification tasks. The underlying assumption is that using smaller feature subsets in classification might have a positive impact on both the accuracy and the running time of the classifier. The presented approach is using Genetic Algorithms as the search tool through the space of feature subsets and K Nearest Neighbors classifier for evaluation.

Contents

1 Introduction

In the context of any machine learning endeavor, the implementation of a successful learning task (be it supervised or unsupervised) is highly dependent upon the properties of the input data set. Namely, the most obvious quantitative measure that accounts for the so called "curse of dimensionality" is represented by the number of features representing each training example. While literature reports that classical data mining and pattern classification problems started accounting for more than 40 features towards the end of the previous millenium, modern machine learning efforts account for up to tens of thousands of features [3]. However, the relevant features are unknown a priori, and removing those that do not add any information towards the target concept can have a positive impact on the running time of a learning algorithm, as well as on the generalization capability of the model [1]. It is the purpose of feature selection to identify the minimal subset of features that are most relevant to the prediction outcome. With vast research efforts dating as early as the 1970's and spanning across multiple fields (data mining, pattern recognition, machine learning), feature selection has become one of the most important data preprocessing techniques [8]. The benefits of the method on data visualization, reduction of storage & processing requirements as well as its impact on the speed of learning algorithms turned it into a valuable tool for many application areas such as text categorization, image retrieval or data mining [8], [7].

Designing one feature selection algorithm takes into account both data set characteristics (data types, data size, noise) and the desired learning approach. It is the scope of the paper at hand to assess the suitability of genetic algorithms for feature selection applied in the context of a classification task. Based on the implementations proposed by Siedlecki et al. [12] and Huang [5], the remainder of the paper will present the formal definition of feature selection and Genetic Algorithms (Section 2), the implementation details (Section 3), the evaluation of feature selection's impact on a specific learning algorithm (Section 4), as well as the concluding remarks (Section 5).

2 Background Information

2.1 Feature selection

The task of feature selection algorithms is to perform a search through the space of feature subsets and return the best one according to a predefined evaluation criterion. Depending on this evaluation criterion, the problem can be formally defined in two ways [12]:

- as an unconstrained combinatorial optimization problem - when the search is guided by the error rate of a classifier and needs to find the feature subset with the smallest error

- as a constrained combinatorial optimization problem - when the search criterion is the number of features in a subset and the result is the smallest subset with the error below a given threshold

Regardless of the choice above, only a small size of the initial feature set N allows for an exhaustive search through the 2^N subsets and the natural option for most of the existing research on the topic is to employ heuristic search strategies [1], [4], [8].

Another consensus in literature is reached with regards to the separation of the feature selection process into four main steps [1], [4], [8], [9]:

1. subset generation - creating subsets of features for evaluation, according to the employed search strategy
2. subset evaluation - triggering the replacement of the current best subset with a new one, in case it is found to be better
3. stopping criterion - preventing an exhaustive search and is usually influenced by the choice of the the first two methods
4. subset validation - asserting the validity of the selected feature subset

Without going into details about the variety of directions possible for each of the steps presented above, it is worth mentioning that subset evaluation is the one that separates feature selection algorithms into two main classes: the so called "filter based methods" and "wrapper based methods". A filter approach assumes no dependence on a specific learning algorithm, but instead uses the characteristics of the data and statistical methods to evaluate a feature subset (e.g. relevance index built from correlation coefficients) [9]. This method is praised in literature due to its fast execution times and its adaptability to any learning algorithm [4]. On the other hand, a wrapper method is tightly coupled to a predefined learning algorithm, and uses its performance as the evaluation function (e.g. accuracy). With the cost of computational resources and re-execution necessity for every new learning algorithm, this method was often reported to provide better results than filter methods [8], [4]. A comparative discussion of the two methods being outside the scope of the paper, it is worth mentioning that the discussed implementation belongs to the second category.

2.2 Genetic Algorithms

Inspired by biological evolution and the principle of natural selection, Genetic Algorithms are a known approach for solving optimization problems. Unlike traditional search techniques, they employ randomness and knowledge from previous iterations to evaluate and develop a population of solutions. Randomness is achieved by population modification, which together with the parallelism of the search, make it possible to converge to a global optimum even in complex landscapes as those of NP-hard problems [12], [13].

The entities of interest of a genetic algorithm are known as chromosomes, which can be visualized as distinct points in the search space and represent possible solutions [13]. With each iteration, a finite set of solutions, the population, is filtered, developed and evaluated according to an objective function [13]. At first, the selection mechanism (inspired from the survival of the fittest principle) picks the individuals that are going to generate offspring in the new generation. Assigned in a subsequent step, the fitness of each solution is needed by the mechanism to identify those with a high chance of survival. As a next step in the development of a population, new solutions are derived from the existing ones by means of genetic operators such as crossover and mutation [13]. In case of crossover (process that can be seen as originating from mating in natural evolution), two chromosomes exchange genetic material in a probabilistic manner, with the expectancy of increasing the quality of the offspring, as a synthesis of their parents' information. On the other hand, mutation performs on a single chromosome and has the purpose of restoring lost genetic material and therefore increasing the variability of the population. The probability with which both of the above operators are applied to a population is determined by two distinct, predefined parameters known as the crossover rate and the mutation rate respectively. As a last step of one iteration, the solutions are evaluated according to an objective function, indicating their suitability for the problem. The fitness measure, as a normalized value of the objective function is to be used in the next iteration in the selection process. The algorithm runs until a certain termination criteria has been met, such as a specific number of processed generations, or having found a solution that is good enough for the problem [13].

3 Implementation

3.1 Problem representation and design decisions

When attempting to solve a particular problem by means of a genetic algorithm, the principal design decisions must account for the encoding of the candidate solutions, the objective function used for ranking and the definition of genetic operators and their running parameters [6], [10]. It is the purpose of this section to outline the rationale of the aforementioned and their representation in the context of the feature selection problem.

Solution encoding

As often encountered in literature ([12],[14]), it is assumed that the search for the best feature subset is performed across a randomly initialized population of individuals. Accordingly, each solution is represented by a d-dimensional binary vector, whose values indicate the presence of features: only a value of 1 on the k-th position will ensure that the k-th feature will be used in the classification.

Specific for wrapper based feature selection methods, each feature subset in the population is to be assessed on the basis of the ability of a classifier to discriminate

the classes in the feature space represented by the evaluated subset. The underlying classification is performed by a K Nearest Neighbors algorithm, whose performance will affect the fitness of the feature subset and its chances of surviving through the next generations.

To the classification accuracy another criteria for evaluation was added, respectively the size of the feature subset, which has a smaller impact on the fitness. This formulation has been implemented before [5] by defining weights for the classification accuracy w_a and for the subset dimension w_f. Thus, the feature subsets that will rank higher are those presenting a high classification accuracy and a small size of the feature subset.

$$f = w_a \times accuracy\_score + \frac{w_f}{\sum F_i} \tag{1}$$

Genetic operators

The advantage of the aforementioned solution encoding is that the standard genetic operators could be applied without further modifications. Therefore, a 1-point crossover mechanism was used for exchanging genetic information to generate new solutions from old ones. Also, mutation operated at a single chromosome level, by (re)setting the usage of features according to a predefined low probability.

4 Evaluation

4.1 Experimental setup

The running parameters of the genetic algorithm have been established empirically, as follows: the crossover rate was set to a value of 0.7, mutation rate of 0.09, with a population size of 500. For crossover, the one-point method was used and the selection was elitist, taking into account the best 100 performing individuals for reproduction. For calculation of the individual fitness, the weight associated with the classification accuracy (w_a) had a value of 0.7, while the weight of the feature subset size (w_f) was set to 0.3. For performing the K nearest neighbors classification, the implementation provided by scikit-learn was used. The experiments were reproduced for a multitude of datasets, whose characteristics are outlined in table 1.

Table 1: Datasets used for experiments

Dataset	**Examples**	**Classes**	**Features**
ionosphere	351	2	34
sonar	208	2	60
Hill-Valley	606	2	617

At first, all generations were evaluated by training and testing the classifier on the same train test split. This resulted in a steady increase of population fitness,

but presented the risk of finding not the best feature subset overall (as desired), but the best feature subset for that particular split. Therefore, it has been attempted to train each generation on a different train test split, which resulted in inconsistent results (noisy population fitness and no clear separation of a leading solution). The same was the case with performing a different random cross validation for every generation and averaging the accuracy scores for each feature subset. In the final approach, all generations throughout the entire evolutionary process have been trained and tested on the same cross validation folds. The average accuracy scores for each feature subset was used to calculate the fitness value.

4.2 Experimental results

A first goal of the experiments was to assess if the genetic algorithm is actually capable of isolating a best performing feature subset. The secondary focus of the experiments was to assess the effectiveness of the feature selection in the context of classification tasks. Namely, it has been attempted to quantify the benefits implied by replacing the full feature set by the subset presenting highest fitness after genetic evolution.

Evolution of population fitness

The first goal of the experiments was quickly achieved by observing the average fitness of the population (figure 1) as well as the fitness of the best feature subset(figure 2) over the generations.

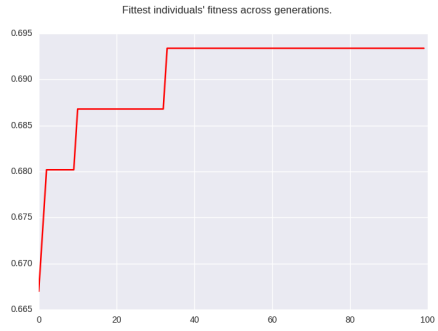

Figure 1: Evolution of best feature subset for the Ionosphere dataset

Figure 2: Evolution of the population fitness for the Ionosphere dataset

Optimizing the genetic algorithm

Even before reaching a quarter of the iterations, it has been observed that a lot of individuals in the population are duplicated. Not only that the best performing solutions are hardly replaced by new ones, but a large number of feature subsets appear in the population more than once. In order to confirm such signs of premature convergence, a primal diversity measure has been plotted i.e. the ratio of

unique individuals in a population. A potential issue in this situation could mean that the individuals in the final population might not represent the best feature subsets, since these configurations were never reached [11].
The way to prevent this was to modify the crossover mechanism by not allowing duplicate offspring to be added to the population. Figures 3 and 4 show the evolution process for the Ionosphere dataset, before and after adding the duplicate check in the crossover mechanism. Since a larger diversity might result in reaching more solutions, this duplicate check has been kept as part of the crossover throughout the rest of the experiments.

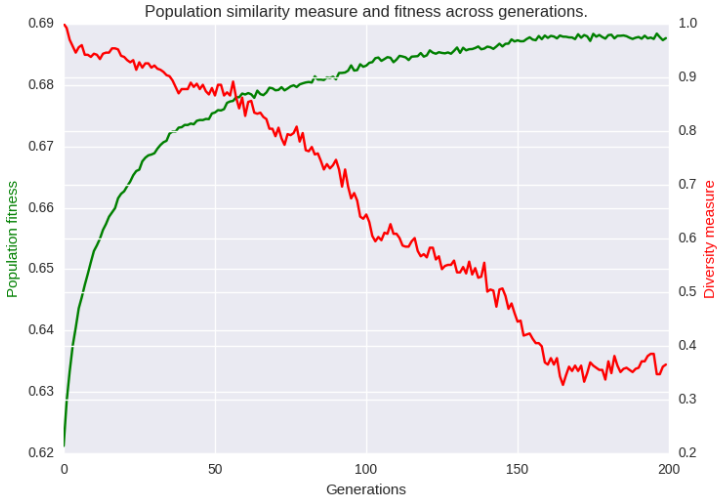

Figure 3: Allowing duplicate offspring

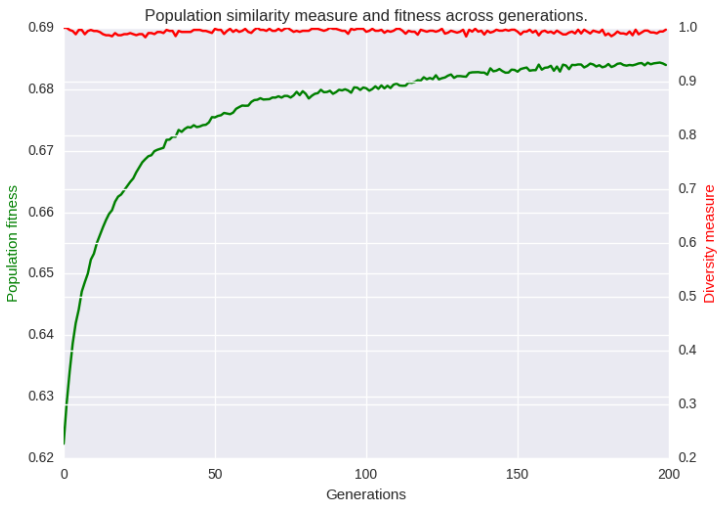

Figure 4: Not allowing duplicate offspring

Feature subset vs full feature set

In order to assess the effectiveness of feature selection, the performances of the best feature subset and the full feature set were compared. A kNN classifier was trained and tested by 10-fold cross validation and the slight increase in the results can be seen in figure 5.

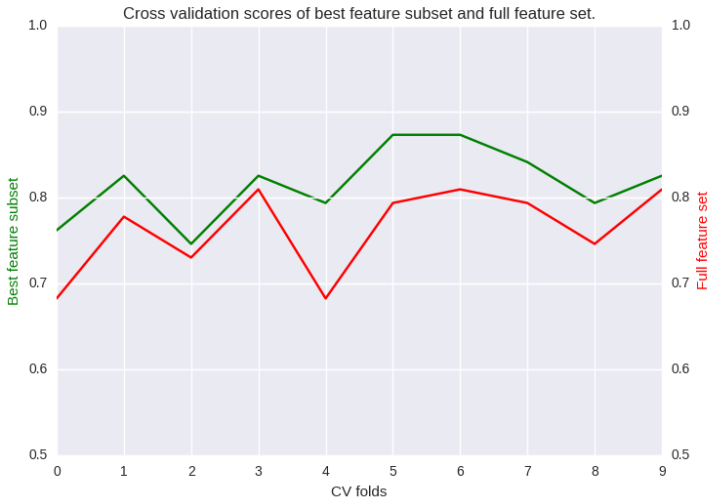

Figure 5: Results of 10-fold CV on the sonar data

The same results are confirmed by looking at the confusion matrix of running the kNN classifier (trained once with the full feature set and once with the best feature subset) on the same test split (figures 6 and 7).

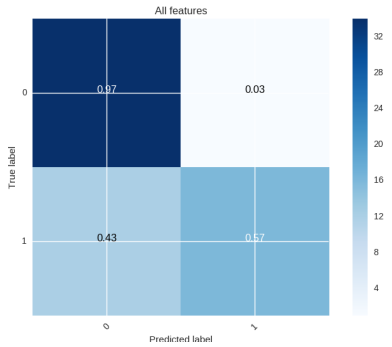

Figure 6: Sonar data

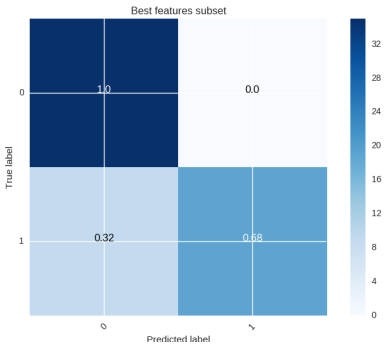

Figure 7: Sonar data

5 Conclusion and Future Work

The experimental results suggest that if such a minor improvement was obtained with a quite classical genetic algorithm implementation, more impact on the classification performance can be obtained by fine-tuning the methods of genetic algorithms. For example, as suggested in related work of [2], a more inquisitive population diversity measure could replace the one presented in section 4.2, taking into account even chromosome or even gene-level information. Further more, a benchmarking of the best feature subsets' performance with different classifiers would show the benefits of employing feature selection prior to learning. A more interesting perspective for further research efforts could consist of looking at feature selection in the context of high-dimensional data and overcoming the potential scalability issues.

References

[1] M Dash ' and H Liu. Feature Selection for Classification. *IDA ELSEVIER Intelligent Data Analysis*, 1(97):131–156, 1997.

[2] Pa Diaz-Gomez and Df Hougen. Initial Population for Genetic Algorithms: A Metric Approach. *Proceedings of the 2007 International Conference on Genetic and Evolutionary Methods*, pages 43–49, 2007.

[3] Isabelle Guyon, André Elisseeff, and Andre@tuebingen Mpg De. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[4] Mark Hall. Correlation-based Feature Selection for Machine Learning. *Methodology*, 21i195-i20(April):1–5, 1999.

[5] Cheng Lung Huang and Chieh J. Wang. A GA-based feature selection and parameters optimizationfor support vector machines. *Expert Systems with Applications*, 31(2):231–240, 2006.

[6] Mineichi Kudo and Jack Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(1):25–41, 2000.

[7] L Li, C R Weinberg, T a Darden, and L G Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics (Oxford, England)*, 17(12):1131–1142, 2001.

[8] H Liu and L Yu. Toward integrating feature selection algorithms for classification and clustering. *Ieee Transactions on Knowledge and Data Engineering*, 17(4):491–502, 2005.

[9] LA Nikravesh, M and Guyon, I and Gunn, S and Zadeh. *Feature Extraction: Foundations and Applications*. 2006.

[10] Michael L. Raymer, William F. Punch, Erik D. Goodman, Leslie A. Kuhn, and Anil K. Jain. Dimensionality Reduction Using Genetic Algorithms. *IEEE Transactions on Control Systems Technology*, 4(2):164–171, 2000.

[11] Simon Ronald. Duplicate Genotypes in a Genetic Algorithm. *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 793–798, 1998.

[12] W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(5):335–347, 1989.

[13] M. Srinivas and Lalit M. Patnaik. Genetic Algorithms: A Survey. *Computer*, 27(6):17–26, 1994.

[14] Cheng-huei Yang, Li-yeh Chuang, and Cheng-hong Yang. IG-GA : A Hybrid Filter / Wrapper Method for Feature Selection of Microarray Data. *Journal of Medical and Biological Engineering*, 30(1):23–28, 2009.