Evolving Linear Neural Networks for Features Space Dimensionality Reduction

Yury Tsoy
Computer Engineering Department
Tomsk Polytechnic University
Department of Economical Mathematics, Informatic and Statistics
Tomsk State University of Control Systems and Radioelectronics
Tomsk, Russia
Email: yurytsoy@gmail.com

***Abstract*—Principal Components Analysis (PCA) is one of the most wide-spread methods for dimensionality reduction, which is being applied in many research and problem domains. So far a lot of approaches to compute data matrix eigenvectors, which correspond to the Principal Components, were proposed, among which numerical methods and Hebbian-based learning for neural networks, including Generalized Hebbian Algorithm. In this paper a novel way for computing eigenvectors using evolving linear neural networks is introduced, which is not relying upon correlation between nodes, but uses special fitness function instead. Early removal of the low-informative linear subspaces is applied, which reduces computational complexity of the method, and besides eigenvectors coordinates are computed approximately to improve convergence and speed. The latter gave rise to the approach's name: pseudo-PCA. Experimental results show that not looking at inexact eigenvectors the approach allows effective reduction of the features space dimensionality with acceptable classification accuracy compared to some "classical" and modern approaches to solve classification problems.**

I. Introduction

Reduction of dimensionality of the features space is one of important problems in data analysis and pattern recognition domains. It can be useful for several reasons, for example:

1) Reduction of computation complexity, since the less the number of parameters to analyse, it's often the less the complexity of analysis.
2) Removal of noisy or irrelevant features, which might mislead the analysis and recognition processes.
3) In large data-bases reduction of objects descriptions size is useful to reduce demands to their storage and infrastructure.

One of the most widely used methods to solve features space reduction problem is Principal Components Analysis (PCA) [4]. It can be explained by the fact that this method is powerful enough and is based upon geometrically tractable and very clear idea. This idea concerns that an object's description is a linear combination of some unknown factors (components), represented in PCA as orthogonal vectors in a features space. Thus the solution to PCA problem is to find some suitable orthogonal basis, so that projection of objects' description (data points) on its 1st coordinate vector would have maximal possible variance. If a linear subspace correspondent to this vector is removed then projection of data points in the remaining subspace onto the 2nd vector would still be maximal, etc. This basis appears to be constructed by eigenvectors of the objects' descriptions covariance matrix. Features space dimensionality reduction is obtained via removal of those coordinate vectors in the basis found, which associated with smaller variances of the data points projections.

Due to necessity to find eigenvectors of the data covariance matrix $\mathbf{C}$ numeric methods are traditionally applied to find a solution for PCA method. Since $\mathbf{C}$ is symmetric special more effective methods can be applied [3]. However one of the problems to numeric computation of eigenvectors is that they are hard to parallelize due to their iterative nature and thus scale badly for large-scale problems.

There's an alternative to "purely" numeric methods like a well-known artificial neural networks (ANN) approach using Generalized Hebbian Algorithm (GHA) based upon Oja's rule [6], which uses Hebbian learning paradigm for linear neural networks to obtain eigenvectors as weights of output nodes. This method is easy to implement in parallel, but its convergence may require hundreds and thousands of iterations.

In this paper a novel method for computation of eigenvectors using neuroevolutionary approach is presented. Like GHA it uses ANNs without hidden layers and with linear activations, but the whole training is performed during an evolutionary process. One of peculiarities of the method is that eigenvector's coordinates are calculated approximately, which leads to better convergence (in many cases $O(10^1)$ iterations) and yet provides powerful tool for dimensionality reduction. The other feature is that online, directly during the ANN training, removal of low-contributing eigenvectors (and corresponding linear subspaces) is performed.

Since the presented method is not aimed at exact calculation of eigenvectors it is referred as **pseudo-PCA** (**pPCA**).

The paper is organized as follows. Baseline idea, which lies behind the method, is described in the Section II. Section III gives explanation for the introduced algorithm and specific procedures for fitness extimation and crossing. Experimental settings are given in the Section IV, while Section V contains results of experiments and their discussion. Conclusion provides brief review of the paper results and sketches-up possible

future research directions.

II. Idea of the Method

Note that traditional PCA formulation considers finding eigenvectors (principal components) with larger eigenvalues, or in other words, linear subspaces with small variance of data points projections do not fit to PCA. Such low-projection-variance subspaces will be referred as low-contributing since their contribution into data is small. The question of interest is if at some training step t some low-contributing subspace is found, can one throw it away without harm for the consequent steps? The answer is positive due to the following

Proposition Let $\mathbf{X} = \{\mathbf{X}_i, i = 1, \ldots, N\}, \mathbf{X}_i \in \mathcal{R}^n$ be a set of data points and $\mathbf{Q} = \{\mathbf{q}_i, i = 1, \ldots, n\}$ is an orthogonal basis in $\mathcal{R}^n$. Denote $proj_{\mathbf{q}_j}(\mathbf{X})$ as projection of data points from $\mathbf{X}$ onto coordinate vector $\mathbf{q}_j$ and $Var(proj_{\mathbf{q}_j}(\mathbf{X}))$ as a variance of correspondent projections. Then summation over all dimensions

$$\sum_{j=1,\ldots,n} Var(proj_{\mathbf{q}_j}(\mathbf{X})) \tag{1}$$

is constant and doesn't depend on $\mathbf{Q}$.

Proof: Variance of the j-th coordinate for the given data set in the standard basis is:

$$Var(proj_{\mathbf{e}_j}(\mathbf{X})) = \frac{1}{N}\sum_{k=1}^{N}(x_k^j)^2 - \frac{1}{N^2}\left(\sum_{k=1}^{N} x_k^j\right)^2.$$

where $\mathbf{e}_j$ – j-th coordinate vector in the standard basis, x_k^j – j-th coordinate of the k-th data point.

Summing over all coordinates we obtain:

$$\begin{aligned}\sum_{j=1}^{n} Var(proj_{\mathbf{e}_j}(\mathbf{X})) &= \sum_{j=1}^{n}\sum_{k=1}^{N}\frac{(x_k^j)^2}{N} - \sum_{j=1}^{n}\sum_{k,l=1}^{N}\frac{x_k^j x_l^j}{N^2} \\ &= \frac{1}{N}\sum_{k=1}^{N}\mathbf{X}_k^T\mathbf{X}_k - \frac{1}{N^2}\sum_{k,l=1}^{N}\mathbf{X}_k^T\mathbf{X}_l\end{aligned}$$

If $\mathbf{Y}_k$ is vector of coordinates of $\mathbf{X}_k$ in the basis $\mathbf{Q}$, then $\mathbf{X}_k = \mathbf{Q}\mathbf{Y}_k$ and:

$$\begin{aligned}&\sum_{j=1}^{n} Var(proj_{\mathbf{e}_j}(\mathbf{X})) = \\ &\frac{1}{N}\sum_{k=1}^{N}\mathbf{Y}_k^T\mathbf{Q}^T\mathbf{Q}\mathbf{Y}_k - \frac{1}{N^2}\sum_{k,l=1}^{N}\mathbf{Y}_k^T\mathbf{Q}^T\mathbf{Q}\mathbf{Y}_l \\ &= \sum_{j=1}^{n} Var(proj_{\mathbf{q}_j}(\mathbf{X}))\end{aligned}$$

which proofs the proposition. ∎

Since covariance matrix is symmetric, its eigenvectors are orthogonal and can be used as coordinate vectors of some orthogonal basis. Thus the proposition says, that since sum of projection variances is constant, then if columns $\hat{\mathbf{Q}}$ are estimates for eigenvectors, the low-contributing columns in $\hat{\mathbf{Q}}$ will be even less significant when coordinates of "primary" eigenvectors' estimates are defined more precisely (with lesser error). Thus we can throw away basis vectors, which do not fit some criterion, as non-informative without much harm for elaboration of coordinates of more significant vectors. This has certain benefits from the computational point of view because dimensionality of the problem is reduced when low-contributing basis vectors are removed.

In other words, one can treat the proposition in the way that sum variance of data points projections onto orthogonal coordinate vectors can be considered as a sort of finite resource, which is distributed among coordinate vectors. Hence if variance of projections onto certain vector is increased, there must be a vector (or vectors), for which variance of projections reduces. This result can serve as a basis for suggestion that low-contributing eigenvectors will not be "enhanced" (in the sense of projection variance) on the following iterations and thus can be neglected.

In this paper the criterion for removal of low-informative eigenvectors (subspaces) is (eigenvectors are supposed to be sorted by decrease of variance):

$$\frac{Var(proj_{\hat{\mathbf{q}}_1}(\mathbf{X}))}{Var(proj_{\hat{\mathbf{q}}_i}(\mathbf{X}))} < \tau, \tag{2}$$

where $\hat{\mathbf{q}}_i$ – is an estimate of the i-th eigenvector, τ – a threshold. The less the value of τ, the more eigenvectors will be neglected and thus more effective dimensionality reduction is possible. Typical choice for τ could be $5, 10, 15, \ldots$. However it's not clear for now how to adjust this parameter properly. There are some experimental evidences, described below, which show that acceptable value for τ for the presented algorithm could be 15, to trade-off between dimensionality reduction and classification accuracy, but this setting should definitely depend on the training data "geometry".

The question of selection of proper value for τ is relative to selection of the target number of features after the dimensionality reduction and doesn't have a universal answer for now, since there' s often no exact data on informativeness of various features. Nevertheless it seems that dynamical tuning of the number of features using τ is more reasonable than trying to guess a suitable features space dimensionality in advance.

Note, that criterion (2) can also be applied for the standard GHA as well to remove nodes, which weights correspond to low-contributing eigenvectors. This removal should not affect more significant nodes, because in the GHA training of the i-th output node doesn't depend on training of outputs $i+1, i+2, \ldots$ [6].

III. Algorithm's Description

Consider ANN without hidden layers and with 1 linear node. The output signal for such a network equals to $y = \mathbf{w}^T\mathbf{x}$. Assuming that $\|\mathbf{w}\| = 1$ it's clear that y is a coordinate of projection of the point $\mathbf{x}$ onto vector $\mathbf{w}$. For ANN with n output nodes, which weights equal to basis vectors for some orthogonal basis in $\mathcal{R}^n$, output signal for input $\mathbf{x}$ will

contain coordinates of projections onto those basis vectors. Thus, an equivalence can be established between weights of some output node and some coordinate vector from orthogonal basis. In what follows these two notions will be considered as synonims.

The above argumentation gives rise to the following neuroevolutionary algorithm:

1) Initialize random population, each individual is a candidate solution for pPCA.
2) Evaluate each individual using the following fitness function:

$$f = \alpha \sum_i Var(proj_{\mathbf{q}_i}(\mathbf{X})) \to \max, \qquad (3)$$

$$\alpha = (\hat{\mathbf{q}}_1^T \mathbf{r})^2, \mathbf{r} = \frac{\mathbf{C}\hat{\mathbf{q}}_1}{\|\mathbf{C}\hat{\mathbf{q}}_1\|}.$$

Coefficient α is introduced to evaluate how close the most "principal" eigenvector estimate to the true eigenvector. Let $\hat{\mathbf{q}}_1$ – be a normalized eigenvector, then $\alpha = \left(\frac{\hat{\mathbf{q}}_1^T \mathbf{C}\hat{\mathbf{q}}_1}{\|\mathbf{C}\hat{\mathbf{q}}_1\|}\right)^2 = \left(\frac{\lambda\hat{\mathbf{q}}_1^T\hat{\mathbf{q}}_1}{\|\lambda\hat{\mathbf{q}}_1\|}\right)^2 = 1$, and if $\hat{\mathbf{q}}_1$ is not an eigenvector: $\alpha < 1$. A small discussion of the selected fitness function is given in the subsection III-A below.

3) a Compute vector of mean variances $\overline{var}$ of data points projections onto eigenvectors' estimates by all individuals in population.
   b Starting from the last element in $\overline{var}$ check whether criterion (2) is satisfied. If it does for some k-th element, then remove correspondent genes from all individuals in population. If not, then proceed to **Step 4**.
4) **Selection**
5) **Crossing and Mutation**.
6) If algorithm's run is completed then proceed to **Step 7**, otherwise proceed to **Step 2**.
7) **Return** the best found individual.

Due to the Step 3 there is a possibility to remove output nodes in the ANN under training. Since the number of output nodes defines the number of principal components to describe our training data, this removal step implies reduction of the features space dimensionality. To implement evaluation of fitness and crossing of individuals special operators procedures are used, which are described below.

A. Fitness Estimation

Fitness evaluation procedure can be described by the following algorithm (for the i-th individual):

1) Assign genes of the individual to ANN weights.
2) Apply Gram-Schmidt orthogonalization to ANN weights.
3) Compute responses of ANN for each training sample.
4) Compute variances of ANN outputs.
5) Sort ANN nodes by decrease of variances.
6) Copy obtained vector of weights back into chromosome.

Sorting of ANN outputs by responses' variance is required to order basis vectors, correspondent to the nodes' weights, by their contribution to training data representation, and to define which coordinate vectors are more or less informative with respect to the training data. Note that evaluation of each individual is independent and can be performed in parallel.

Also note that expression (3) contains sum of projection variances, which was prooved to be constant in the previous Section. However this sum is constant only when estimates of all n principal components are considered. When some component is removed the sum over coordinate vectors for remaining subspaces will depend on those vectors' coordinates and since individuals in population represent different principal components estimates this sum will be different for reduced feature spaces.

B. Crossing

The presented algorithm uses a special crossover operator, which utilizes linear approximation of the fitness function gradient.

Let i-th and j-th individuals are to be crossed, each representing combination of all weights of ANN ($\mathbf{w}_i$ and $\mathbf{w}_j$ respectively). Suppose that the i-th individual is the better one (more fit than the j-th individual). Each vector of weights is split into N_O non overlapping parts, where N_O – is a number of ANN outputs:

$$\mathbf{w}_i = \{\mathbf{w}_i^{(k)}, k = 1, \ldots, N_O\},$$

so that each k-th part corresponds to weights of the k-th output node. With each part $\mathbf{w}_i^{(k)}$ a weight $v_i^{(k)}$ is associated, which equals to variance of projection of training data onto the correspondent coordinate vector.

The crossing of two individuals is performed part-wise to produce one offspring using:

$$\mathbf{c}^{(k)} = \mathbf{w}_i^{(k)} + \frac{|v_i^{(k)} - v_j^(k)|}{\left\|\mathbf{w}_i^{(k)} - \mathbf{w}_j^{(k)}\right\|}\left(\mathbf{w}_i^{(k)} - \mathbf{w}_j^{(k)}\right), \qquad (4)$$

where $\mathbf{c}^{(k)}$ – k-th part of the offspring's chromosome.

The fraction in (4) is necessary to approximate the absolute value of gradient of k-th vector so that overall expression (4) can be treated as linear approximation for the update of k-th part moving from the point $\mathbf{w}_i^{(k)}$.

IV. Experimental setup

Main goal of the experimental study is twofold:

1) Since pPCA doesn't obligatory yield exact principal components it's important to know how this affects classification accuracy.
2) It is also important to find out whether efficient dimensionality reduction is possible using the pPCA algorithm.

Testing of the proposed pPCA method will be conducted using several classification problems from the Proben1 set [7], namely: *cancer1*, *card1*, *diabetes1*, *glass1*, *heart1*, *horse1*, *soybean1*, *thyroid1*. Some numerical characteristics of these problems are given in the table I.

TABLE I
INFORMATION ON THE PROBLEMS FROM THE PROBEN1 TEST SET

Problem Name	# of features	# of classes	Training / Validation / Test sets sizes
cancer1	9	2	350 / 175 / 174
card1	51	2	345 / 173 / 172
diabetes1	8	2	384 / 192 / 192
glass1	9	6	107 / 54 / 53
heart1	35	2	460 / 230 / 230
horse1	58	3	182 / 91 / 91
thyroid1	21	3	3600 / 1800 / 1800

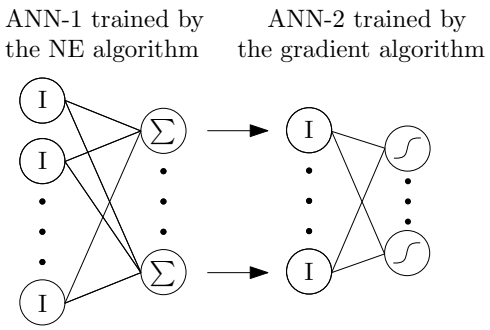

Fig. 1. Combination of ANN-1, trained using neuroevolutionary approach for dimensionality reduction, and ANN-2 trained for classification. Both networks doesn't contain hidden layers and output signals from ANN-1 serve as inputs for ANN-2. ANN-1 has linear activations, while output nodes for ANN-2 use "standard" sigmoid activation functions: $f(\mathbf{x}) = (1 + \exp(-\mathbf{w}^T\mathbf{x}))^{-1}$. "I" means identity activations, $f(x) = x, x \in \mathcal{R}$.

One of the peculiarities of the Proben1 dataset is that training, validation and testing subsets are defined explicitely by the Proben1's author. Also all preprocessing steps were also done by L. Prechelt including data normalization, numeric representation for categorical entries and filling gaps. All this was made to provide equal initial conditions for proper comparison of different classification methods [7].

NE approach is to be used for reduction of the features space dimensionality and after that transformed objects descriptions will be used to train feed-forward ANN using traditional gradient training. Scheme for combination of the resulting ANNs is shown in fig. 1. After ANN-1 and ANN-2 are trained they can be united into a single ANN in an obvious way.

The following values for criterion value τ for removal of non-informative linear subspaces according to (2) to be considered: 5, 10, 15, 20.

During each run a pPCA solution to be found and for this solution 10 ANNs will be trained, using the RPROP algorithm [8]. From these 10 ANNs only one is selected using classification error on a validation data set. This winning ANN is used for classification of samples from a test data set. For each problem 10 runs are performed in this way and mean classification accuracy is used for comparison and analysis.

Parameters settings for EA and RPROP are presented in tables II and III respectively. These parameters were picked experimentally and parameters for the RPROP algorithm were set according to the recomendations from [8].

TABLE II
PARAMETERS SETTINGS FOR EVOLUTIONARY ALGORITHM.

Parameter	Value
Encoding	Real
Population Size	50
Generations number	50
Selection	Tournament (tour size = 5)
Crossover probability	0.7
Mutation probability	0.05

TABLE III
PARAMETERS SETTINGS FOR RPROP ALGORITHM.

Parameter	Value
Epochs	1000
η^+	1.2
η^-	0.5
Δ_0	0.1
$\Delta_{\max}$	50
$\Delta_{\min}$	$1e^{-6}$
Early stopping threshold	10%

TABLE IV
CLASSIFICATION ERRORS (%) FOR DIFFERENT VALUES OF τ. VALUES IN BRACKETS SHOW INITIAL (FIRST COLUMN) AND RESULTING (ALL OTHER COLUMNS) DIMENSIONALITY OF THE FEATURES SPACE.

Problem	$\tau = 5$	$\tau = 10$	$\tau = 15$	$\tau = 20$
cancer1 (9)	2.30 (1)	2.82 (1.2)	1.78 (4.6)	1.84 (6.3)
card1 (51)	16.28 (28.5)	15.41 (50.7)	15.64 (51)	15.76 (51)
diabetes1 (8)	24.95 (7.6)	25.00 (8)	25.00 (8)	25.00 (8)
glass1 (9)	36.23 (5.5)	33.02 (6.7)	32.07 (7.9)	32.26 (8.4)
heart1 (35)	21.13 (22.3)	19.91 (31.5)	20.00 (34.2)	20.04 (35)
horse1 (58)	28.79 (35.3)	29.23 (57.7)	30.66 (58)	29.56 (58)
soybean1 (82)	50.65 (3.3)	20.47 (11.7)	11.94 (26.1)	10.47 (37.3)
thyroid1 (21)	7.19 (8.9)	6.03 (16.3)	5.87 (18)	5.92 (19.8)

Software implementation of pPCA and its application for solving classification problems uses the MentalAlchemy[1] and the Encog[2] open-source libraries and is available as part of the MentalAlchemy project.

V. RESULTS OF EXPERIMENTS AND DISCUSSION

A. Classification accuracy results

Test set classification errors are given in the Table IV. Also the number of ANN outputs (remaining principal components estimates) in result of neuroevolutionary pPCA is given in brackets to evaluate dimensionality reduction.

For different problems change of τ had different impact on the resulting number of features due to pPCA. Nevertheless

[1]http://code.google.com/p/mentalalchemy/
[2]http://www.heatonresearch.com/encog

TABLE V
CLASSIFICATION ERRORS (%) OBTAINED USING SOME OTHER APPROACHES. FOR [7], [1], [10] THE BEST OBTAINED RESULTS ARE GIVEN. GA – TRAINING OF ANN USING GENETIC ALGORITHM. PRUNNING – TRAINING OF ANN USING POSTPROCESSING WITH PRUNNING ALGORITHMS.

Problem	ANN[7]	GA[1]	Prunning[10]	pPCA, $\tau = 15$
cancer1 (9)	1.38	1.24	1.1	1.78 (4.6)
card1 (51)	14.05	14.27	13.7	15.64 (51)
diabetes1 (8)	24.10	23.70	20.8	25.00 (8)
glass1 (9)	32.7	47.62	30.2	32.07 (7.9)
heart1 (35)	19.72	21.87	18.5	20.00 (34.2)
horse1 (58)	29.19	26.44	26.9	30.66 (58)
soybean1 (82)	9.06	8.47	N/A	11.94 (26.1)
thyroid1 (21)	2.32	6.12	5.7	5.87 (18)

it can be seen that recommendable value for τ is likely to be not less than 15.

For comparison Table V contains the best classification error values for the problems under consideration for some other methods. Results from [7] are obtained using traditional approach for ANN training with manual tuning of ANN structure and using RPROP algorithm for weights tuning; results of a pure evolutionary training of neural weights for ANN with a fixed structure are taken from the [1]; and research results from [10] are obtained for ANN trained using gradient-based method with application of prunning algorithms.

Comparison of these results shows that for many problems features space dimensionality reduction obtained using pPCA provides comparable classification accuracy with respect to more "traditional" training methods. Note that for some problems there was a significant reduction of the number of features (problems *cancer1* and *soybean1*).

B. Features space dimensionality reduction

Figures 2 and 3 depict examples of change of a mean features space dimensionality (the number of remaining components for pPCA); and variance of projections onto eigenvectors estimates with respect to algorithm iterations (generations). All results are averaged over 100 independent runs. It can be seen that variance of projections onto the "main" principal component vector increases over time, while variances for other vectors decrease, which shows the improvement of the principal components coordinates by the pPCA.

In order to estimate efficiency of the dimensionality reduction using the pPCA ($\tau = 15$) a comparison with results from the literature was made (table VI). The methods are: Generalized Hebbian Algorithm with dynamical removal of output nodes using (2) for $\tau = 15$ (GHA*); Layered Genetic Programming (FLGP) [5]; Recursive Feature Elimination combined with multi-layered neural network (RFENN) and support vector machines (RFESVM) [9].

The results from the table VI show that pPCA shows slightly worse performance considering the resulting dimensionality of the features space, but it is compensated by lower classification error rates. Also note results for the GHA*, which

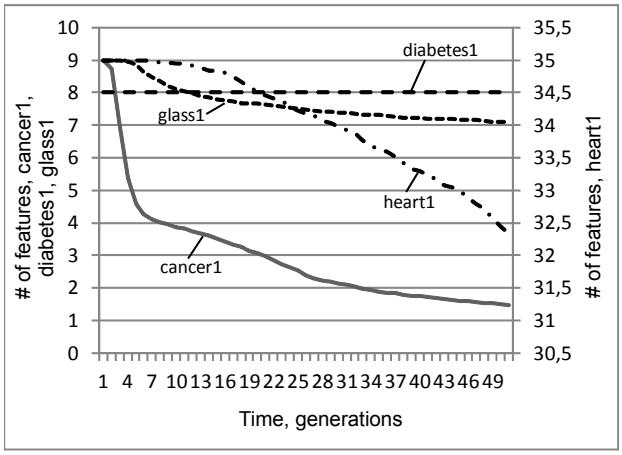

Fig. 2. Change of a mean number of ANN outputs (features space dimensionality) trained using pPCA over time for the problems *cancer1*, *diabetes1*, *glass1*, *heart1*, for $\tau = 10$. The results are averaged over 100 runs.

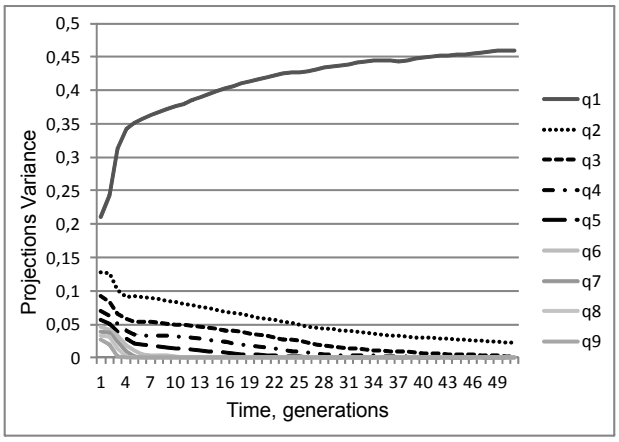

Fig. 3. Dynamics of a mean variance of projections onto eigenvectors estimates for the problem *cancer1*, for $\tau = 10$. The results are averaged over 100 runs.

TABLE VI
COMPARISON OF THE TEST SET CLASSIFICATION ERRORS (%) OBTAINED USING DIFFERENT FEATURES SELECTION METHODS FOR *cancer1*, *diabetes1* AND *heart1* PROBLEMS. AVERAGE DIMENSIONALITY OF THE RESULTING FEATURES SPACE IS GIVEN IN BRACKETS.

Method	cancer1	diabetes1	heart1
GHA*	2.13 (1)	25.93 (1.8)	22.17 (3.1)
FLGP [5]	2.24 (5.2)	27.24 (6.1)	22.40 (11.0)
RFENN [9]	4.00 (7)	24.90 (2)	21.00 (27)
RFESVM [9]	3.70 (7)	24.50 (3)	20.00 (18)
pPCA	1.78 (4.6)	25.00 (8)	20.00 (34.2)

outperformed all other methods for dimensionality reduction while providing a comparable classification accuracy.

With all these results the following quite interesting conclusion can be made:

There are cases when it's not necessary to know exact principal components of data covariance matrix to perform a reliable dimensionality reduction.

Interesting question is what the price for use of approximate principal components is. If exact non-informative linear subspaces were ripped off, then transformed features would be living in a "correct" subspace and hence there is a linear transformation, which converts remaining eigenvectors estimates obtained by pPCA to the exact solutions for PCA. But if dimensionality reduction was made for inexact eigenvectors then there's a transformation error for pPCA comparing with the PCA. The measurement of this error is tricky because exact eigenvectors are supposed to be unknown. However the method seems to have a "self-repairing" feature: even if inexact linear subspaces were removed to cause a representation error, the remaining components' coordinates will be defined more precisely on consequent steps of the algorithm, which means that the remaining linear subspace is rotated and this rotation should diminish the error. Hence pPCA error is an open question, which should be studied thoroughly in order to understand features of the pPCA and hopefully develop new efficient methods for dimensionality reduction.

VI. Conclusion

The paper presents novel way for dimensionality reduction using pseudo-PCA. The method is based upon application of neuroevolutionary approach for feed-forward linear neural networks without hidden nodes and uses special procedures for fitness estimation and a crossover operator.

Experimental results show that even though the reduction of features space dimensionality was performed using inexact eigenvectors the obtained classification results are comparable with that of ANNs trained in more traditional ways.

Interesting questions for the future research are parallelization of the pPCA algorithm and analysis of the computational error of eigenvectors estimates. Also a question for selection of the proper value for τ parameter, which regulates the resulting features space dimensionality, is open and its study could improve the adaptive properties of the pseudo-PCA method.

Acknowledgment

The research is supported by the Russian Foundation for Basic Researches (project no. 11-07-00027-a).

Author thanks Dr. Yu. Burkatovskaya for notes on the paper contents, O. Abdulganeev for classification results using GHA with dynamical reduction of dimensionality, and anonymous reviewers for their valuable comments, which helped to improve the paper.

References

[1] F.Y. Barrera *Busqueda de la estructura optima de redes neurales con Algoritmos Geneticos y Simulated Annealing. Verificacion con el benchmark PROBEN1* Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. 2007. Vol. 11. No. 34. P. 41-61.

[2] I. De Falco, A. Della Cioppa, E. Tarantino *Discovering interesting classification rules with genetic programming* Applied Soft Computing. 2002. Vol. 1. No. 4. P. 257-269.

[3] G.H. Golub, C.F. Van Loan *Matrix Computations*. Second Edition. Baltimor, London: The John Hopkins University Press, 1989.

[4] I.T. Jolliffe *Principal Component Analysis*. Springer Series in Statistics. NY: Springer, 2002.

[5] J.-Y. Lin, H.-R. Ke, B.-C. Chien, W.-P. Yang *Classifier design with feature selection and feature extraction using layered genetic programming* Expert Systems with Applications. 2008. Vol. 34. No. 2. P. 1384-1393.

[6] E. Oja *Simplified neuron model as a principal component analyzer* Journal of Mathematical Biology. 1982. Vol. 15. No. 3. P. 267-273.

[7] L. Prechelt *PROBEN1 – a set of neural network benchmark problems and benchmarking rules*. Technical Report 21/94. – Universitat Karlsruhe, Karlsruhe, Germany, 1994.

[8] M. Riedmiller *Rprop - Description and Implementation Details* Technical report. Universtiy of Karlsruhe, 1994.

[9] T. Windeatt, R. Duangsoithong, R. Smith *Embedded Feature Ranking for Ensemble MLP Classifiers* IEEE Transactions on Neural Networks. 2011. Vol. 22. No. 6. P. 988-994.

[10] N. Weman *Empirical Investigation of the Effect of Pruning Artifical Neural Networks With Respect to Increased Generalization Ability* Linkopings universitet, Sweden, 2010.

[11] F. Zhu, S.-U. Guan, P. Li *Feature Selection for Modular GA-based Classification* Applied Soft Computing. 2004. Vol. 4. No. 4. P. 381-393.