Evolving Linear Neural Networks for Features Space Dimensionality Reduction

Yury Tsoy

Tomsk Polytechnic University
Tomsk State University of Control Systems and Radioelectronics
Tomsk, Russia

June 14, 2012

Table of contents

Principal Components Analysis

Principal Components Analysis (PCA) is one of the most popular methods for dimensionality reduction for pattern recognition problems. Concerns computing of eigenvectors for the data covariance matrix. Fast and efficient ($O(n^{2.36})$ with all the tricks).

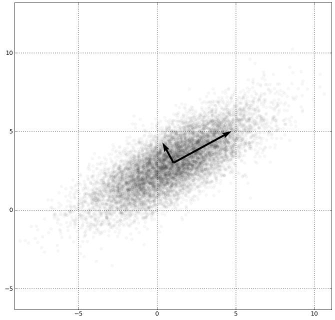


Figure: PCA illustrative example

Generalized Hebbian Algorithm

1. Initialization of the linear ANN without hidden nodes. The number of outputs = required dimensionality.
2. Update ANN weights. For each training sample:

$$\begin{aligned} y_j(t) &= \sum_{i=1}^{m} w_{ji}(t) x_i(t), \\ \Delta w_{ji}(t) &= \eta \left[y_j(t) x_i(t) - y_j(t) \sum_{k=1}^{j} w_{ki}(t) y_k(t) \right], \end{aligned}$$

3. If stopping criterion is failed go to **Step 2**.

Generalized Hebbian Algorithm

Two options

1. Compute all eigenvectors and eigenvalues and apply selection mechanism to reduce dimensionality. **Higher computational complexity**.
2. Set the required dimensionality beforehand. **Requires guessing of ”true” data set dimensionality.**

Sweet dreams

Itd be good if we could remove output nodes dynamically.

1. Reduces computational complexity.
2. Doesnt require guessing the data dimensionality.

A bit of theory

Can we remove inexact non-informative eigenvectors?

Proposition

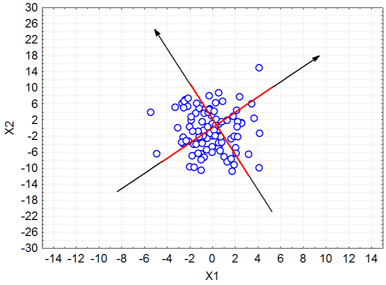
Let $\mathbf{X} = \{\mathbf{X}_i, i = 1, \ldots, N\}, \mathbf{X}_i \in \mathcal{R}^n$ be a set of data points and $\mathbf{Q} = \{\mathbf{q}_i, i = 1, \ldots, n\}$ is an orthogonal basis in $\mathcal{R}^n$. Denote $proj_{\mathbf{q}_i}(\mathbf{X})$ as projection of data points from $\mathbf{X}$ onto coordinate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

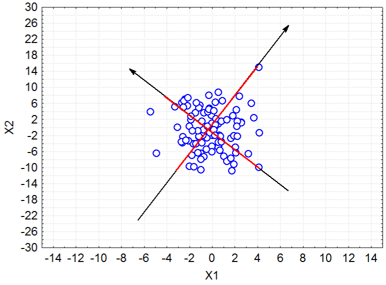
$$\sum_{i=1,\ldots,n} Var(proj_{\mathbf{q}_i}(\mathbf{X}))$$

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

In other words . . .

Illustrative example





Sum variance of projections can be treated as a finite resource.

How to decide?

We suppose that all eigenvectors estimates are sorted by projection variances (e.g. significance).

Criterion for throwing away "bad" eigenvectors estimates:

$$\frac{Var(proj_{\hat{\mathbf{q}}_0}(\mathbf{X}))}{Var(proj_{\hat{\mathbf{q}}_i}(\mathbf{X}))} > \tau \qquad (1)$$

where $\hat{\mathbf{q}}_i$ – estimate of the *i*-th eigenvector, τ is a threshold. Typical values for τ are 5, 10, 15, 20, ...

It is possible to truncate low-informative subspaces without knowing exact coordinates of principal eigenvectors $\Rightarrow$ **pseudo-PCA (pPCA)**.

Sorry for the typo :(

The Neuroevolutionary Algorithm

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

$$\begin{aligned} f &= \alpha * \sum_{i=1,\dots,n} Var(proj_{\hat{\mathbf{q}}_i}(\mathbf{X})) \rightarrow max, \\ \alpha &= (\hat{\mathbf{q}}_0^T \mathbf{r})^2, \mathbf{r} = \mathbf{C}\hat{\mathbf{q}}_0 / \|\mathbf{C}\hat{\mathbf{q}}_0\|. \end{aligned}$$

   and remove nodes, for which criterion (1) is satisfied.
3. **Selection**
4. **Crossing** and **Mutation**.
5. If algorithm's run is completed then proceed to **Step 6**, otherwise proceed to **Step 2**.
6. **Return** the best found individual.

Fitness evaluation

1. Assign genes of individual to Artificial neural network with linear nodes.
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 the decrease of variances.
6. Copy obtained vector of weights back into chromosome.

Special crossing operator

Two parents $\rightarrow$ one child.
Crossing is performed "by neurons" using formula (for the k-th output node):

$$\mathbf{c}^{(k)} = \mathbf{w}_i^{(k)} + \frac{|\widetilde{v}_i^{(k)} - \widetilde{v}_j^{(k)}|}{\|\mathbf{w}_i^{(k)} - \mathbf{w}_j^{(k)}\|}(\mathbf{w}_i^{(k)} - \mathbf{w}_j^{(k)}) \qquad (2)$$

where $\widetilde{v}_i^{(k)} = v_i^{(k)}/(v_i^{(k)} + v_j^{(k)})$ – normalized "weight" of the k-th node; v_i^k – variance of projection onto the (k)-node of i-th individual; $\|\cdot\|$ – Euclidian norm.

Overall expression (2) can be treated as linear approximation of the gradient ascent for the update of k-th part, moving from the point $\mathbf{w}_i^{(k)}$.

Goals & Test Problems

Goals

1. It is important to find out whether efficient dimensionality reduction is possible.
2. Since pPCA doesn't yield linear subspaces associated with the principal components it's also important to know how this affects classification accuracy.

Proben1 data set

Proben1 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

Comparison

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)

Table: Classification errors (%) for different values of τ

Comparison

Problem	Proben1	GA	Prunning	$\tau = 15$
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)

Table: Classification errors (%), for some other approaches on the Proben1 data set

Change of averaged mean dimensionality

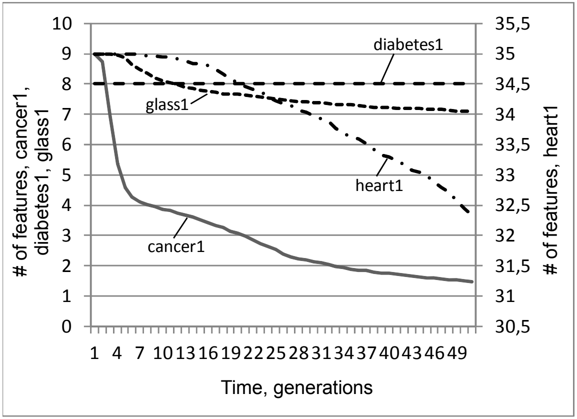


Figure: Change of averaged dimensionality for *cancer1*, *diabetes1*, *glass1* and *heart1* problems. $\tau = 10$.

Change of averaged variance of projections

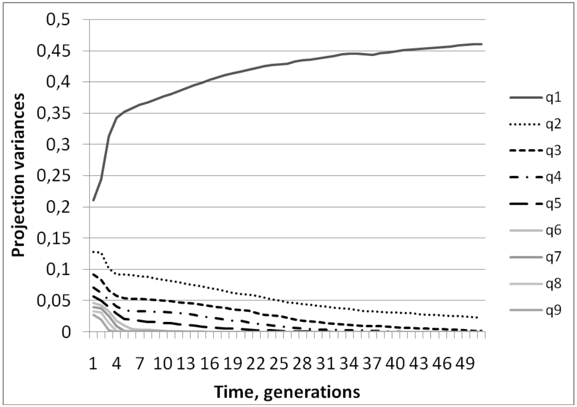


Figure: Change of averaged variances of projections of data points onto the first 3 eigenvectors estimates for *cancer1* problem. $\tau = 10$.

Comparison

- Dynamical Generalized Hebbian Algorithm with dynamical removal of output nodes using (1) for $\tau = 15$ (GHA*);
- Layered Genetic Programming (FLGP) (Lin etal., 2008);
- Recursive Feature Elimination combined with multi-layered neural network (RFENN)
- ...and support vector machines (RFESVM) (Windeatt, 2011).

Method	cancer1	diabetes1	heart1
DGHA	1.44 (4)	24.43 (8)	22.52 (17.7)
DGHA (autocor)	2.13 (1)	25.93 (1.8)	22.17 (3.1)
FLGP	2.24 (5.2)	27.24 (6.1)	22.40 (11.0)
RFENN	4.00 (7)	24.90 (2)	21.00 (27)
RFESVM	3.70 (7)	24.50 (3)	20.00 (18)
NE pPCA	1.78 (4.6)	25.00 (8)	20.00 (34.2)

Table: 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.

Experimental corollary

Pre-Conclusion

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

Self-repairing feature

Even if inexact linear subspaces were removed to cause a representation error, the remaining components' coordinates will be refined in consecutive steps, which means that the remaining linear subspace is rotated and this rotation should diminish the error.

Precedent

Neuroevolutionary algorithm which produces neural network with tractable functionality.

Dynamical GHA

1. Initialization of the linear ANN without hidden nodes. The number of outputs = required dimensionality.
2. Compute projections variances and remove output nodes, which satisfy to the criterion (1).
3. Update ANN weights. For each training sample:

$$\begin{aligned} y_j(t) &= \sum_{i=1}^{m} w_{ji}(t) x_i(t), \\ \Delta w_{ji}(t) &= \eta \left[y_j(t) x_i(t) - y_j(t) \sum_{k=1}^{j} w_{ki}(t) y_k(t) \right], \end{aligned}$$

4. If stopping criterion is failed go to **Step 2**.

Approximate eigenvectors

Ok, we can work with approximate covariance matrix eigenvectors.
Sources of inexactness:

- Approximate methods to compute eigenvectors.
- Inexact covariance matrix.

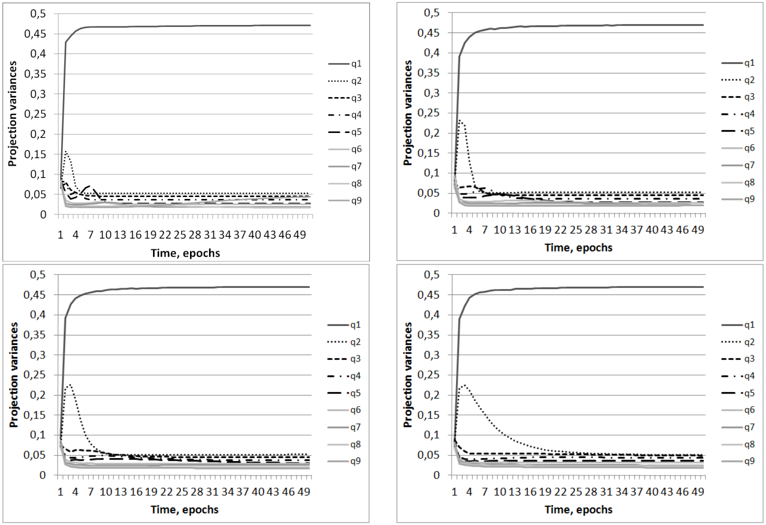
Dynamical GHA with reduced data set

1. Initialization of the linear ANN without hidden nodes. The number of outputs = required dimensionality.
2. Compute projections variances and remove output nodes, which satisfy to the criterion (1).
3. Sample $r\%$ of the data from the training set to update ANN weights.
4. Update ANN weights. For each training sample:

$$\begin{aligned} y_j(t) &= \sum_{i=1}^{m} w_{ji}(t) x_i(t), \\ \Delta w_{ji}(t) &= \eta \left[y_j(t) x_i(t) - y_j(t) \sum_{k=1}^{j} w_{ki}(t) y_k(t) \right], \end{aligned}$$

5. If stopping criterion is failed go to **Step 2**.

Change of projection variances (cancer1)



a) 100% of data; b) 50% of data;
c) 25% of data; d) 10% of data.

Speed-up and Accuracy (cancer1, 9 features, 350 samples)

Method	$\tau = 5$	$\tau = 10$	$\tau = 15$	$\tau = 20$
cancer1, 100%	5.67	3.62	2.17	1.38
cancer1, 50%	9.13	5.93	3.35	2.37
cancer1, 25%	12.80	9.07	5.29	3.98
cancer1, 10%	20.32	13.35	8.81	7.31

Table: Speed-up of the DGHA using partial data in relation to the GHA (218.74 ms).

Method	$\tau = 5$	$\tau = 10$	$\tau = 15$	$\tau = 20$
cancer1, 100%	2.30 (1.0)	1.78 (2.5)	1.44 (4.0)	1.67 (5.6)
cancer1, 50%	2.30 (1.0)	2.24 (1.2)	1.49 (4.3)	1.49 (6.3)
cancer1, 25%	2.30 (1.0)	2.24 (1.2)	2.30 (4.4)	1.90 (6.2)
cancer1, 10%	2.30 (1.0)	2.18 (1.8)	1.78 (4.7)	1.90 (7)

Table: Classification error of the DGHA (average dimensionality).

Speed-up and Accuracy (horse1, 58 features, 182 samples)

Method	$\tau = 5$	$\tau = 10$	$\tau = 15$	$\tau = 20$
horse1, 100%	11.81	4.38	2.93	2.36
horse1, 50%	21.81	8.62	5.73	4.41
horse1, 25%	43.35	16.23	10.68	8.35
horse1, 10%	90.32	34.04	22.43	17.25

Table: Speed-up of the DGHA using partial data in relation to the GHA (12993.62 ms).

Method	$\tau = 5$	$\tau = 10$	$\tau = 15$	$\tau = 20$
horse1, 100%	34.07 (1.0)	32.86 (5.3)	29.89 (27.3)	26.81 (32.4)
horse1, 50%	34.18 (1.0)	32.86 (15.9)	29.23 (26.9)	28.68 (32.6)
horse1, 25%	34.06 (1.0)	30.77 (20.1)	29.01 (28.6)	28.57 (33.7)
horse1, 10%	33.85 (1.0)	29.34 (22.8)	28.02 (30.6)	28.68 (36.3)

Table: Classification error of the DGHA (average dimensionality).

Change of averaged mean dimensionality (DGHA)

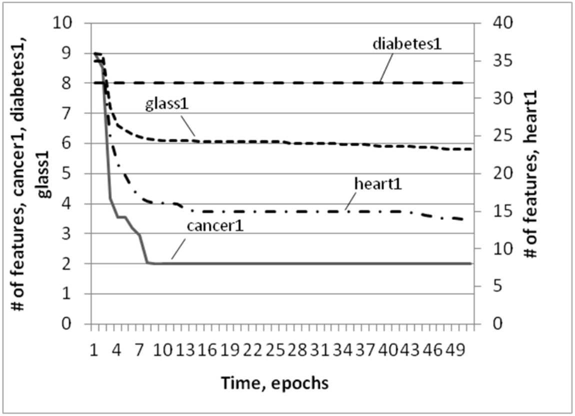


Figure: Change of averaged dimensionality for *cancer1*, *diabetes1*, *glass1* and *heart1* problems for DGHA. $\tau = 10$.

Conclusion

Quite a simple proposition lead to:

- Novel way for dimensionality reduction using pseudo-PCA.
- NE pPCA – way to evolutionary training of ANN with tractable and understandable results.
- Dynamical modification of the GHA algorithm (DGHA).
- Use of part of data to speed-up the DGHA.

Future Research:

1. Parallelization of the NE pPCA. The most time consuming part is computation of fitness ($\tilde{7}$5-80% of time). Each individual can be evaluated in parallel.
2. Constraints for pPCA: use criteria from PCA and/or try to keep certain amount of information when performing nodes removal.

Acknowledgements

The Foundation (*The Life*)

The research is supported by the Russian Foundation for Basic Researches (projects no. 11-07-00027-a, 12-08-00296-a, and 12-07-09226-mob_z).

The Colleagues (*The Universe*)

Author thanks Dr. Yu. Burkatovskaya for her notes on the paper contents, and O. Abdulganeev for classification results for GHA with dynamical reduction of dimensionality for data auto-correlation matrix.

The Source Code (*Everything. . .*)

Mental Alchemy (`http://code.google.com/p/mentalalchemy`) and Encog (`http://www.heatonresearch.com/encog`) open-source projects were used to implement all the algorithms and experiments.

Thank you for attention!

Yury Tsoy
yurytsoy@gmail.com