UDC 004.032.26

TRAINING PROBABILISTIC NEURAL NETWORKS ON THE SINGLE CLASS PATTERN MATRIX AND ON CONCATENATION OF PATTERN MATRICES

Romanuke V.V., Yegoshyna G.A., Voronoy S.M.

O. S. Popov Odesa National Academy of Telecommunications,
1 Kuznechna St., Odesa, 65029, Ukraine.
romanukevadimv@gmail.com, yegoshyna@onat.edu.ua, vsm3aic@gmail.com

НАВЧАННЯ ІМОВІРНІСНИХ НЕЙРОННИХ МЕРЕЖ НА ПОКЛАСОВІЙ ЕТАЛОННІЙ МАТРИЦІ ТА НА КОНКАТЕНАЦІЇ ЕТАЛОННИХ МАТРИЦЬ

Романюк В.В., Єгошина Г.А., Вороной С.М.

Одеська національна академія зв'язку ім. О. С. Попова,
65029, Україна, м. Одеса, вул. Кузнечна, 1.
romanukevadimv@gmail.com, yegoshyna@onat.edu.ua, vsm3aic@gmail.com

ОБУЧЕНИЕ ВЕРОЯТНОСТНЫХ НЕЙРОННЫХ СЕТЕЙ НА ПОКЛАССОВОЙ ЭТАЛОННОЙ МАТРИЦЕ И НА КОНКАТЕНАЦИИ ЭТАЛОННЫХ МАТРИЦ

Романюк В.В., Егошина А.А., Вороной С.М.

Одесская национальная академия связи им. А. С. Попова,
65029, Украина, г. Одесса, ул. Кузнечная, 1.
romanukevadimv@gmail.com, yegoshyna@onat.edu.ua, vsm3aic@gmail.com

Abstract. A possibility to optimize the probabilistic neural network is studied by building an efficient training set. Commonly, the training dataset for a probabilistic neural network is a matrix whose columns represent classes. If every class has only one column, the matrix is called the single class pattern matrix. However, the simple architecture of the probabilistic neural network does not imply that the class pattern must be single. First, the range of values for a class feature can be too wide. Then it is desirable to break it into subranges, each of which will give its own average and thus a few patterns for this class will be formed. Second, a class feature can have a finite number of its values, where every value has the same importance. Then it would be incorrect to calculate an average and use it in the respective single class pattern. Thus, it is studied whether concatenation of pattern matrices into a long pattern matrix is reasonable. In fact, the goal of the research is to ascertain whether it is efficient to build probabilistic neural networks on long pattern matrices. The criterion of the efficiency is performance of the probabilistic neural network, i. e. either its accuracy or percentage of errors. To achieve the goal, performance of the probabilistic neural network is estimated on the case when the class is described by a few class patterns. The probabilistic neural networks are then tested for the two subcases: when objects generated by different class patterns are fed to the input, and when objects are generated by a generalized single class pattern. Eventually, it is ascertained that training probabilistic neural networks on the single class pattern matrix (obtained either by averaging over the available pattern matrices or just by using one pattern per class) is more efficient when the objects to be classified do not inherit any class pattern numerical properties. On the contrary, when the objects to be classified may have some distinct numerical properties of a few class patterns, then training probabilistic neural networks on long pattern matrices is more efficient ensuring noticeably higher accuracy. The smooth training method appears inefficient in improving the performance.

Key words: classification problem, probabilistic neural network, pattern matrix, single class pattern matrix, horizontal concatenation of matrices, smooth training, noised pattern matrices, long pattern matrix, generalized single class pattern.

Анотація. Вивчається можливість оптимізувати імовірнісну нейронну мережу на основі ефективної навчальної множини. Зазвичай навчальний набір даних для імовірнісної нейронної мережі представляє собою матрицю, чиї стовпці репрезентують класи. Якщо кожен клас має лише один стовпець, ця матриця називається покласовою еталонною матрицею. Однак проста архітектура імовірнісної нейронної мережі не означає, що кожен клас має бути представлений одним еталоном. По-перше, діапазон значень ознаки класу може бути занадто широким. Тоді бажано розбити його на піддіапазони, кожен з яких дасть своє середнє, і таким чином буде сформовано декілька еталонів для даного класу. По-друге, ознака класу може мати скінченну кількість значень, де кожне значення має однакову значимість. Тоді було б некоректно обчислювати середнє і використовувати його у відповідному покласовому еталоні. Тому вивчається, чи доцільною буде конкатенація еталонних матриць у довгу еталонну матрицю. Фактично метою дослідження є встановлення того, чи є ефективною побудова імовірнісних нейронних мереж на довгих еталонних матрицях. Критерієм ефективності є продуктивність імовірнісної нейронної мережі, тобто її точність або відсоток помилок. Для досягнення цієї мети продуктивність імовірнісної нейронної мережі оцінюється на випадку, коли клас описується декількома еталонами. Далі імовірнісні нейронні мережі тестуються для двох підвипадків: коли об'єкти, що подаються на вхід, генеруються за різних класових еталонів, і коли об'єкти генеруються за узагальненого покласового еталону. Зрештою, встановлюється, що навчання імовірнісних нейронних мереж на покласовій еталонній матриці (отриманій або на основі усереднення за наявними еталонними матрицями або просто за використання одного еталону для кожного класу) є більш ефективним, коли об'єкти для класифікації не наслідують жодних числових властивостей класових еталонів. І навпаки, коли об'єкти для класифікації можуть мати деякі виразні числові властивості кількох класових еталонів, тоді навчання імовірнісних нейронних мереж на довгих еталонних матрицях є більш ефективним, забезпечуючи значно вищу точність. Метод гладкого навчання в удосконаленні продуктивності виявляється неефективним.

Ключові слова: задача класифікації, імовірнісна нейронна мережа, еталонна матриця, покласова еталонна матриця, горизонтальна конкатенація матриць, гладке навчання, зашумлені еталонні матриці, довга еталонна матриця, узагальнений покласовий еталон.

Аннотация. Изучается возможность оптимизации вероятностной нейронной сети на основе эффективного обучающего множества. Обычно обучающее множество данных для вероятностной нейронной сети представляет собой матрицу, чьи столбцы соответствуют классам. Если каждый класс имеет лишь один столбец, эта матрица называется поклассовой эталонной матрицей. Однако простая архитектура вероятностной нейронной сети не означает, что каждый класс должен быть представлен одним эталоном. Во-первых, диапазон значений признака класса может быть слишком широким. Тогда желательно разбить его на поддиапазоны, каждый из которых даст своё среднее, и таким образом будет сформировано несколько эталонов для данного класса. Во-вторых, признак класса может иметь конечное количество значений, где каждое значение имеет одинаковую значимость. Тогда было бы некорректно вычислять среднее и использовать его в соответствующем поклассовом эталоне. Поэтому изучается, целесообразной ли будет конкатенация эталонных матриц в длинную эталонную матрицу. Фактически цель исследования состоит в установлении того, является ли эффективным построение вероятностных нейронных сетей на длинных эталонных матрицах. Критерием эффективности является производительность вероятностной нейронной сети, т. е. её точность или процент ошибок. Для достижения этой цели производительность вероятностной нейронной сети оценивается на случае, когда класс описывается несколькими эталонами. Дальше вероятностные нейронные сети тестируются для двух подслучаев: когда объекты, которые подаются на вход, генерируются при разных классовых эталонах, и когда объекты генерируются при обобщённом поклассовом эталоне. В конце концов, устанавливается, что обучение вероятностных нейронных сетей на поклассовой эталонной матрице (полученной или на основе усреднения по имеющимся эталонным матрицам или просто при использовании одного эталона для каждого класса) является более эффективным, когда объекты для классификации не наследуют никаких числовых свойств классовых эталонов. И наоборот, когда объекты для классификации могут обладать некоторыми выразительными числовыми свойствами нескольких классовых эталонов, тогда обучение вероятностных нейронный сетей на длинных эталонных матрицах является более эффективным, обеспечивая значительно высшую точность. Метод гладкого обучения для улучшения производительности оказывается неэффективным.

Ключевые слова: задача классификации, вероятностная нейронная сеть, эталонная матрица, поклассовая эталонная матрица, горизонтальная конкатенация матриц, гладкое обучение, зашумленные эталонные матрицы, длинная эталонная матрица, обобщённый поклассовый эталон.

Probabilistic neural networks (PNNs) are a very powerful tool for object state classification and diagnostics [1, 2]. They are much faster than multilayer perceptron networks: PNNs are trained extremely fast and they rapidly operate on objects (i. e., classify them). Another merit is that PNNs are relatively insensitive to outliers. The accuracy of PNNs are comparable to that of other feedforward neural networks [1, 3, 4]. As PNNs approach Bayes optimal classification, they can be even more accurate than multilayer perceptron networks. This is especially useful when objects to be classified have up to a few tens of features [4, 5].

Commonly, the training dataset for a PNN is just an $N \times M$ matrix whose columns represent classes. In fact, this is the pattern matrix in which column m is composed of the most expected feature values representing class m, $m=\overline{1, M}$. However, the class may have several patterns that can differ even in one feature value. Is making thus the pattern matrix as an average over class patterns optimal? Obviously, it depends on the classification problem and its conditions.

Background and analysis. The PNN has a simple architecture. It has two layers between the input and output layers. The weights in the pattern layer are determined by the pattern matrix. Thus, the pattern layer contains one neuron for each case in the training data set. Along with that, the target values are stored also. The Euclidean distance of the test case from the neuron's center point is computed and then the radial basis function kernel function is applied [1, 2]. In the summation layer, the pattern neurons add the values for the class they represent [2, 6]. Finally, the output layer compares the weighted scores for each target category accumulated in the pattern layer and the largest score corresponds to the target category predicted by this PNN [3, 6].

This simple architecture does not imply that the class pattern must be single. The reason for this can be described in two cases. First, the range of values for a class feature can be too wide. Then it is desirable to break it into subranges, each of which will give its own average and thus a few patterns for this class will be formed. Second, a class feature can have a finite number of its values, where every value has the same importance. Then it is impossible (at least, it would be incorrect) to calculate an average and use it in the respective single class pattern [3].

Whether it is efficient to use a concatenation of pattern matrices is an open question. The matter is that the criterion of the concatenation is not yet substantiated. On the one hand, wide ranges of input objects to be classified could be handled by the smooth training effective for two-layer perceptrons [7], although it is not yet proved to be effective for PNNs. The smooth training implies building the training set with increasingly noised matrices obtained by adding normal noise to the pattern matrix, where the noise is just a model to generate a wide variety of the class feature values. Then a long pattern matrix is built by concatenating single class pattern matrices, each of which corresponds to a definite noise level. On the other hand, the initial single class pattern matrix (without noise), which is an estimation of the expected pattern matrix, still can be used for training. The case with a few pattern matrices initially given slightly differs from this, but there are also two ways to do: either averaging over the pattern matrices or concatenating them in to a long pattern matrix.

The goal and tasks to achieve it. The goal of the research is to ascertain whether it is efficient to build PNNs on long pattern matrices. The criterion of the efficiency is performance of the PNN, i. e. either its accuracy or percentage of errors. To achieve the goal, the PNN performance will be studied on the case when the class is described by a few class patterns. The PNNs are to be tested for the two subcases: when objects generated by different class patterns are fed to the input, and when objects are generated by a generalized single class pattern. Additionally, the method of smooth training will be attached for the subcase of the generalized single class pattern. Every result with the long pattern matrix will be compared to the respective result with the single class pattern matrix.

Different pattern matrices. Let the single class pattern matrix be an $N \times M$ matrix

$$\mathbf{P}_0 = \frac{\psi\left(100 \cdot \Xi(N, M)\right) + 1}{101}, \tag{1}$$

where $\Xi(N, M)$ is an $N \times M$ matrix of pseudorandom numbers drawn from the standard uniform distribution on the open interval $(0; 1)$ and function $\psi(x)$ returns the integer part of number x (e. g., see [8]). Consider K class patterns, each of which is described by an $N \times M$ matrix $\mathbf{P}_k$:

$$\mathbf{P}_k = \mathbf{P}_0 + \sigma \cdot \mathbf{\Theta}(N, M) + \sigma_{\text{shift}} \cdot \text{rep}\left\{\mathbf{\Theta}(1, M), N, 1\right\} \text{ for } k = \overline{1, K}, \tag{2}$$

where $\Theta(N, M)$ is an $N \times M$ matrix of pseudorandom numbers drawn from the standard normal distribution (with zero mean and unit variance), σ is a standard deviation of the additive variations in initial pattern matrix (1), σ_{shift} is a standard deviation of the shifts in the class patterns, and operator $\text{rep}\{\mathbf{A}, n, m\}$ returns a tiled matrix whose rows and columns are of n and m copies of matrix $\mathbf{A}$, respectively.

A long pattern matrix $\mathbf{P}$ is formed as a set

$$\mathbf{P} = \{\mathbf{P}_k\}_{k=1}^{K} \tag{3}$$

which is factually a matrix built by horizontally concatenating matrices $\mathbf{P}_1$, ..., $\mathbf{P}_K$. Matrix (3) has the size $N \times (K \cdot M)$. The averaged pattern matrix (the single class pattern matrix) is

$$\tilde{\mathbf{P}} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{P}_k . \tag{4}$$

Denote a PNN trained on single class pattern matrix (4) by

$$S_{\sim} = \tau(\tilde{\mathbf{P}}, \mathbf{I}) \tag{5}$$

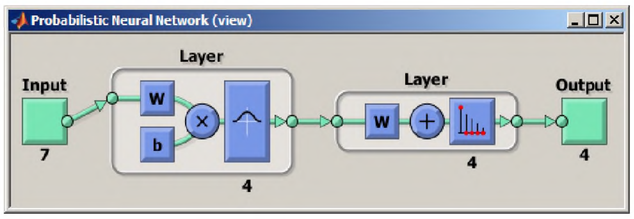
with $M \times M$ identity matrix $\mathbf{I}$ and a training operator τ. While training on long pattern matrix (3), a long target matrix

$$\mathbf{T} = \{\mathbf{I}\}_{k=1}^{K} \tag{6}$$

is used (again, this is the horizontal concatenation), whereupon a PNN

$$S_{\text{long}} = \tau(\mathbf{P}, \mathbf{T}) \tag{7}$$

is built (see examples of the PNN architecture view in Fig. 1).



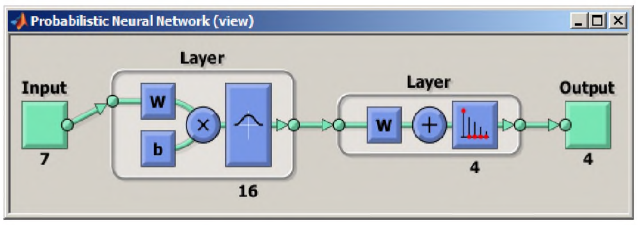


Figure 1 – The PNN architecture MATLAB view trained on the single class pattern matrix (the left side) and trained on the long pattern matrix by four class patterns (the right side, $K = 4$) for objects having 7 features by four classes (i. e., here $M = 4$)

To test the performance of PNNs (5) and (7), 500 test instances are randomly generated. Then the percentage of errors is

$$\gamma(S) = q/(5 \cdot M), \tag{8}$$

where S is a PNN and q is a total number of misclassifications. Obviously, error percentage (8) depends on the standard deviations in (2) showing how the pattern matrices are formed and it also depends on how test instances are generated. So, let

$$\sigma \in \{0.2, 0.3, 0.4, 0.5\} \text{ and } \sigma_{\text{shift}} = \sigma/2 . \tag{9}$$

The standard deviation value, which factually indicates the strength of variations added to the initial pattern matrix, will be varied from 0 to one of the four values in (9) by a step of one tenth of σ. The test instance is an $N \times M$ matrix

$$\mathbf{P}_{\text{test}} = \mathbf{P}_{k_*} + \sigma \cdot \boldsymbol{\Theta}(N, M) + \sigma_{\text{shift}} \cdot \text{rep}\{\boldsymbol{\Theta}(1, M), N, 1\} \tag{10}$$

by an integer k_* randomly chosen between 1 and K.

Fig. 2 shows error percentage (8) versus σ for the examples with only two pattern matrices for the long pattern matrix (square-dotted polylines) and for the averaged pattern matrix (dash-dotted polylines), where the six rows correspond to the following pairs of the number of features and the number of classes:

$$\begin{gathered} N=7,\ M=2, \\ N=7,\ M=4, \\ N=7,\ M=8, \\ N=11,\ M=2, \\ N=11,\ M=4, \\ N=11,\ M=8. \end{gathered}$$

The averaged polyline for the long pattern matrix is shown as a thicker circle-dotted polyline, and the averaged polyline for the averaged pattern matrix is shown as a thicker dash-dotted polyline. There are five polylines at each σ (separately for the long pattern matrix and for the averaged pattern matrix). The examples with $K = 4$ are shown in Figure 3 in the same style. In both example cases, the thick circle-dotted polyline is below the thick dash-dotted polyline. This means that the PNNs trained on the long pattern matrices perform with a lesser inaccuracy.

Another version of the test instance is when an object is generated by a generalized single class pattern, not "adhered" to a pattern like in (10). In this subcase, the test instance is an $N \times M$ matrix

$$\mathbf{P}_{\text{test}} = \mathbf{P}_0 + \sigma \cdot \boldsymbol{\Theta}(N, M) + \sigma_{\text{shift}} \cdot \text{rep}\{\boldsymbol{\Theta}(1, M), N, 1\}. \tag{11}$$

Note that matrix (11) is formed in the same manner as a pattern matrix by (2). Figure 4 shows error percentage (8) versus σ for the examples with only two pattern matrices (using the same styles of the polylines in Fig. 2), where test objects are generated by a generalized single class pattern according to (11). Fig. 5 shows the examples with $K = 4$. Unlike the previous two examples with test instances (10) for $k_* \in \{\overline{1, K}\}$, the subcase of the generalized single class pattern implying test instances (11) does not show any efficiency in using the long pattern matrix. Indeed, the thick circle-dotted polyline in Fig. 4 and 5 is generally above the thick dash-dotted polyline.

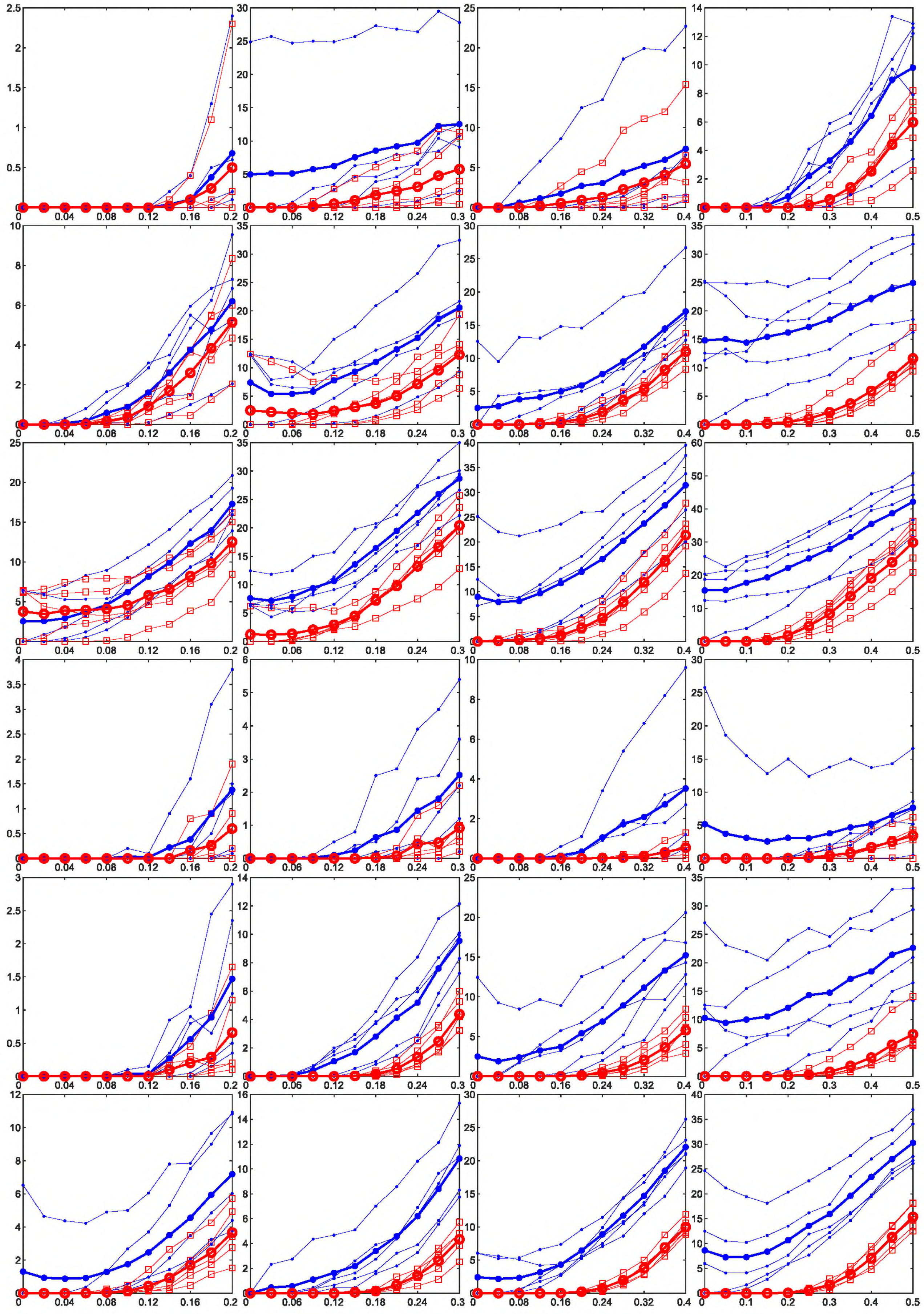


Figure 2 – The percentage of errors versus the strength of variations added to the initial pattern matrix for the examples of test objects generated by two class patterns

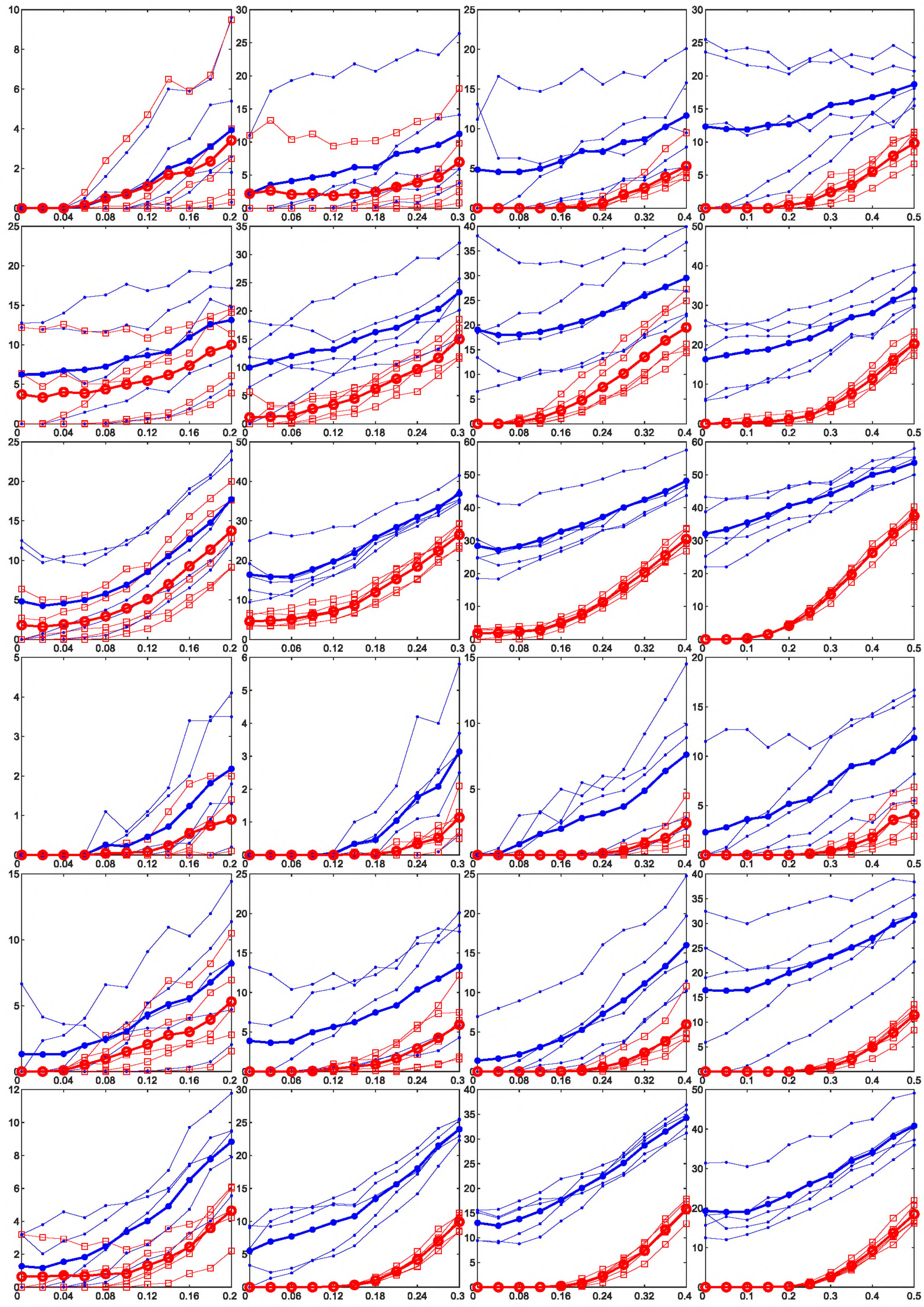


Figure 3 – The percentage of errors versus the strength of variations added to the initial pattern matrix by four class patterns, where a more distinct efficiency of the long pattern matrix is seen

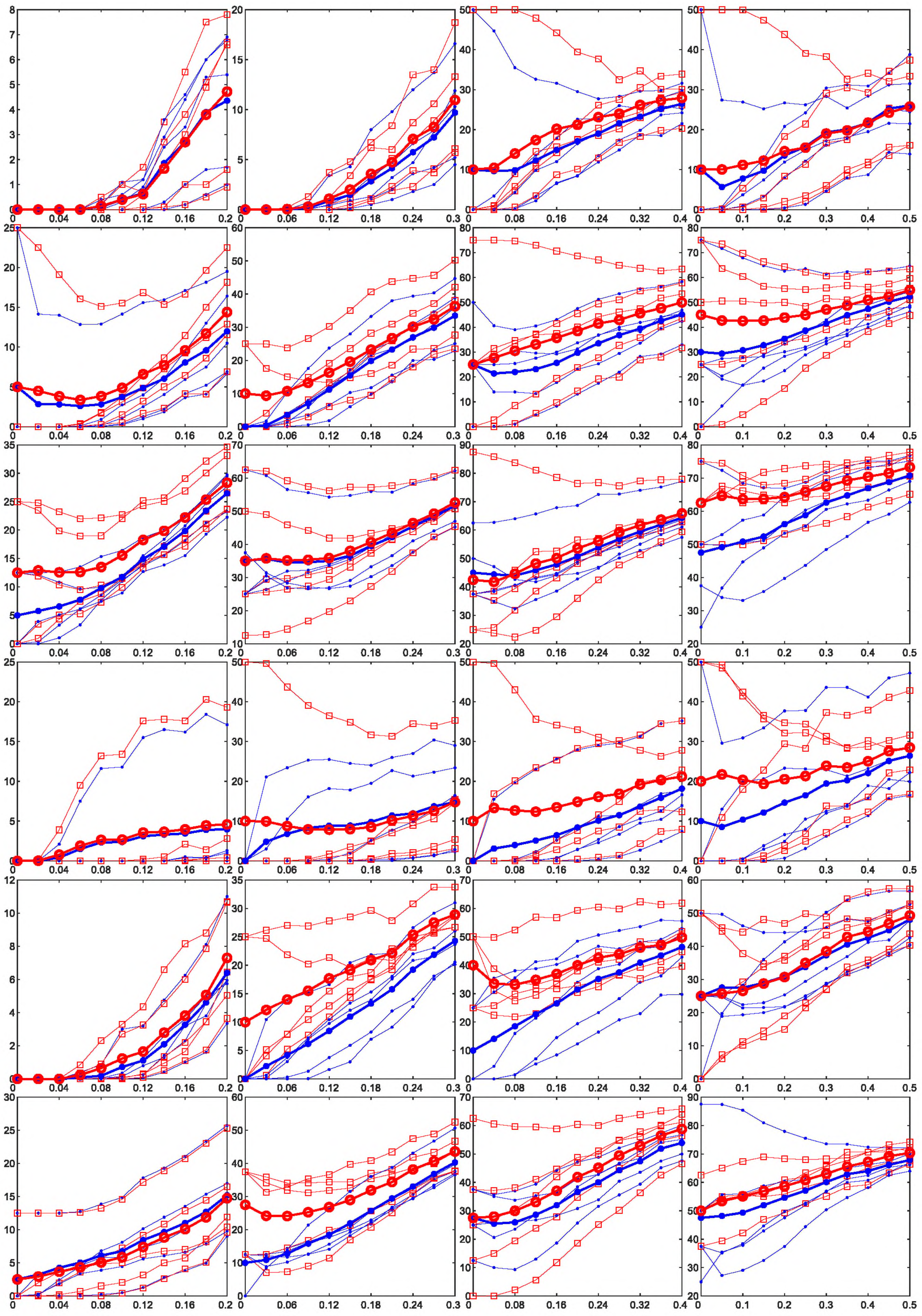


Figure 4 – The percentage of errors versus the strength of variations for the examples of test objects generated by a generalized single class pattern at $K = 2$

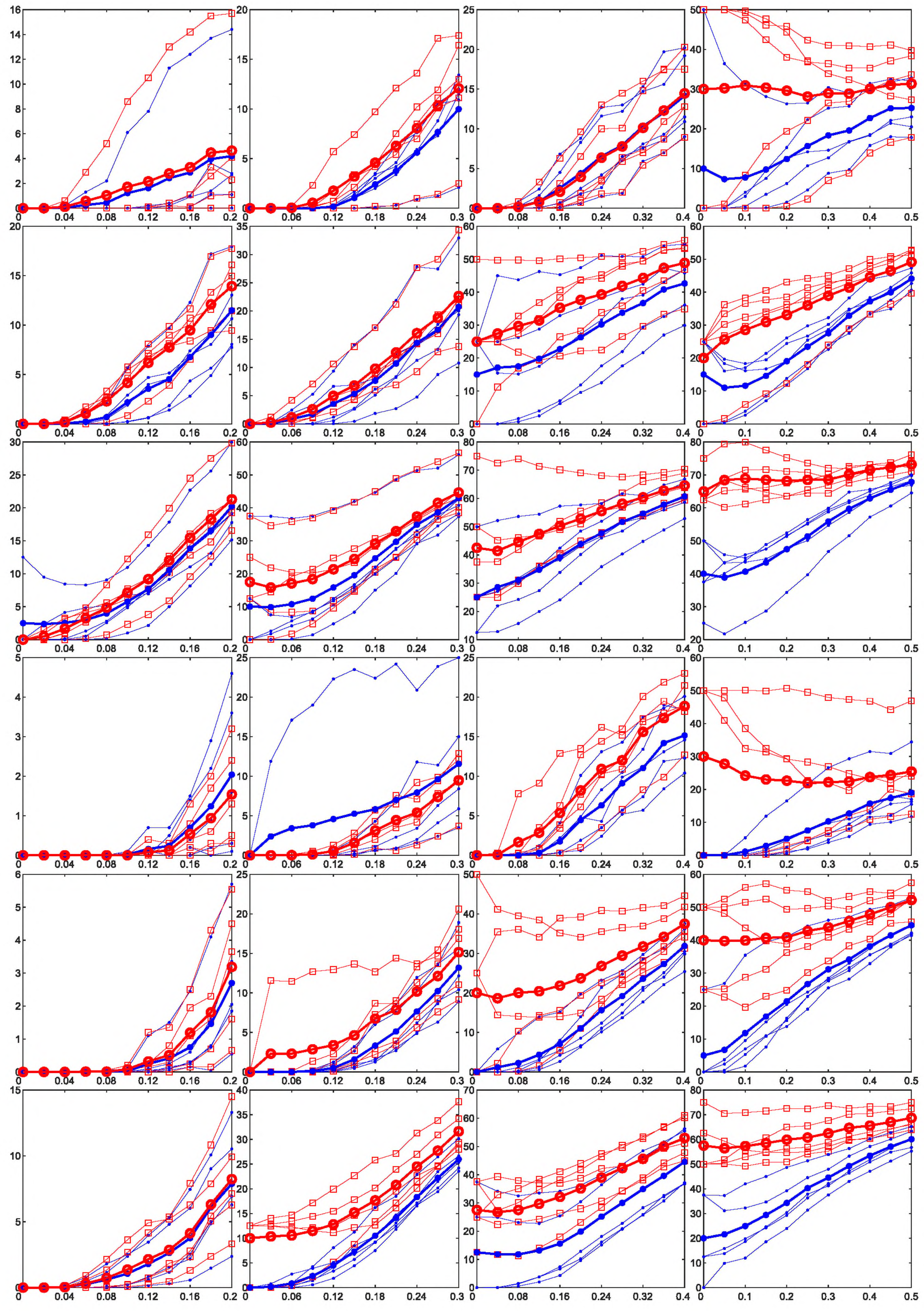


Figure 5 – The percentage of errors versus the strength of variations for test objects generated by a generalized single class pattern at $K = 4$ (inefficiency of the long pattern matrix is seen more)

Therefore, it is better to classify test instances (11) by simpler PNNs trained on averaged pattern matrix (4), which is the single class pattern matrix. Some exclusions may occur just as that in row 4 and column 2 (and, particularly, in column 1 of the same row) of Figure 5 corresponding to a binary classification problem of objects having 11 features. However, they are computational (pseudorandom) artifacts rather than a statistically reliable result.

Smooth training for PNNs. Consider the possibility to train PNNs to fit classifying test instances (11) by building a long pattern matrix in another way. This is the method of smooth training, which was experimentally proved to be rather effective for two-layer perceptrons [4, 7], at least in classifying objects having not a great number of features or small images. Smooth training implies building a training set as a concatenation (which, later on, is going to be a shuffled mixture of training samples, not arranged as in the concatenation) of a "pure" pattern matrix and a series of "noised" pattern matrices. The strength of the noise added to the pattern matrix is successively increased. In the case of smooth training for PNNs, the long pattern matrix is formed using the "pure" pattern matrix as single class pattern matrix (1) by the following concatenation rule:

$$\mathbf{P}_{\text{smooth}} = \left\{ \mathbf{P}_0, \left\{ \mathbf{P}_{0,j} \right\}_{j=1}^{F} \right\} \tag{12}$$

with a noised pattern matrix $\mathbf{P}_{0,j}$ at the j-th level of the noise by F such levels, where matrix $\mathbf{P}_{0,j}$ is formed similarly to test instances (11), i. e.

$$\mathbf{P}_{0,j} = \mathbf{P}_0 + \frac{j}{F} \cdot \sigma \cdot \boldsymbol{\Theta}(N, M) + \frac{j}{F} \cdot \sigma_{\text{shift}} \cdot \text{rep}\left\{ \boldsymbol{\Theta}(1, M), N, 1 \right\} \quad \text{for} \quad j = \overline{1, F}. \tag{13}$$

Just as it is in operations (3) and (6), the operation on the right side of (12) is a horizontal concatenation. Obviously, in training on long pattern matrix (12) with noised pattern matrices (13), a long target matrix

$$\mathbf{T}_{\text{smooth}} = \left\{ \mathbf{I} \right\}_{j=1}^{F+1} \tag{14}$$

is used (this is the horizontal concatenation), whereupon a PNN

$$S_{\text{smooth}} = \tau\left(\mathbf{P}_{\text{smooth}}, \mathbf{T}_{\text{smooth}} \right) \tag{15}$$

is built. Fig. 6 shows error percentage (8) versus σ for the examples of PNN (15) compared to the error percentage of PNNs

$$S_0 = \tau\left(\mathbf{P}_0, \mathbf{I} \right) \tag{16}$$

trained on single class pattern matrix (1). As previously, the error percentage for the long pattern matrix (with eight noise levels) is shown by square-dotted polylines, and the error percentage for the single class pattern matrix is shown by dash-dotted polylines (each of the six rows corresponds to the pairs of the number of features and the number of classes mentioned above). The averaged polyline for the long pattern matrix is presented by a thick circle-dotted polyline, and the averaged polyline for the single class pattern matrix is presented by a thick dash-dotted polyline. It is quite clear that PNNs trained smoothly do not have better performance than "ordinary" PNNs (16) trained without any complications (concatenations). Nevertheless, the advantage of the simpler PNNs here is less significant than that in the case of the averaged pattern matrix (Fig. 4 and 5).

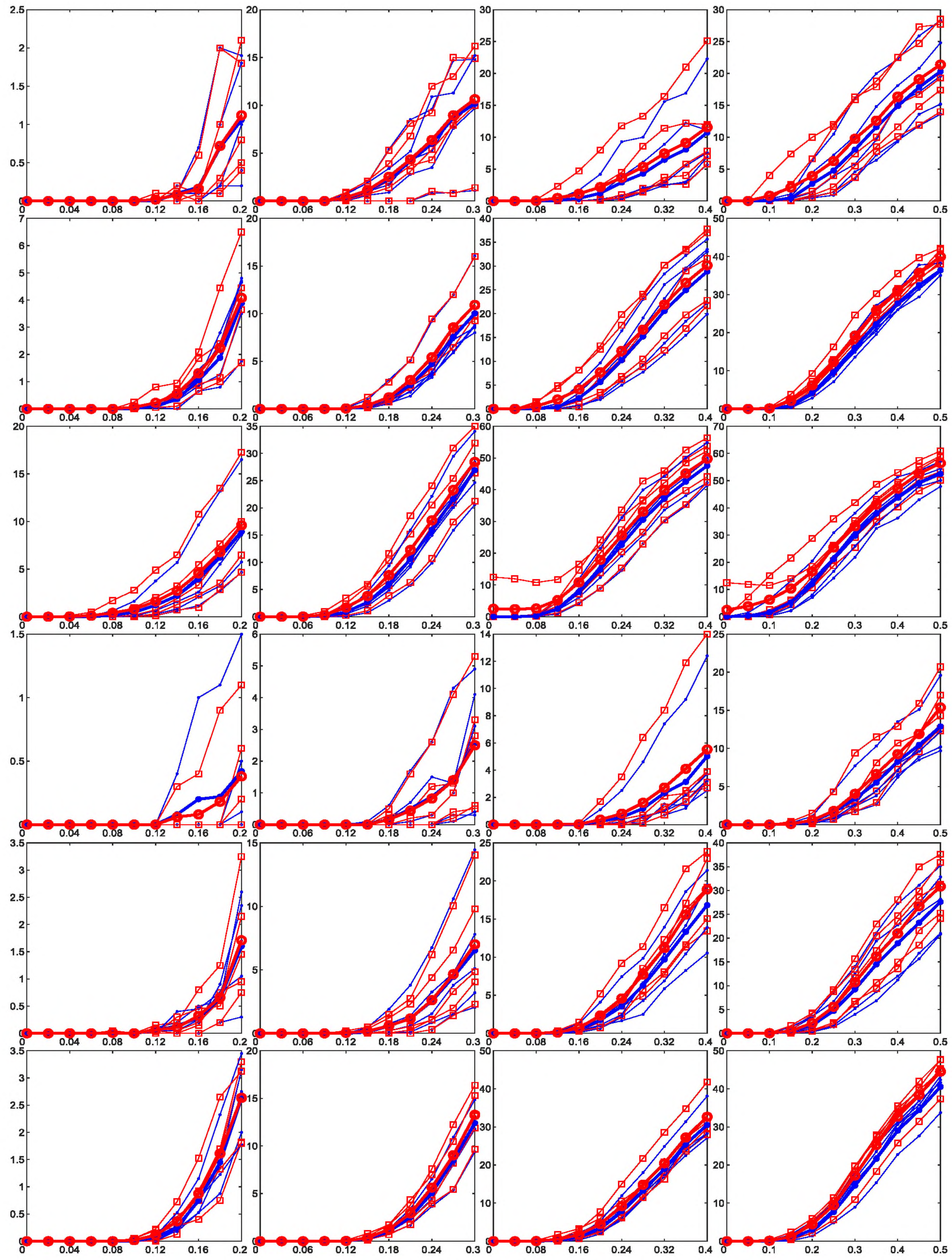


Figure 6 – The percentage of errors versus the strength of variations by smooth training with $F = 8$

Discussion and conclusions. The question of efficiently building PNNs on long pattern matrices is openly arguable. On one side, when test objects are generated by a few different class patterns, Fig. 2 and 3 confirm that PNNs trained on long pattern matrices perform a way better than PNNs trained on the single class pattern matrix simplified owing to averaging over the available (known) class patterns. On the other side, when generation of test objects does not include any "prints" of the patterns, Fig. 4 and 5 confirm that the simplest PNNs (those which are trained on the single class pattern matrix) perform even slightly better than the PNNs trained on long pattern matrices. Furthermore, the smooth training method appears inefficient in improving the PNN performance for classifying such "pattern-print-less" objects. Inasmuch as it is not always possible to predict the exact type of the object at the PNN input, the training method based on concatenating different pattern matrices cannot be certainly affirmed to be efficient.

Therefore, it is efficient to train PNNs on the single class pattern matrix (obtained either by averaging over the available pattern matrices or just by using one pattern per class) when the objects to be classified do not inherit any class pattern numerical properties. On the contrary, when the objects to be classified may have some distinct numerical properties of a few class patterns, then training PNNs on long pattern matrices is more efficient ensuring noticeably lesser percentage of errors (i. e., a higher accuracy).

REFERENCES:

1. Annema J. Feed-Forward Neural Networks. Springer, 1995: 238 p.
2. Masters T. "Probabilistic Neural Networks," in: Practical Neural Network Recipies in C++. Academic Press, 1993, pp. 201 – 222.
3. Dreyfus G. Neural Networks: Methodology and Applications. Springer-Verlag Berlin Heidelberg, 2005: 498 p.
4. Romanuke V. V. "Setting the hidden layer neuron number in feedforward neural network for an image recognition problem under Gaussian noise of distortion." Computer and Information Science, vol. 6, no. 2, 2013, pp. 38 – 54.
5. Romanuke V. V. "Combining the backpropagation algorithm training functions for a bit map character recognition problem under noise with feed-forward neural network." Scientific Bulletin of CNU. Series: Computer systems and components, vol. 3, iss. 1, 2012, pp. 75 – 80.
6. Haenni R., Romeijn J.-W., Wheeler G., Williamson J. Probabilistic Logics and Probabilistic Networks. Springer Netherlands, 2011: 155 p.
7. Romanuke V. V. "Dependence of performance of feed-forward neuronet with single hidden layer of neurons against its training smoothness on noised replicas of pattern alphabet." Herald of Khmelnytskyi National University. Technical sciences, no. 1, 2013, pp. 201 – 206.
8. Romanuke V. V. "Decision making criteria hybridization for finding optimal decisions' subset regarding changes of the decision function." Journal of Uncertain Systems, vol. 12, no. 4, 2018, pp. 279 – 291.

ЛІТЕРАТУРА:

1. Annema J. Feed-Forward Neural Networks. – Springer, 1995. – 238 p.
2. Masters T. Probabilistic Neural Networks / Practical Neural Network Recipies in C++. – Academic Press, 1993. – P. 201 – 222.
3. Dreyfus G. Neural Networks: Methodology and Applications. – Springer-Verlag Berlin Heidelberg, 2005. – 498 p.
4. Romanuke V. V. Setting the hidden layer neuron number in feedforward neural network for an image recognition problem under Gaussian noise of distortion / V. V. Romanuke // Computer and Information Science. – 2013. – Vol. 6, No. 2. – P. 38 – 54.
5. Romanuke V. V. Combining the backpropagation algorithm training functions for a bit map character recognition problem under noise with feed-forward neural network / V. V. Romanuke // Scientific Bulletin of CNU. Series: Computer systems and components. – 2012. – Vol. 3, Iss. 1. – P. 75 – 80.
6. Haenni R., Romeijn J.-W., Wheeler G., Williamson J. Probabilistic Logics and Probabilistic Networks. – Springer Netherlands, 2011. – 155 p.
7. Romanuke V. V. Dependence of performance of feed-forward neuronet with single hidden layer of neurons against its training smoothness on noised replicas of pattern alphabet / V. V. Romanuke // Herald of Khmelnytskyi National University. Technical sciences. – 2013. – No. 1. – P. 201 – 206.
8. Romanuke V. V. Decision making criteria hybridization for finding optimal decisions' subset regarding changes of the decision function / V. V. Romanuke // Journal of Uncertain Systems. – 2018. – Vol. 12, No. 4. – P. 279 – 291.

DOI: 10.33243/2518-7139-2019-1-2-86-97