Journal of Intelligent Learning Systems and Applications, 2015, 7, 75-86
Published Online August 2015 in SciRes. http://www.scirp.org/journal/jilsa
http://www.scirp.org/Journal/PaperInformation.aspx?doi=10.4236/jilsa.2015.73007

Disparity in Intelligent Classification of Data Sets Due to Dominant Pattern Effect (DPE)

Mahmoud Zaki Iskandarani

Faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan
Email: m.iskandarani@hotmail.com

Received 29 April 2015; accepted 7 July 2015; published 10 July 2015

Abstract

A hypothesis of the existence of dominant pattern that may affect the performance of a neural based pattern recognition system and its operation in terms of correct and accurate classification, pruning and optimization is assumed, presented, tested and proved to be correct. Two sets of data subjected to the same ranking process using four main features are used to train a neural network engine separately and jointly. Data transformation and statistical pre-processing are carried out on the datasets before inserting them into the specifically designed multi-layer neural network employing Weight Elimination Algorithm with Back Propagation (WEA-BP). The dynamics of classification and weight elimination process is correlated and used to prove the dominance of one dataset. The presented results proved that one dataset acted aggressively towards the system and displaced the first dataset making its classification almost impossible. Such modulation to the relationships among the selected features of the affected dataset resulted in a mutated pattern and subsequent re-arrangement in the data set ranking of its members.

Keywords

Pattern Recognition, Neural Networks, Ranking, Datasets, Weight Elimination, Pruning, Mutation, Genetic Algorithms

1. Introduction

The main problem in neural network design and application for many years was the choice of an appropriate network for a given application. In general, network size affects network complexity and learning time, but most importantly it affects the generalization capabilities of the network to accurately predict results for data outside the used training sets. A network with unnecessary complex structure and large connections will definitely fit the training data patterns in the training set but performs very poorly on unknown patterns. On the other hand, a network having a structure simpler than necessary cannot give good results even for patterns included in its training sets.

Emphasis has been given to the development of algorithms that reduce network size by modifying not only the connection weights but also the network structure during training, such as pruning, where unnecessary nodes or connections are eliminated using a sensitivity based measure to indicate how the solution is affected from this change, or modifying the error function of the network in such a way that unnecessary connection weights are pushed gradually to zero during training in a decay like process. Other algorithms start with a small network and gradually build it up by increasing the number of nodes or connections during training. In addition, weight sharing through assigning identical set of weights to each node in a group is also researched and tried. All efforts aimed at improving network design and reducing size and training times without affecting its generalization [1]-[4].

Network size and connection structure are definitely important in deciding the functionality of the design and the ability to optimize through pruning techniques and algorithms. However, research always approached the used data in terms of its size, validity, selected features, training samples, and cross validation, but hardly touched the issue and possibility that another important factor could lead to over fitting and under fitting of examined data sets and their pattern besides network size, number of connections, training algorithm, cross validation, and pruning.

The hypothesis that a data set processed under the same conditions as the rest of datasets, which will have similar effects on another dataset and the neural algorithm performance in classification and prediction, to the previously mentioned factors in negatively influencing the performance of the designed neural network structure, should be proved and catered for alongside the all-time concerns of neural network designers. If not recognized, this dataset will have devastating effects on the designed neural net, and will disable the network from correct functionality. It will also result in endless trials to modify the design and used algorithms, as the focus will be on traditional variables that lead to such over fit-under fit behavior of the neural system [5]-[7].

Studying datasets that cause such neural network malfunction is very important as they will reveal specific information which can be traced back to the original process that results in their production. The severe effect of a dataset on the behavior of a well-designed and optimized neural network, when used as a training data set, is worth analyzing [8] [9].

In this paper, a clear evidence of the existence of the Dominant Pattern (DP) and its undesirable effects in creating chaos and randomness among the rankings of elements in datasets are presented and proved.

2. Methodology

The goal of the proposed approach is to discover the effect of a dominant pattern on the accuracy of classification of neural systems in order to provide better solutions, and to identify hostile datasets that contribute in addition to other factors such as the size, number of layers, number of nodes and interconnections to the failure of the system to converge and correctly classify, which will result in a distorted pattern. Thus, permutations of neural network trials (N), using various parameter settings, until the most promising and well optimized structure is obtained as shown in **Figure 1**. Instead of training each network separately for each dataset, the same structure is used for all permutations and possible combinations, resulting in identical testing conditions [10].

Table 1 and **Table 2** show the post processed datasets used for training the neural infrastructure in **Figure 1**.

The four selected features are not apparently correlated and the relationship between them is non-linear. Each feature is ranked under separate domain that has specific conditions. The final ranking or positioning is achieved through correlation of the four ranked features that describe a position of a data record within a data set. The feature ranking can be expressed as in Equation (1) below:

$$\text{Position} = f\left(\text{Feature}_1, \text{Feature}_2, \text{Feature}_3, \text{Feature}_4, \cdots, \text{Feature}_i\right) \tag{1}$$

where:

$$\text{Feature}_i = \underset{\text{Record 1}}{\overset{\text{Record } K}{\text{Rank}_i}}\left(\text{Domain}_i\right) \tag{2}$$

The Weight Elimination Algorithm (WEA), which is concerned with weights pruning is expected to help uncovering of the existence of dominant set. The neural structure would be optimized and fine-tuned using WEA.

Table 1. Training dataset 1.

Position	Feature_1	Feature_2	Feature_3	Feature_4
1	2	3	1	1
2	4	5	3	4
3	6	4	7	2
4	9	6	2	3
5	1	1	5	18
6	13	8	10	5
7	5	2	9	23
8	11	12	8	8
9	16	9	14	6
10	8	11	12	14
11	19	7	13	13
12	7	10	11	34
13	15	13	15	7
14	14	22	4	10
15	18	19	6	17
16	10	16	20	11
17	24	17	21	9
18	22	14	17	19
19	3	32	35	16
20	17	30	16	12
21	23	15	23	29
22	12	25	36	21
23	20	18	22	28
24	30	24	28	15
25	25	21	18	27
26	33	20	32	20
27	21	26	19	26
28	29	23	30	24
29	26	28	27	33
30	32	27	24	30
31	34	29	25	31
32	28	33	26	32
33	27	35	31	25
34	35	34	29	22
35	36	31	33	35
36	31	36	34	36

Table 2. Training dataset 2.

Position	Feature$_1$	Feature$_2$	Feature$_3$	Feature$_4$
1	1	1	1	1
2	3	3	2	3
3	2	2	3	4
4	7	7	5	2
5	4	8	7	5
6	11	6	8	6
7	10	9	4	17
8	6	5	14	38
9	26	4	32	37
10	8	10	10	10
11	5	11	26	25
12	13	12	11	28
13	19	13	17	8
14	14	16	21	29
15	25	15	12	36
16	17	20	9	27
17	18	22	22	7
18	15	21	13	24
19	22	18	20	20
20	20	19	38	15
21	31	14	27	9
22	12	23	37	14
23	16	25	16	31
24	35	27	6	33
25	9	28	31	35
26	21	26	39	16
27	39	24	24	22
28	40	17	15	26
29	38	29	18	18
30	27	31	30	34
31	34	30	35	12
32	28	32	36	13
33	23	36	33	39
34	32	33	25	23
35	36	35	23	11
36	37	34	34	40
37	24	36	40	21
38	29	37	29	32
39	30	39	28	30
40	33	40	19	19

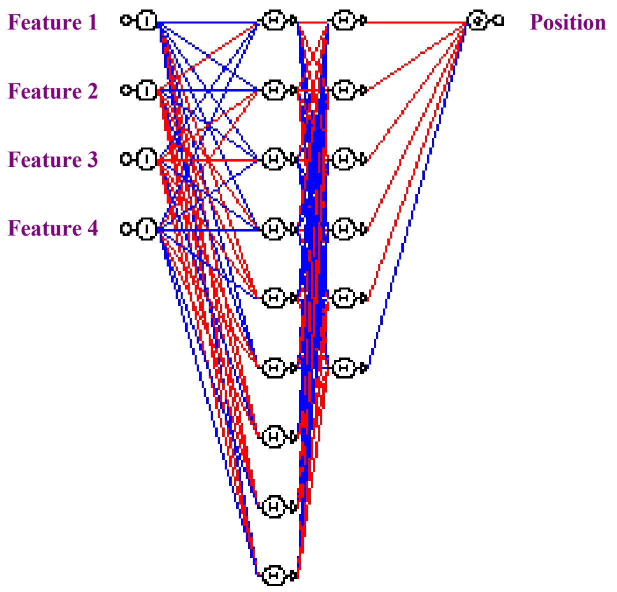

Figure 1. Neural network infrastructure.

Weight Elimination Algorithm is used to carry out weight decay process. It minimizes a modified error function which is formed by adding a penalty, liability or cost term to the original error function of the used algorithm [11].

The liability variable in weight decay penalizes large weights, as it causes weights under consideration to converge to smaller absolute values. Large weights can negatively affect generalization according to their position in the network. If the weights with large values are between input layer and hidden layer, they can result in the output function to be too rough, possibly with near discontinuities. However, if they lie between the hidden layer and the output layer, they can result in wild outputs beyond the range of the data. Hence, large weights can cause excessive variance of the output and instability of the neural structure, as their values and instability will be outside the range of the output activation function. Weights size is very important and in some cases will have more effect than the number of weights in determining generalization.

Weight elimination mathematical representation is based on Ridge regression principles, whereby it describes the dynamic changes in neural network convergence through error functions. The overall weight elimination error function is presented by Equation (3), and it consists of two parts [12] [13]:

I. Initial error function described by Equation (4).

II. Penalty error function described by Equation (5)

$$E_{WE} = E_{Initial} + E_{Penalty} \tag{3}$$

$$E_{Initial} = \frac{1}{2}\sum_{k}(d_k - o_k)^2 \tag{4}$$

$$E_{Penalty} = \xi\left(\sum_{jk}\frac{\left(\frac{w_{jk}}{w_{neural}}\right)^2}{1+\left(\frac{w_{jk}}{w_{neural}}\right)^2}\right) \tag{5}$$

where;

E_{WE} : The combined overhead function that includes the initial overhead function, $E_{Initial}$ and the weight-elimination term $E_{Penalty}$.

ξ : The weight-reduction factor.

w_{jk} : Represents the individual weights of the neural network model.

w_{neural} : A scale parameter computed by the WEA.

d_k : The desired Output: It represents the wanted position as a function of the four used features, as indicated in **Figure 1**, **Table 1** and **Table 2**.

o_k : The actual Output: It represents the actual position, as a function of the four used features, as indicated in **Figure 1**, **Table 1** and **Table 2**.

The dynamic weight changes is calculated through a modified version of the gradient descent algorithm as shown in equation (6)

$$\Delta w_{jk} = \left(-\eta \frac{\partial E_{Initial}}{\partial w_{jk}}\right) - \left(\xi \frac{\partial E_{Penalty}}{\partial w_{jk}}\right) \tag{6}$$

where;

η : The Learning Rate (between 0 and 1)

The parameter, w_{neural}, is a scale parameter computed by the WEA, and chosen to be the smallest weight from the last epoch or set of epochs to force small weights to zero. w_{neural} guides the computing algorithm to find solutions with either fewer large weights or many small weights, depending on the w_{neural} values small or large.

Two clear cases can be realized:

$$\text{Case 1: } w_{neural} \to \infty \Rightarrow E_{Penalty} \to 0 \Rightarrow E_{WE} = E_{Initial} \tag{7}$$

Here the algorithm needs to drive the weights values down and obtain small weights in large numbers.

$$\text{Case 2: } w_{neural} \to 0 \Rightarrow E_{Penalty} \to 1 \Rightarrow E_{WE} = E_{Initial} + \xi \tag{8}$$

Here the algorithm needs to keep large weights in small numbers.

The error sum is carried out over all training examples and overall output nodes, with the initial error representing the complexity of the network as a function of the weight magnitude, which is relative to the scale parameter w_{neural}. The weight factor ξ, determines the importance of the network complexity with respect to network performance over training sets, and can be adjusted during training and computed from equation (9).

$$\xi = \xi_0 \exp\left(-\gamma\left(1 - R_P\right)\right) \tag{9}$$

where; ξ_0 is a scaling factor and can be set to 1, γ is multiplication constant, and R_P is the ratio of correctly classified patterns from testing sets over the overall number of patterns. ξ dynamically responds to R_P changes, so when R_P increases, so does ξ, which leads to more contributions by $E_{Penalty}$ in driving small weights towards zero to further increase generalization. However, when R_P decreases, it will result in ξ approaching a very small value, hence, suppressing the contribution by $E_{Penalty}$ to the overall error expression and the neural structure would be at a bad state.

The role of the weight-reduction factor is to determine the relative importance of the weight-elimination term. Larger values of ξ push small weights to further reduce their size. Small values of ξ will not affect the network. The choice of ξ should be optimized such that it is not too large or too small. Too values of ξ will result in fast decay of small weights, and too small values of ξ will eliminate the process of pruning.

Selecting the stop point is critical to avoid over fitting and over pruning. When performance is poor, corresponding connections to weaker weights are removed, which will lead to redundant weights. To remove a weight, the algorithm, also looks at the relationship between w_{jk} and w_{neural}, and when the relationship leads to $E_{Penalty}$ becoming very small, weights are both driven to smaller values and removed. Weight Elimination Algorithm (WEA) is a bidirectional Bottom-Up, Top-Down pruning algorithm, starts with a simple, then complex network and drives unnecessary weights during training towards zero.

Preprocessing of the used training sets is carried out as follows:

1) Statistical Data Transformation (SDT) and Data Clustering (DC);

2) Each selected feature is a function of multi correlated variables that contribute to the positioning of the data record within the dataset.

The position of each record within a data set is a function of the correlation between the four features and between each record and the next one and the one before. To achieve correct and reliable analysis showing the disparity due to dominant pattern effect, the same neural structure is used for individual and joined patterns. The designed network needed to be complex as after few trials with simple designs and small number of nodes, the network did not converge with very high Mean Squared Error (MSE) [14] [15].

3. Results

Figure 2 and **Figure 3** show initial statistical representation of the features in both dataset 1 and dataset 2. **Table 3** and **Table 4** show the classified and predicted data using WEA.

Table 3. Predicted dataset 1 as a function of training datasets 1 and 2.

Reference	Predicted Positions for Dataset 1					
	Training with Dataset 1		Training with Dataset 2		Training with Datasets 1 & 2	
	Testing					
	Dataset 1	Datasets 1 & 2	Dataset 1	Datasets 1 & 2	Dataset 1	Datasets 1 & 2
1	1	1	1	1	1	1
2	2	2	3	3	3	2
3	3	3	5	5	4	3
4	4	4	2	2	5	4
5	5	5	9	9	5	5
6	6	5	7	7	7	6
7	7	7	7	7	8	7
8	8	7	8	8	8	8
9	9	8	9	9	10	9
10	10	8	12	12	8	10
11	11	10	6	6	12	11
12	12	12	12	12	13	12
13	13	12	13	13	14	13
14	14	13	8	8	20	14
15	15	14	**11**	11	24	15
16	16	15	14	14	13	16
17	17	16	**17**	17	20	17
18	18	16	**16**	16	23	18
19	19	18	27	27	28	19
20	20	19	23	23	30	20
21	21	20	17	17	22	21
22	22	21	22	22	33	22
23	23	21	**18**	18	26	23
24	24	21	**26**	26	26	24
25	25	23	23	23	26	25
26	26	22	27	27	28	26
27	27	25	**25**	25	29	27
28	28	25	26	26	29	28
29	29	28	26	26	32	29
30	30	28	30	30	32	30
31	31	29	32	32	34	31
32	32	30	**34**	34	37	32
33	33	29	35	35	37	33
34	34	28	35	35	38	34
35	35	32	**34**	34	38	35
36	36	33	38	38	39	36

Table 4. Predicted dataset 1 as a function of training datasets 1 and 2.

Reference	Predicted Positions for Dataset 2					
	Training with Dataset 2		Training with Dataset 1		Training with Datasets 1 & 2	
	Testing					
	Dataset 2	Datasets 1 & 2	Dataset 2	Datasets 1 & 2	Dataset 2	Datasets 1 & 2
1	1	1	1	1	1	1
2	2	2	1	1	2	2
3	3	3	1	1	3	3
4	4	4	4	4	4	4
5	5	5	5	5	5	5
6	6	6	3	3	6	6
7	7	7	14	14	7	7
8	8	8	5	5	8	8
9	9	9	14	14	9	9
10	10	10	6	6	10	10
11	11	11	13	13	11	11
12	12	12	17	17	12	12
13	13	13	12	12	13	13
14	14	14	16	16	14	14
15	15	15	20	20	15	15
16	16	16	20	20	16	16
17	17	17	17	17	17	17
18	18	18	21	21	18	18
19	19	19	19	19	19	19
20	20	20	19	19	20	20
21	21	21	16	16	21	21
22	22	22	18	18	22	22
23	23	23	23	23	23	23
24	24	24	19	19	24	24
25	25	25	20	20	25	25
26	26	26	21	21	26	26
27	27	27	19	19	27	27
28	28	28	17	17	28	28
29	29	29	17	17	29	29
30	30	30	29	29	30	30
31	31	31	28	28	31	31
32	32	32	25	25	32	32
33	33	33	28	28	33	33
34	34	34	26	26	34	34
35	35	35	27	27	35	35
36	36	36	33	33	36	36
37	37	37	27	27	37	37
38	38	38	32	32	38	38
39	39	39	32	32	39	39
40	40	40	25	25	40	40

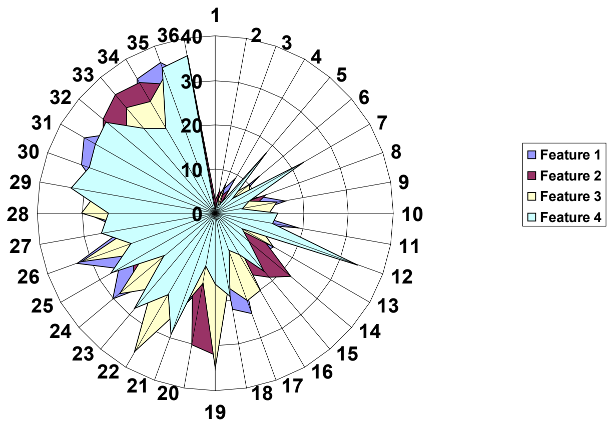

Figure 2. Dataset 1 features spread.

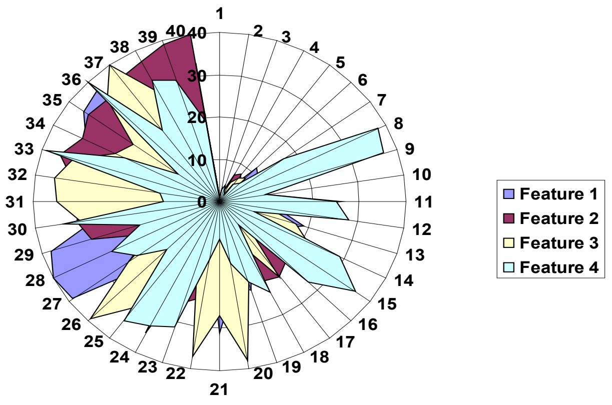

Figure 3. Dataset 2 features spread.

The approach for this work included three elements:

1) Collecting and pre-processing data to represent ranking or position of data records in each of the datasets;

2) Selecting an appropriate neural network algorithm to uncover the dominant pattern;

3) Correlating both data selection to the dynamics of the selected neural system.

Training of the designed neural network structure is carried out using three different sets:

I. Dataset 1;

II. Dataset 2;

III. Combined datasets 1 & 2.

4. Discussion

From **Figure 2** and **Figure 3**, it is deduced that even though both datasets belong to the same general parent category, with exactly same rules applied to produce the sets and with many general common features in between them, dataset 2 seems to possess an abnormal spread that goes beyond the expected signature of the parent dataset that produced both sets.

Testing of the system for generalization, classification, and prediction, is carried out using both dataset 1 and dataset 2. This approach together with WEA, guarantee uncovering of which dataset is dominant as a result of the neural network output when either dataset is tested against all three permutations or sets, as shown in **Table 3** and **Table 4**, while **Table 5** summarizes and provides an evidence that dataset 2 is dominant as it dominates the training, testing process and the results obtained for dataset 1, whenever both datasets are either present in training or testing. Dataset 1 shown to be recessive with no effect on the way system behaves.

Table 5 is transformed in logic table presented in **Table 6**. From **Table 6**, it is clear that dataset 1 has no effect on dataset 2, but only on itself, as a perfect prediction and classification match for dataset 1 occurs when

Table 5. Pattern matching.

Testing Datasets	Training Datasets		
	Dataset 1	Dataset 2	Datasets 1 & 2
Dataset 1	Match	No Match	No Match
Dataset 2	No Match	Match	Match
Datasets 1 & 2	No Match	Match	Match

Table 6. Logical representation: pattern matching.

Testing Datasets	Training Datasets		
	Dataset 1 → a	Dataset 2 → b	Datasets 1 & 2 → c
Dataset 1 → a	1	0	0
Dataset 2 → b	0	1	1
Datasets 1 & 2 → c	0	1	1

dataset 1 is used for both training and testing. For dataset 2, the table shows that whenever it is present in the training, alone or in combination with dataset 1, the system classifies accurately for either dataset 1 or the combined datasets 1 & 2. This is a solid proof that dataset 2 plays the dominant pattern with dataset 1 as the recessive pattern, which supports the initial plots in **Figure 2** and **Figure 3** regarding their initial characteristics.

Looking back at **Table 3** and **Table 4**, it is realized that the mutation and disturbance in the results obtained for dataset 1 by dataset 2 when dataset 1 is used for training than the mutation in the results for dataset 2 by dataset 1 when dataset 1 is used for training. Such evidence goes along way to show the superior effect of dataset 2 on dataset 1 and on the combined sets.

From Table (6), three logical relationships can be represented:

$$1.\ \{a,\bar{b},c\} \Rightarrow \{a,\bar{b},\bar{c}\} \tag{10}$$

$$2.\ \{\bar{a},b,\bar{c}\} \Rightarrow \{\bar{a},b,c\} \tag{11}$$

$$3.\ \{\bar{a},\bar{b},c\} \Rightarrow \{\bar{a},b,c\} \tag{12}$$

It is noticed from the logical expressions that {a} occurs only once as a positive logic, but {b, c} occurs twice as positive logic with {a} in both cases in negation status. This further supports the dominant pattern effect with mutation consequences on dataset 1 or pattern {a}. Now, considering the dynamics of WEA, it is clear that the presence of dominating pattern affected and displaced the weight elimination process, whereby, excessive variance is caused when prediction and classification of dataset 1 is carried out after the neural structure is subjected to the influence to dataset 2. This dominating pattern affected weight distribution and ability of the system to identify the pattern associated with dataset 1, with total ability to perfectly reproduce the pattern associated with dataset 2 even when both datasets are used for training.

From **Table 3** and **Table 4**, it is clear that dataset 1 has 2 matches out of 6 trials, and dataset 2 has 4 matches out of 6 trials, this gives:

$$M_{P_1} = \frac{2}{6} = \frac{1}{3} \tag{13}$$

$$M_{P_2} = \frac{4}{6} = \frac{2}{3} \tag{14}$$

$$M_{Total} = \frac{4}{6} + \frac{2}{6} = \frac{6}{6} = 1 \tag{15}$$

These results mean,

Since M_{P_1} has some contribution to the value, then M_{Total} both dataset 1 and dataset 2 share some common properties, but both did not originate from the same place, neither they are subsets of a larger dataset, hence, the domination of dataset 2 through its pattern.

If both datasets do not hold similar properties or been processed using similar features, then one of them should have Zero contribution to expression in (15).

If both datasets have equal contribution with no one pattern dominates, then it is expected that each one would have a 50% contribution value to the total pattern recognition.

If both datasets have no common features and not subsets of an original set, then it is expected that one of them would have Zero contribution with the other having 100% contribution value.

From above, we deduce the expression in (16) and (17):

$$M_{P_i} = \frac{\text{Number of Correctly Recognized Patterns}}{\text{Total Number of Tested Patterns}} \tag{16}$$

$$M_{Total} = \sum_{i=1}^{n} M_{P_i} \leq 1 \tag{17}$$

This approach can be used to detect intruding and out of place datasets and the ones that do not belong to the general characteristics of the prescribed behavior of collection of datasets. Hence, it can be used as a filter and isolator which detects and removes datasets that suffer damage or that might have damaging effect on the overall system. Also, it can be used to analyze change in patterns and their effect on the rest of the holding main matrix.

Figure 4 shows the Mean Squared Error (MSE), computed for the same neural structure with training datasets:

I: Dataset 1;

II: Dataset 2;

III. Combined datasets 1 and 2.

It is evident that when dataset 2 has larger MSE compared to dataset 1, however, the interesting result is when both datasets are used for training, the MSE value dramatically increased. This behavior is consistent with the effect that dataset 2 has on the system dynamics and on dataset 1, where it unbalances the system due to its dominance and mutation properties [14] [15].

5. Conclusions

This work proved a hypothesis that a dominant pattern would negatively affect the performance of a neural based pattern recognition system and would influence its operational behavior and dynamics in terms of its capabilities to classify, predict, and generalize.

Classification and prediction results of the WEA-BP showed clear dominance for dataset 2 on dataset 1. Such negative effect of dataset 2 through its pattern representation affected the weight elimination process and its corresponding weight change dynamics with subsequent mutation to the obtained results for dataset 1.

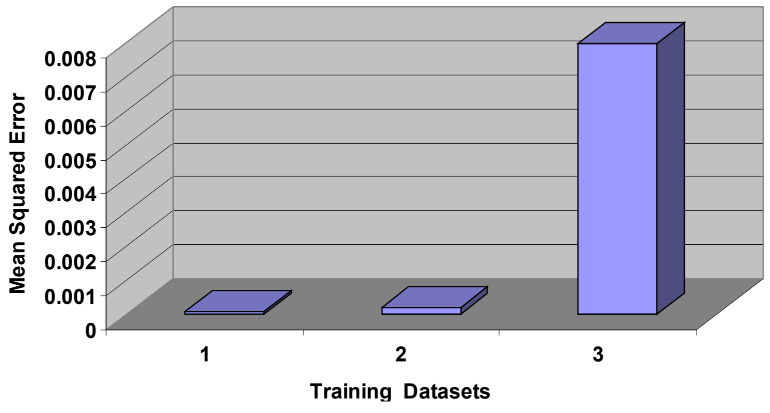

Figure 4. Mean squared error as a function of training datasets.

It is proved in this work that a neural network algorithm converges to the dataset with the most dominant effect. Such feature might be considered a good one, but on the contrary, if one or more dataset dominated the neural network behavior, then its pattern recognition, classification, and prediction would become deviated and unbalanced.

These findings in the research can be utilized to uncover foreign and hostile datasets and analyze their effects. Also, it can be a good approach to modify preprocessing techniques that will produce such dominant patterns.

References

[1] Belghinia, N., Zarghilib, A., Kharroubic, J. and Majdad, A. (2012) Learning a Backpropagation Neural Network with Error Function Based on Bhattacharyya Distance for Face Recognition. *International Journal of Image*, *Graphics and Signal Processing*, **4**, 8-14. http://dx.doi.org/10.5815/ijigsp.2012.08.02

[2] Wilamowski, B.M. and Yu, H. (2010) Neural Network Learning without Backpropagation. *IEEE Transactions on Neural Networks*, **21**, 1793-1803. http://dx.doi.org/10.1109/TNN.2010.2073482

[3] Subrahmanya, N. and Yung, C.S. (2010) Constructive Training of Recurrent Neural Networks Using Hybrid Optimization. *Neurocomputing*, **73**, 2624-2631. http://dx.doi.org/10.1016/j.neucom.2010.05.012

[4] Puma-Villanuevaa, W.J., Santos, E.P. and Zuben, F.J.A. (2012) Constructive Algorithm to Synthesize Connected Feedforward Neural Networks. *Neurocomputing*, **75**, 14-32. http://dx.doi.org/10.1016/j.neucom.2011.05.025

[5] May, P., Zhou, E. and Lee, C.W. (2014) Improved Generalization in Recurrent Neural Networks Using the Tangent Plane Algorithm. *International Journal of Advanced Computer Science and Applications*, **5**, 118-126. http://dx.doi.org/10.14569/IJACSA.2014.050317

[6] Ahmed, S.U., Shahjahan, M.D. and Kazuyuki, M. (2011) A Limpel-Ziv Complexity Based Neural Network Pruning Algorithm. *International Journal of Neural Systems*, **21**, 427-441. http://dx.doi.org/10.1142/S0129065711002936

[7] Bavafaye Haghighia, E., Palm, G., rahmati, M. and Yazdanpanahc, M.J. (2015) A New Class of Multi-Stable Neural Networks: Stability Analysis and Learning Process. *Neural Networks*, **65**, 53-64. http://dx.doi.org/10.1016/j.neunet.2015.01.010

[8] Nie, X. and Zheng, W.X. (2015) Multistability of Neural Networks with Discontinuous Non-Monotonic Piecewise Linear Activation Functions and Time-Varying Delays. *Neural Networks*, **65**, 65-70. http://dx.doi.org/10.1016/j.neunet.2015.01.007

[9] Wright Shrestha, S.B. and song, Q. (2015) Adaptive Learning Rate of SpikeProp Based on Weight Convergence Analysis. *Neural Networks*, **63**, 185-198. http://dx.doi.org/10.1016/j.neunet.2014.12.001

[10] Li, X., Gong, X., Peng, X. and Peng, S. (2014) SSiCP: A New SVM Based Recursive Feature Elimination Algorithm for Multiclass Cancer Classification. *International Journal of Multimedia and Ubiquitous Engineering*, **9**, 347-360. http://dx.doi.org/10.14257/ijmue.2014.9.6.33

[11] Qingsong, Z., Jiaqiang, E., Jinke, G., Lijun, L., Tao, C., Shuhui, W. and Guohai, J. (2014) Functional Link Neural Network Prediction on Composite Regeneration Time of Diesel Particulate Filter for Vehicle Based on Fuzzy Adaptive Variable Weight Algorithm. *Journal of Information & Computational Science*, **11**, 1741-1751. http://dx.doi.org/10.12733/jics20103209

[12] May, P., Zhou, E. and Lee, C.W. (2013) A Comprehensive Evaluation of Weight Growth and Weight Elimination Methods Using the Tangent Plane Algorithm. *International Journal of Advanced Computer Science and Applications*, **4**, 149-156. http://dx.doi.org/10.14569/IJACSA.2013.040621

[13] Ennett, C.M. and Friz, M. (2003) Weight-elimination Neural Networks Applied to Coronary Surgery Mortality Prediction. Information Technology in Biomedicine. *IEEE Transactions on Information Technology in Biomedicine*, **7**, 86-92. http://dx.doi.org/10.1109/TITB.2003.811881

[14] Lalis, J.T., Gerardo, B.D. and Byun, Y. (2014) An Adaptive Stopping Criterion for Backpropagation Learning in Feedforward Neural Network. *International Journal of Multimedia and Ubiquitous Engineering*, **9**, 149-156. http://dx.doi.org/10.14257/ijmue.2014.9.8.13

[15] Iskandarani, M.Z. (2014) A Novel Approach to System Security Using Derived Odor Keys with Weight Elimination Neural Algorithm (DOK-WENA). *Transactions on Machine Learning and Artificial Intelligence*, **2**, 20-31. http://dx.doi.org/10.14738/tmlai.22.138