Application of the neural networks in events classification in the measurement of spin structure of the deuteron

R Sulej[1], **K Zaremba**[1], **K Kurek**[2] **and E Rondio**[2]

[1] Institute of Radioelectronics, Warsaw University of Technology, Warsaw, Poland

[2] Soltan Institute for Nuclear Studies, Warsaw, Poland

Abstract. In this paper, we present the application of a neural network for events classification in high energy physics experiment. As a network model we use multi-layer perceptron (MLP) with a dynamic topology adjustment algorithm. Our solution covers both adding new hidden neuron units and removing unnecessary units. Neural network results are compared to the standard kinematical cuts technique and to the well known *k*-nearest neighbor (kNN) classifier.

1. Introduction.

COMPASS (**CO**mmon **M**uon **P**roton **A**pparatus for **S**tructure and **S**pectroscopy) is a high-energy physics experiment at the Super Proton Synchrotron (SPS) at CERN in Geneva, Switzerland [1]. The purpose of this experiment is the study of hadron structure and hadron spectroscopy with high intensity muon and hadron beams. This experiment is a continuation of the program that was initiated by the observation that only a small fraction of the proton spin originates from the spin of the quarks (EMC experiment [2]). One of the hypothesis to be tested, is that a significant fraction of the nucleon spin originates from polarized gluons. Analysis of the process that involves gluons, so-called photon-gluon fusion (PGF), is required to measure the gluon polarization. This process occurs in a deep inelastic lepton-nucleon scattering among the other processes (that we consider as a background) when muon beam hits the nucleons of the target. Diagrams of the PGF and the two lowest order processes (most probable) are presented in figure 1. As we cannot observe quarks and gluons directly, it is not easy to distinguish which process was the origin of final particles (*X*) that appear in the detectors after scattering. Two approaches are used to select PGF from the background. The first one is looking for pairs of hadrons with high transverse momentum (p_t) which are likely to appear as a product of $\mathrm{q}-\overline{\mathrm{q}}$ pairs produced in PGF. This suppresses significantly the dominant contribution from the virtual photon absorption process (LO), where the only source of p_t is relatively small momentum of quarks inside the nucleon [3]. The other approach is to look for charm particles (containing c quark) which may be produced in PGF and not in LO and Compton processes, as they are interactions with quarks existing in nucleon (practically only u and d quarks). Unfortunately, heavy c quarks are also rarely produced.

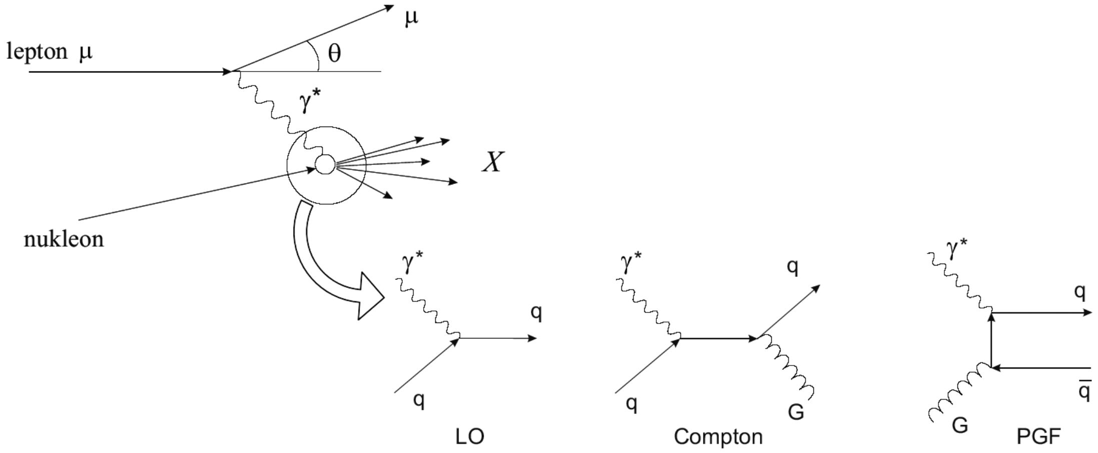

Figure 1. Scattering on a nucleon and diagrams of PGF and two lowest order processes; symbols in the diagrams: G – gluon, q – quark, γ^* – virtual photon, μ – scattered muon, X – hadrons in the final state.

In this work, we present the method of selection of PGF events from high-p_t sample using MLP network model (with sigmoid activation function) with automated control of the network structure. Fixed-structure MLP networks have been used in HEP pattern recognition problems for decades [4, 5] and have been shown to be a powerful tool in this field. However, many parameters like training algorithm parameters or number of the hidden units have to be tuned manually with no prior indication about the correct values. Too small and too large network structures lead to decreased performance of the network due to the lack of capacity and poor generalization capabilities respectively. Our approach eliminates the need for manual trial-and-error adjustment of the network size. Moreover, the structure modifications change the parameter space which is more favorable for back-propagation optimization algorithms, giving a chance of escaping from possible local minimums. The presented technique is general and may be used in the MLP network preparation for any task.

The application of a neural network event classification is tested here on the sample of Monte Carlo simulated events obtained with the LEPTO generator [6]. Obtained results are compared to *k*-nearest neighbors classifier and to the standard technique used for PGF events selection – manually optimized cuts applied independently to measured variables (despite obvious limitations, this technique is still popular due to the possible relationship between chosen cuts and the knowledge about the interaction model).

2. Algorithm.

There is no *a priori* knowledge for PGF selection (as for most of the real life tasks that neural networks are faced with) giving indications for the size of the network. The aim of the algorithm is to establish the size of each hidden layer by adding and removing neurons during the single run of the network training.

Between the structure modifications, the network is trained with a back-propagation algorithm (we use quick-prop [7], but other algorithms are also applicable).

2.1. Insertion of new hidden units.

The idea of the network growth is similar to the one used in Cascade-Correlation network [8]. Like in this model, a pool of neuron candidates is used. Candidate input weights are initialized as a small random values; range for the initialization of the candidate output weights is chosen as comparable to the output weights range of the other neurons in the layer. Candidates are trained while the rest of the network weights are frozen. When training is completed, the best candidate (resulting with the smallest network error) is chosen to extend the network structure. It is done only if this neuron decreases the network error, otherwise the network remains unchanged. New neurons in Cascade-Correlation model become a part of the fixed structure. In contrary, in our algorithm all network weights are retrained after successful insertion of new neuron. New neurons are not fixed on particular features of the training data; they are just pre-trained to fit into the existing structure in the best possible way. This allows a better integration of the new neuron with the network structure and it results in a smoother network response than in case of Cascade-Correlation network.

2.2. Pruning redundant neurons.

Training algorithms usually start from a random neuron weights. We either use the fixed structure or decide to add new neurons during the training, there is a chance that not fully utilized hidden units will appear. The presented techniques also may recognize neurons responsible for overtraining effect. We distinguish three types of redundant neurons that can be safely removed. They will be denoted as "twin", "constant" and "dead" neurons. After pruning network is retrained; it usually takes only few iterations to eliminate influence of structure reduction, when the error change per iteration returns to the level similar as before the modification (figure 2).

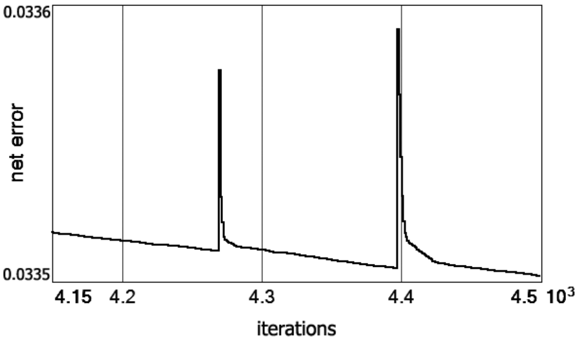

Figure 2. Network error as a function of the iterations – late training stage section (small error decrease) with short spikes after pruning of the two neurons is shown (vertical scale stretched strongly).

2.2.1. „Twin" neurons. Neurons which respond to any excitation with nearly the same output value are likely to be joined into a new single unit. This observation has been used in [9] for joining neurons in a radial basis function (RBF) network, where the difference between activation values in the whole activation function domain is used for the detection of the neuron pairs. This requires activation function to be *local* (like gaussian function). In case of sigmoid activation functions used in MLP network, the detection of the *twin* neuron pairs must take into account the distribution of the neuron input vectors, which must limit the effective range of the activation function domain. This is always true for layers preceded with hidden layer which responds with the limited activation values; also for the first hidden layer this condition is met in most real-life cases. The measure of similarity of the A and B neurons can be calculated using training data:

$$t_{AB} = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(o_i^A - o_i^B\right)^2}{\sigma_A^2 + \sigma_B^2}, \tag{2.1}$$

where o_i is the neuron output for i-th training pattern and σ^2 is the neuron output variance for the training patterns. As we observed, neurons with $t < 0.05$ may be joined with no significant increase of the network error. New neuron connection weights to the subsequent layer should be calculated as sums of the corresponding weights of the removed pair. Vector of the input weights for the new neuron is calculated as $\frac{1}{2}(\mathbf{w}_1 + \mathbf{w}_2)$, where $\mathbf{w}_1$ and $\mathbf{w}_2$ are the input weights vectors of the joined pair.

2.2.2. „Constant" neurons. Some neurons learn irrelevant statistical fluctuations, or produce nearly constant output for all training vectors. These neurons are detected by a small standard deviation σ_o of their activation for training vectors, when compared to the full possible range r of activation values:

$$c = \frac{\sigma_o}{r}. \tag{2.2}$$

Neurons with $c < 0.05$ usually may be removed. When *constant* neuron is removed, its mean output value should be included in the bias of neurons in the following layer. Neurons of this type correspond to neurons that appear in the network trained with the error function extended by a function of squared activation of the hidden units (neuron *energy* term) [10], but our algorithm do not modify the error function and the network is not forced to produce such a neurons.

2.2.3. „Dead" neurons. The simplest, but also a frequent scenario, is that all weights connecting neuron with units in the subsequent layer are decreased by the training algorithm to insignificant level. If the norm of neuron output weights vector $|\mathbf{w}|$ is much below the mean value for all neurons in the layer $\mu_{|\mathbf{w}|}$, the tested neuron can simply be removed without the influence on functioning of the whole network. For measure expressed as:

$$d = \frac{|\mathbf{w}|}{\mu_{|\mathbf{w}|}}, \tag{2.3}$$

safe value to remove dead neuron is $d < 0.05$.

Training stops when the network structure becomes stable or the further growth does not change the error significantly. Pruning techniques may be combined with weight elimination algorithms like optimal brain surgeon [11].

Presented structure modifications do not waste efforts of previously done training. Training of the small initial structure takes significant part of the process, then gradually growing network is trained in a relatively short time. Training process is more repeatable than training the big, fixed-sized network. All of this causes that although the number of training iterations is bigger than in case of the fixed structure network, time spent on the network preparation is much shorter for proposed algorithm.

3. Results.

Classification tool for PGF selection was prepared using the Monte-Carlo simulated data, then it was tested on another set of simulated events before applying to real data – it is either the neural network, cuts or any other technique. To measure the quality of classification we use the following variables:

$$purity = 100\% \cdot N_{\mathrm{sig}}(OutputSet) \,/\, (\, N_{\mathrm{bkg}}(OutputSet) + N_{\mathrm{sig}}(OutputSet)\,), \tag{3.1}$$

$$efficiency = 100\% \cdot N_{\mathrm{sig}}(OutputSet) \,/\, N_{\mathrm{sig}}(InputSet), \tag{3.2}$$

where $N_{\mathrm{sig}}(Set)$ and $N_{\mathrm{bkg}}(Set)$ are the numbers of the signal and background events in the *Set*; *InputSet* is the set of events which we classify and *OutputSet* is the set of events with the classifier answer above a given threshold. The full information about the signal-background separation capabilities is contained on a plot, where *purity-efficiency* curve is constructed from points obtained for different threshold values.

There are 11 kinematical variables available as an input feature vectors. Simple normalization to 0-mean and unitary standard deviation was applied before classification. As the background and PGF events have almost the same probability distributions, it is not simple to increase the fraction of PGF in the selected sample. Concurrently, even a small gain in the *purity-efficiency* (equations (3.1), (3.2)) of the PGF selection can significantly improve the accuracy of further data analysis. Figure 3 presents the classification results obtained for cuts technique (performed on a combination of the two most informative variables; handling longer feature vectors is difficult with this technique), neural network prepared with dynamic size adjustment algorithm and *k*-nearest neighbor classifier. The kNN classifier answer is calculated as a ratio of PGF events among the *k* training vectors nearest to the classified event feature vector (Euclidean measure of distance is used). Error bars marked on the curves in figure 3 indicate the possible statistical fluctuation of the results related to the chosen testing sample. Fluctuation of the method itself may be approximated only by repeated training; as it is shown in figure 4 this effect is at the level of 1σ of the statistical fluctuations. Particular purity-efficiency balance (threshold applied to the network output) appropriate for further data analysis is selected by minimization of statistical error of the gluon polarization evaluation (both the number of the selected PGF events and the ratio of PGF events in the selected sample affects the accuracy of the calculations).

The systematic errors are related to a Monte-Carlo description of the data and to the application of the presented NN-based method of classification. A good description of the real data by the Monte-Carlo generator (together with full simulation of the experiment's apparatus and reconstruction) is a crucial point in this type of analysis (also for cuts-based method). Especially all parameters used as an input for network training should be well reproduced by simulated ones. To estimate this part of the systematic error the set of Monte-Carlo generator parameters can be changed (e.g. sets of parton distribution functions, internal generator regulators, scales etc., see [3]) and the evaluation of the gluon polarization is repeated to estimate the possible differences. The systematic error introduced by the classification method can be estimated by changing the purity-efficiency balance (network output threshold) around optimized value.

Comparison shown in figure 3 indicates that there is a significant amount of information contained in all available variables (and in their mutual correlations) which cannot be fully explored with a simple manual technique. Also, the generalization capabilities of the neural network allow it to classify better in a multi-dimensional feature space than it is possible with well-known kNN technique.

The distribution of the network output (figure 5) confirms the expectations mentioned in section 1, that PGF process is relatively easy to be separated from LO events, while Compton events are much more difficult to be classified correctly due to the similarity of process products to the products of PGF (quark and emitted gluon in Compton process are the sources of high-p_t hadrons like $\mathrm{q}-\overline{\mathrm{q}}$ pair in PGF process, see figure 1).

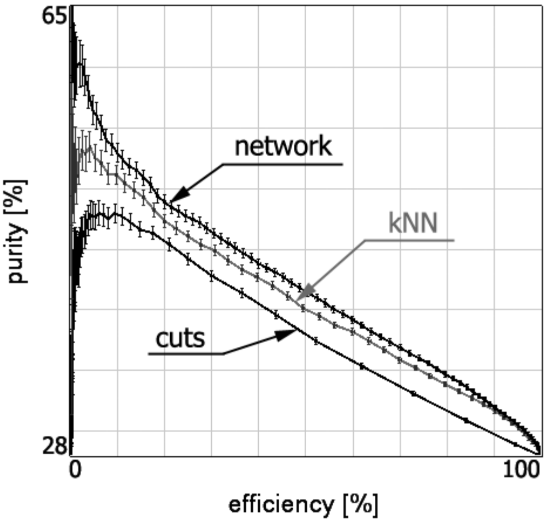

Figure 3. Selection results for different classifiers.

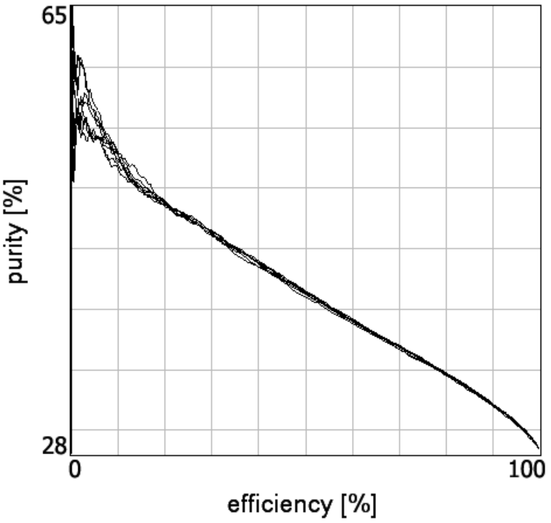

Figure 4. Selection results for 8 attempts of training the network using the same set of events.

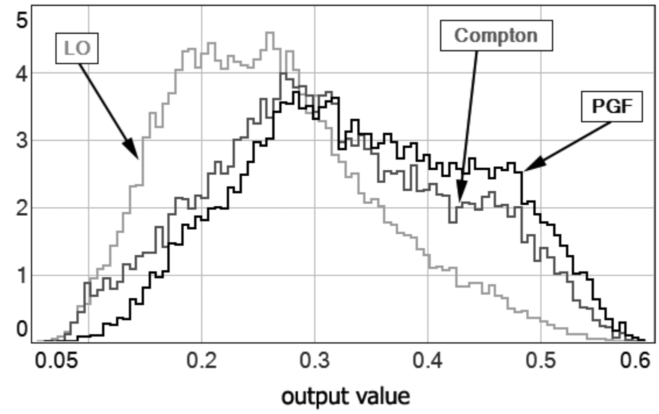

Figure 5. Distributions of the network output values for LO, Compton and PGF processes.

4. Conclusions.

The presented algorithm establishes the MLP network structure adequate to the given problem. Network grows in a controlled way – new neurons are accepted only if they contribute to the error decrease. Redundant neurons are detected and removed without the noticeable influence on the whole network functioning. This process also helps to keep the generalization capabilities of the network and to escape from possible local minimums. Training stops when the network structure becomes stable, or the further growth does not change the network error. Algorithm has been examined on the PGF events selection, resulting with significant improvement of classification, and consequently the higher precision of the gluon polarization analysis – statistical error may be significantly decreased when selection with neural network is compared to the cuts method. Presented techniques are not limited to classification tasks and may be used in the preparation of the MLP neural network for any application.

Acknowledgements.

This work was supported by KBN:
134/E-365/SPB/CERN/P-03/DWM 62/2004-2006
621/E-78/SPB/CERN/P-03/DWM 576/2003-2006

References.

1. Ageev E S *et al.* (the COMPASS collaboration) 1996 Common Muon and Proton Apparatus for Structure and Spectroscopy *CERN/SPSLC* **(SPSC/P297)** 96-14
2. Ashman J *et al.* (the EMC collaboration) 1989 An investigation of the spin structure of the proton in deep inelastic scattering of polarized muons on polarized protons *Nucl. Phys.* B **328 1** 1-35
3. Adeva B *et al.* (the SMC collaboration) 2004 Spin asymmetries for events with high p_T hadrons in DIS and an evaluation of the gluon polarization *Physical Review* D **70** 012002
4. Denby B 1988 Neural networks and cellular automata in experimental high energy physics *Computer Physics Communications* **49** 429-448
5. Denby B, Shpakov D and Wyss L J 1995 Applications of neural networks in high energy physics *Proc. of the International Conference on Artificial Neural Networks ICANN'95* Paris, France **1** 615-622
6. Ingelman G, Edin A and Rathsman J 1997 Lepto 6.5 - A Monte Carlo generator for deep inelastic lepton-nucleon scattering *Comput. Phys. Commun.* **101** 108-134.
7. Fahlman S 1988 An empirical study of learning speed in back-propagation networks *CMU-CS-88-162 report* Carnegie Mellon University
8. Fahlman S and Lebiere C 1991 The cascade-correlation learning architecture *CMU-CS-90-100 report* Carnegie Mellon University
9. Jankowski N and Kadirkamanthan V 1997 Statistical control of growing and pruning in RBF-like neural networks *Proc 3rd Conf. Neural Networks and Their Applications* Kule, Poland 663-670
10. Chauvin Y 1989 A back-propagation algorithm with optimal use of hidden units *Advances in Neural Information Processing Systems 1* CA Morgan Kaufmann, San Mateo, 519-526
11. Hassibi B and Stork D 1993 Second order derivatives for network pruning: Optimal brain surgeon *Advances in Neural Information Processing Systems 5* CA: Morgan Kaufmann, San Mateo 164-171