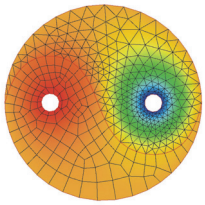
Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE

Informatyka w Technologii Materiałów

Vol. 8, 2008, No. 2



ARTIFICIAL NEURAL NETWORK TECHNIQUES IN COLD ROLL-FORMING PROCESS DESIGN

ANTHONY DOWNES, PETER HARTLEY

School of Mechanical Engineering, University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK
Corresponding Author: p.hartley@bham.ac.uk

Abstract

There are several types of Artificial Neural Networks (ANN) each having different capabilities and characteristics that permit a wide range of applications. The purpose of this paper is to review three applications of ANN systems in the context of cold roll-forming process design. A brief description of each system explains the significance of the network architecture and training technique, and emphasizes the importance of selecting the most suitable system for the problem being processed.

Key words: artificial neural network, cold roll-forming

1. INTRODUCTION

1.1. Artificial Neural Networks (ANN)

An interpretation of a biological neuron, (Fausett, 1994) commonly called a brain cell, is shown in figure 1(a). Signals are transmitted between neurons by the movement of ions called "neurotransmitters" which flow into the "synaptic gap", where some will become attached if conditions are favourable. The "soma" receives neurotransmitters from the "dendrites", and if excitation exceeds a threshold value, they move into the "axon" where the dendrites of other neurons have the opportunity to receive them through their synaptic gaps. McCullock and Pitts, (Anderson et al, 1998) described the rudiments of the biological neuron in 1943, from which the conceptual diagram shown in figure 1(b) is derived. The input signals are multiplied by a value, called a "weight", which can be a positive or negative value. An output signal is only transmitted if the summation of all the input signals exceeds the threshold value of +1. Artificial neural networks originally consisted of McCullock and Pitts neurons connected together and several training techniques were developed. The value of the network weights is adjusted during the learning process, which is a representation of the changes in the synapse transmission efficiency of the neurotransmitters in a biological neuron.

The embryonic phase of ANN systems development was based on modelling biological neurons but the research has evolved towards the development of software that provides a better solution to problems compared to alternative options. There are many different ANN systems but mostly they are designed to solve a pattern recognition problem. An input vector is processed, and depending on some mapping relationship, an output vector is produced that classifies the input data. There are several commercial packages available which provide a comprehensive range of different ANN systems. If MatLab™ (MatLab Manuals 2000) is used, however, the powerful programming language can be utilized to process the input data and present the results using the graphics facilities. Advantage is taken of this facility in the examples presented in this paper.

ISSN 1641-8581

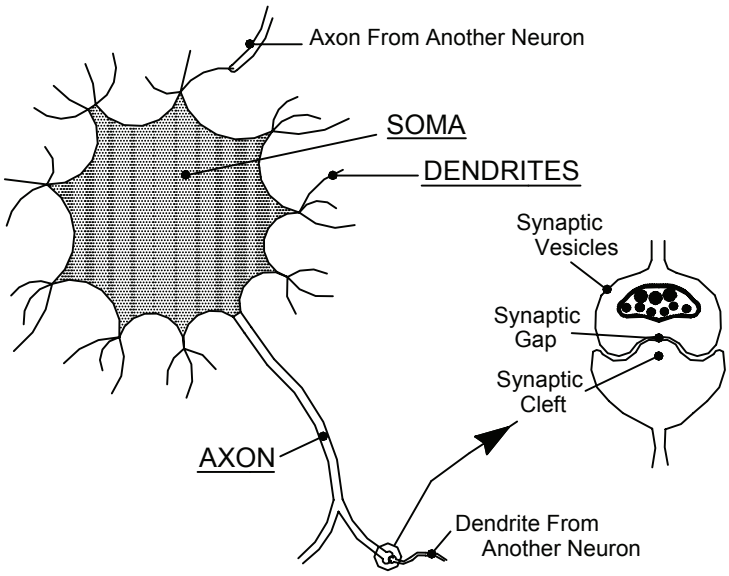


Fig. 1(a). *Biological Neuron, (Fausett, 1994)*

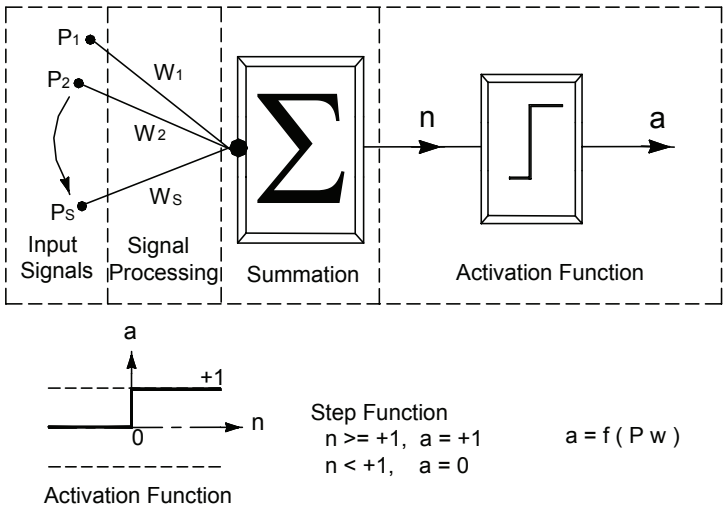


Fig. 1(b). *Artificial Neuron, (Anderson et al, 1998)*

1.2. Cold Roll-Forming

Cold roll-forming is based on the simple process of guiding sheet metal at room temperature through sets of rotating profiled rolls. Designing the roll-forming tools to produce the specified section geometry is not straightforward however, particularly the profiles machined on the rolls. The roll profiles do not normally match the geometry of the section being produced, but rely on localised contact to generate the bending required at each forming stage, and there are often several forming stages for any section. The final section is therefore a result of the accumulation of several bending processes. Undesirable strains are generated in the sheet during the forming process and their magnitude may result in section defects, such as lengthwise curvature or twist. The current tool design process is subjective and there are few design rules that are universally acknowledged. Empirical knowledge garnered from many years of tool design experience is usually the basis of every tool design. The wide use of devices called "bending fixtures" that are positioned at the exit of the forming machine, to remove the geometric defects, highlights the difficulty of forming the section to the desired quality standards solely by using profiled roles.

Some recent examples of the application of artificial neural networks related to the design of metal forming processes are Kim et al (2007) who used a knowledge-based ANN to determine the heating strategy for a warm forging process, and Ozerdem and Kolukisa (2008) who used ANN techniques to determine material properties in hot rolling. The problem of geometric defects has been addressed by John et al (2008) using a combination of neural network and genetic algorithm techniques to predict the flatness of hot rolled strip, and by Peng et al (2007) who developed a neural network-based control system to improve strip shape in cold rolling. Cold roll forming however does not appear to be a subject in which ANN techniques have recently been applied, which is surprising as this type of forming process, with such a variety of process uncertainties, is ideal to benefit from the ANN approach. The following examples demonstrate how different ANN approaches may be used to address various problems associated with cold roll-forming processes.

2. FEED-FORWARD NETWORKS USING BACK-PROPAGATION TRAINING

The industrial collaborator for this project (Bradbury, 2008), had stored the details of previous roll-forming tool design work in electronic format on their computer system. However, there was no rigorous classification of the design data and this was not helpful for the tool designers and cost estimators who occasionally search previous tool designs to find similar sections to the current design task. The design team decided that a classification of the design data should refer to the three section features listed below.

1) The total number of bends,
2) The overall width of the sheet in mm prior to forming,
3) The sheet thickness in mm.

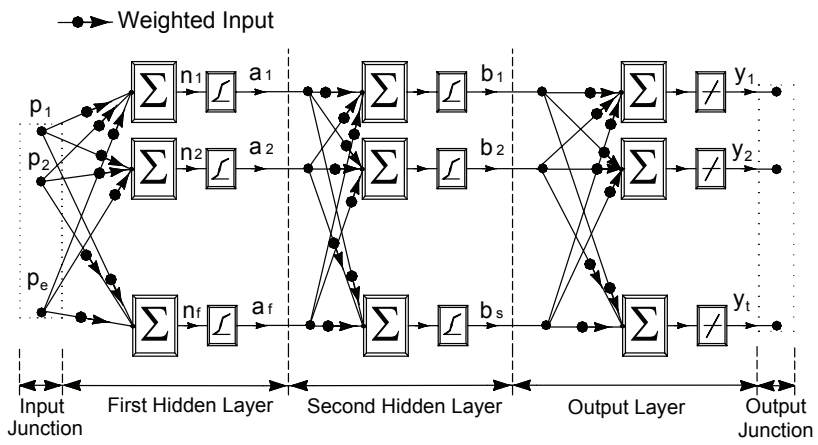


Fig. 2. *Feed-Forward Network Architecture*

The built-in programming language in AutoCAD (AutoCAD Manuals, 1996) which is a bespoke variant of LISP called AutoLISP^{TM}, was used to evaluate these three section features directly from AutoCAD drawings of the section geometry. A total of 54 storage locations are used in the classification system. This involved 6 computer hard disc folders named "BIN1" to "BIN6", each having 9 sub-folders named "box1" to "box9". An ANN system was required to perform an accurate mapping operation that processed a 3 element input vector, representing the section features, and produced a 54 element output vector, representing the storage locations. The feed-forward network architecture was chosen and trained using the back-propagation algorithm (FFBP network). This ANN system is frequently used and according to Wasserman (1993) over 85 percent of ANN system applications use the FFBP network. The element in the input vector that represents the sheet width must cover a range of values between 50 mm to 1500 mm while the number of bends is usually in the range of 1 to 44. Although the sheet thickness may be as large as 6 mm, the roll-formed products produced by the industrial collaborator generally had a thickness in the range of 0.5 mm to 3.5 mm. The output vector had 54 binary elements, and if the mapping is exact, one element would be unity to identify the storage location and all other elements had a value of zero.

The FFBP network architecture is shown in figure 2. A "Step" activation function is used in the McCullock and Pitts model of the biological neuron, which has an output signal of +1. If the activation function is linear the output signal is a variable that is proportional to the value of the summation of the input signals. To allow the powerful training algorithms to be applied however, and overcome the limitation on the type of problem that can be processed, a smooth non-linear function is required that is differentiable in all regions of the range. The most frequently used non-linear activation function is the sigmoid function. If the classification task involves an output vector with elements ranging from zero to +1 the log-sigmoid is the preferred option. From empirical studies, however, the hyperbolic-sigmoid function frequently results in faster training if the back-propagation algorithm is applied (Bishop, 1995). During the trials carried out in this project the use of log-sigmoid artificial neurons in both the hidden layer and the output layer resulted in a significant improvement in performance.

The number of hidden layers that is required is dependent on the complexity of the mapping operation. Two hidden layers can carry out more complicated division of the feature space compared to one hidden layer. The Kolmogorov theorem (Kurkova, 1992) states that any continuous non-linear mapping can be performed by a network with no more than two hidden layers. Experimentation to find the best network architecture, however, should acknowledge that there are examples of successful FFBP networks with more than two hidden layers. A single layer network is limited to the processing of linearly separable problems, but for relatively simple pattern matching tasks it is capable of producing a satisfactory solution (Bishop, 1995).

The problem of finding the number of artificial neurons in each layer of the network that produces the best performance validation is usually the most time-consuming phase of the FFBP network design. Generally, more artificial neurons are required for complicated mapping operations but too many will result in a poor network performance. If the network has a sufficient number of network weights to accurately map all the input vectors in the training data, including any noise they contain, it is likely that the generalization properties will be poor. When input vectors not used in the training data are processed the mapping will frequently be inaccurate. There is no widely accepted theory that provides a good approximation for the best number of artificial neurons in a FFBP network and usually a trial and error procedure must be used. Starting the trials with a small number of artificial neurons and progressively increasing their number, (network growing), was the

preferred approach in this project. The alternative is "network pruning", which begins with a large network and gradually reduces its size. The network architecture which finally produced satisfactory results in this project consisted of one hidden layer of 19 log-sigmoid artificial neurons and 54 log-sigmoid artificial neurons in the output layer

The back-propagation algorithm was used to determine the value of the network weights. It applies a supervised learning technique that uses training data consisting of vectors selected from the input space paired with "target" vectors that represent the corresponding desired output in the output space. At the start of training the network weights are set to small random values, typically between ±5. Small network weights will help to optimize the generalization properties of the network (Swingler, 1996). The input vectors are presented sequentially to the network and the network output vector error is calculated by referring to the target vector. In this project the "sum-of-squares" error function is used and the network weights are adjusted in a manner that results in an efficient error minimization. The back-propagation algorithm searches the error surface to find the location where the error is a minimum value. Apart from the simple network architecture involving a single layer, the error surface will be a highly non-linear function of the network weights (Bishop, 1995). Consequently, the extremely irregular surface can cause the minimization process to become trapped in a local minimum. Training a FFBP network can be troublesome but fortunately it is not always necessary to find the global minimum. In most cases the error does not have to be reduced to a very small value because the generalization properties are affected due to over-fitting the training data. A technique used in statistics called "cross-validation" is often used to determine the training time that will optimize the generalization properties. There are several training algorithms suitable for FFBP networks, including the conjugate gradient and quasi-Newton methods. Selecting the best algorithm will depend largely on the problem being solved, the size of the network and the activation functions of the artificial neurons (Bishop, 1995). The Resilient Back-propagation algorithm (Riedmiller et al, 1993) was found to perform better than the other options used in this project.

It is essential that the input data used in the training phase has sufficient information to allow the formulation of the mapping relationships. A large number of elements in the input vectors will require a large number of network weights and this will also increase the training time. It is generally helpful if the input vectors have a relatively small number of elements, therefore having only three elements in the input vectors in this project will significantly increase the likelihood of successfully training the network. The numerical values of the elements in the input vectors can also create difficulties and adjustments are often made. They should all be of a similar order of magnitude and not substantially larger than the input range of the artificial neurons in the first layer of the network. The simplest pre-processing is a linear re-scaling. For this project the numerical values of the input vector elements were obtained by multiplying the sheet thickness by 10, dividing the sheet width by 100, the number of bends was not adjusted.

The generation of the training data is straightforward and this was an important reason why a FFBP network was chosen. When the performance of the network is poor a standard remedy is to change the training data and re-train the network. For example, for any input vectors that represent a section with a sheet width equal to, or less than 200 mm, a sheet thickness of 1 mm or less, and 4 or less bends, the target vector will be the same and will represent the storage location of BIN1-box1. Therefore if the mapping to this storage location is poor it is a simple task to add extra input vectors to the training data from the corresponding input space to improve its representation.

The design of a FFBP network can involve a large number of trials and may become tedious and time-consuming. The network architecture must be optimized and the most suitable training algorithm and training data must be found. During the training phase in this project the training data was progressively increased to 3330 input and target vector pairs. The fully trained FFBP network was tested using selected input vectors that were close to the boundaries between the different classes. A test that involved the mapping of 108 input vectors resulted in only two placed in an incorrect category, and both of the faulty mapping operations were directly adjacent to the correct class. When randomly selected input vectors were used the trained FFBP network correctly classified over 98 percent of the input vectors. The storage and retrieval system for roll-formed sections developed in this project achieved all the key requirements that had been requested by the industrial collaborator (Downes et al, 2004).

3. RADIAL BASIS FUNCTIONS

Manufacturing sheet metal products using the cold roll-forming process has many important advantages. The likelihood of section quality problems is the foremost drawback however, and is particularly true when a new tool design is involved where a similar design task has not previously been carried out. It is not a complex geometric section with a large number of bend regions that is most likely to result in tool design problems. Frequently it is the roll-forming of a region of the section profile, which may involve only 3 or 4 bends, that causes section defects. In figure 3, a region of a roll-formed section is highlighted. The industrial collaborator for this project (Bradbury, 2008) suggested that a search of previous design projects to establish if a similar integral shape was present in a section was a good strategy. Although it is unlikely that a previous tool design will be copied exactly, because there are several additional issues that must be considered, there is a good possibility that useful information may be gleaned from a similar integral shape.

An integral shape, such as the one shown in figure 3, has several important section features. The sheet thickness and the length of the linear portions between the bend regions will all influence the tooling that is required. The design team recommended, however, that only the bend angles should be used to assess the similarity between integral shapes. An AutoLISP program was utilized to evaluate the bend angles of selected integral shapes directly from AutoCAD drawings of the section. Consequently, a procedure was required that referred to the bend angles of selected shapes and performed a search to find if a similar integral shape was present in previously roll-formed sections. The approach used in this project allows the tool designer to select integral shapes from the section geometry of previous tool design projects and an ANN system is trained to recall the storage location of the section. Each input vector will consist of elements that represent the numerical values of the bend angles in the integral shape. The maximum bend angle is 180 degrees and this angle was divided into 15 degree intervals to obtain 12 categories. Therefore if the bend angle is 15 degrees or less it is represented in the input vector by an element equal to one, and if the bend angle is 166 degrees or greater the element is equal to 12. A pre-processing operation was also carried out on each input vector to obtain a mean of zero and a standard deviation of unity. This will reduce the training time and may improve the overall network performance, (MatLab Manuals, 2000). The industrial design team decided that a maximum of 8 bends in the integral shape was sufficient for most practical sections, therefore the input vectors have a total of 8 elements.

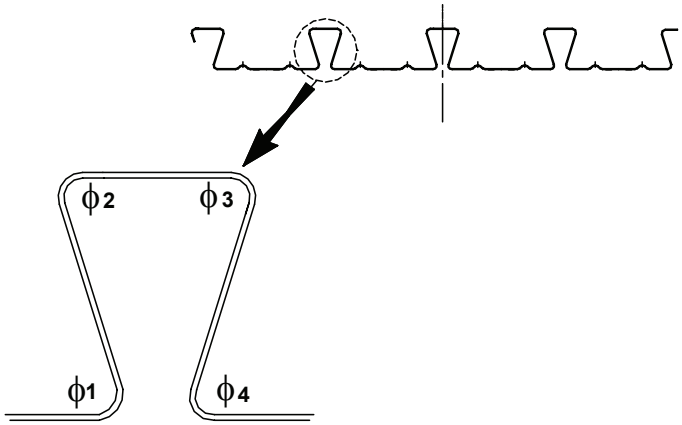


Fig. 3. *Example of an Integral Shape*

It is important that a relatively large number of integral shapes are represented in the training data. This will increase the likelihood of a search for a similar integral shape being successful. Therefore it is desirable for the industrial collaborator to continually increase the knowledge of the ANN system by adding new integral shapes and re-training the system using a relatively simple and rapid procedure. A FFBP network, which may become trapped in a local minimum during the training phase, will not be the ideal choice so other options were considered. A widely-used ANN system which has rapid training times is the Radial Basis Function (RBF) network (Powell, 1987). This has one hidden layer of artificial neurons with Gaussian activation functions and an output layer of artificial neurons with linear activation functions. The RBF network was used in this project and the processing of an 8 element input vector using the "Exact" training method (MatLab Manuals, 2000) is shown in figure 4. The network weight vector for the Gaussian artificial neuron is made equal to the training input vector that is being processed, as shown in figure 4(a). This produces a maximum output signal because the "DIST" function performs a Euclidean vector distance calculation between the two vectors, which in this case will be zero to produce a +1 output from the Gaussian distribution. Therefore if an input vector with slightly different element values is processed the Euclidean distance will not be zero and the output signal from this artificial neuron will be less

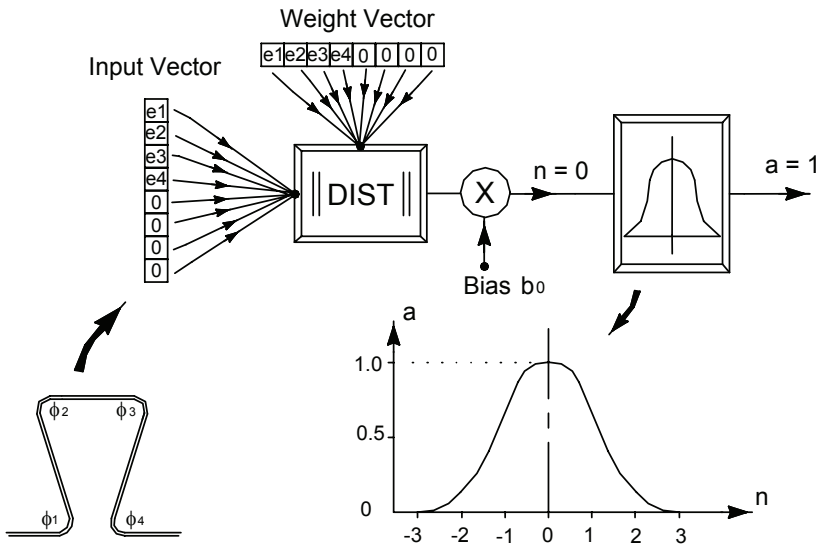


Fig. 4(a). *Artificial Neuron in the Hidden Layer of a RBF Network*

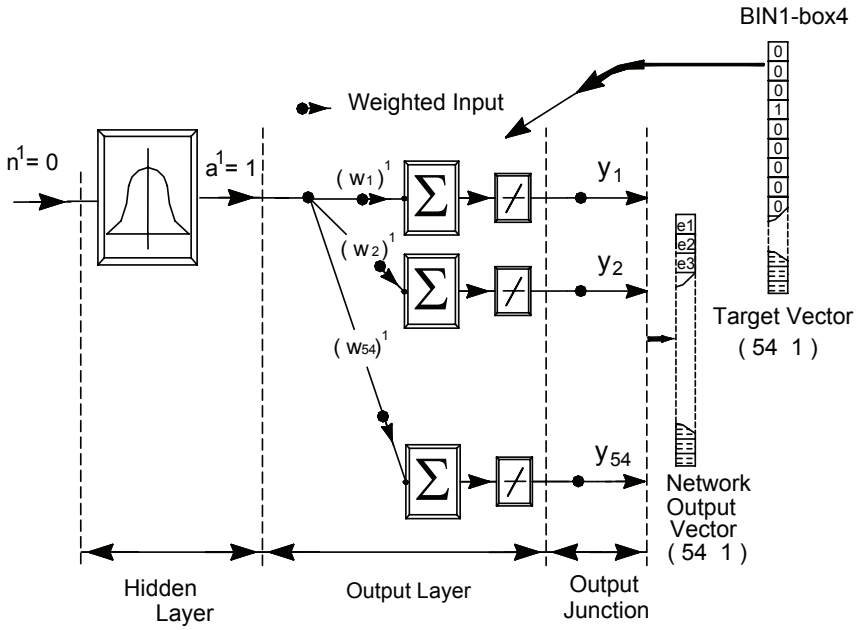


Fig. 4(b). *Output Layer of the RBF Network*

than unity. There are 54 storage locations and therefore there are 54 linear artificial neurons in the output layer of the network as shown in figure 4(b). The target vector in the training data is a 54 binary element vector with only one element set to unity to indicate the storage location of the integral shape that is being processed. When the MatLab "Exact" training method is used, one artificial neuron is created in the hidden layer for every input and target vector pair in the training data. The network weights in the output layer are adjusted during the training phase to produce a network output vector that is identical to the target vector. Unless the size of the training data is extremely large, the training phase for RBF networks is completed in negligible computation time. Therefore the industrial collaborator can expand the knowledge of the system by adding new integral shapes in a rapid and efficient manner. When an input vector not in the training data is processed by the trained RFB network a significant output signal from a hidden layer artificial neuron will only be generated if the input vector is similar to one of the input vectors used in the training data. How similar they must be to produce a significant output signal is dependent upon the Gaussian activation function. The MatLab "Exact" training method allows the user to increase the range of the Euclidean distances that produces significant output signals through an adjustment of the "Spread" parameter. This allows the sensitivity of the artificial neurons in the hidden layer to be set to a value that is appropriate to the application.

An input vector representing the integral shape shown in figure 5(a), was included in the training data to investigate the effect of the "Spread" parameter. When this input vector was processed by the trained RBF network, the Gaussian artificial neuron created by this integral shape during training produced the maximum output signal a^1 of +1. Two adjustments were carried out to the integral shape. The 90 degree angle is changed to 46 degrees and the 180 degree angle is changed to 151 degrees. The modified integral shape and the corresponding values of a^1 for different "Spread" settings are shown in figure 5(b). Additional examples are shown in figure 5(c) and figure 5(d). The trained network has clearly recognized the similarity between the three modified integral shapes and the integral shape shown in figure 5(a) that was used in the training data.

The output vector from the trained RBF network may have several element values in the region of +1 indicating that there is more than one storage location that may contain useful information. An advan-

tage of using MatLab to provide the ANN system is that the powerful MatLab language can be utilized to process the output vectors from the trained network and present the results in an orderly manner. This allows the tool designer to quickly select the most appropriate storage location to study. A test to estimate the training time was carried out by creating training data that consisted of 1536 input and target vector pairs. The relatively low-powered computer (100 MHz-P3 with 128 Mb RAM) completed the training phase in 64 seconds using the MatLab "Exact" method. All the tests carried out to assess the overall performance of the RBF network developed in this project produced satisfactory results (Downes et al, 2006).

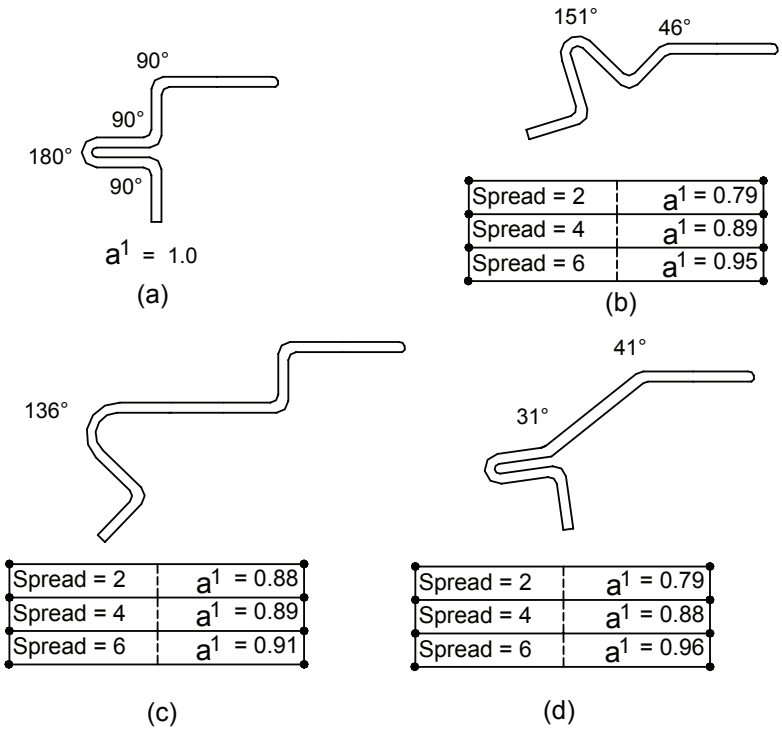


Fig. 5. *Comparison of Integral Shapes*

4. SELF-ORGANIZING ANN SYSTEMS

The cold roll-forming tool design industry in the UK is relatively small and competition for work is extremely intense. An important part of the formulation of a work tender for a prospective customer is the estimation of the roll-forming tool costs. The industrial collaborator for this project (Hadley, 2008) requested a less costly method for obtaining an approximation for the number of profiled rolls that will be required to produce the specified section geometry. This task was carried out by an experienced tool designer and the commissioning of their time significantly increased the overall cost of generating tenders. The estimation of tool costs is based on the number of forming stations that will be required to produce the section on a roll-forming machine. The objective of this project was to develop a computer-aided technique that processes the section geometry and predicts the number of forming stations that will be required. Therefore the cost of creating work tenders will be reduced because the time of a skilled tool designer will not be needed.

A forming station on the roll-forming machine will typically change the angle of a bend by 10 to 15 degrees. Therefore it is logical to conclude that the total number of bend regions and the summation of all the bend angles in the section geometry will influence the number of forming stations that will be required. The longitudinal strains that are generated at the edge of the sheet as it travels through the rolls are frequently the cause of section defects. Consequently, the displacement of the sheet edge as it moves through the machine is an important forming parameter. The three section features used to predict the number of forming stations are shown in figure 6. An AutoLISP program was used to evaluate the features directly from an AutoCAD drawing of the section geometry.

The standard of performance of ANN systems will depend largely on the training data that was used. For the FFBP network discussed in Section 2.0 the generation of input and target vectors to improve the network performance was a simple matter. Additional input vectors from regions of

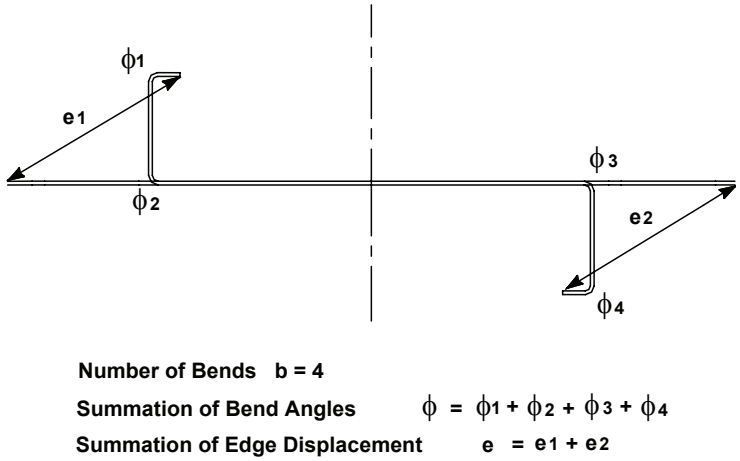


Fig. 6. *Three Section Features Used to Estimate the Number of Forming Stations*

the input space where representation was sparse, or from regions close to the classification boundary, could easily be obtained. When there is uncertainty on the categorization of input data the creation of the input and target vector pairs is not straightforward. The training of ANN systems using target vectors is called "supervised" learning while the alternative, "unsupervised" technique, does not require target vectors in the training data. An ANN system that is trained by unsupervised learning classifies the input data by detecting statistical trends in the training data. The industrial collaborator provided examples of section geometry and the corresponding number of forming stations used to manufacture them. This information however, was not suitable for training an ANN system. Additional forming stations are always required if the sheet is cut to the overall length prior to the roll-forming operation, compared to a continuous sheet feed from a large coil. A precision bend angle will probably require two additional forming operations and the sheet material will also influence the process. For example, stainless steel requires additional forming stations compared to low carbon steel. The objective of this project was to study the section geometry, and from this information alone, predict the minimum number of forming stations that will be required.

The most simple network architecture for unsupervised or "self-organizing" ANN systems consists of a single layer of artificial neurons (MatLab Manuals, 2000). Each artificial neuron has a weight vector equal in dimension to the input vectors. Prior to the commencement of training the network weights are set to small random values, therefore every weight vector in the layer is initially unique. The training involves the sequential presentation of each input vector in the training data. A computation is carried out to find the artificial neuron with a weight vector most similar to the input vector being processed. The negative of the distance between the two vectors is calculated. Consequently, if the two vectors are identical the vector distance is the maximum value of zero. The artificial neuron in the layer with the weight vector most similar to the input vector being processed is called the "winner". A weight adaptation is then applied to the weight vector of the winning neuron so the vector distance is reduced. This is called a "winner-take-all" method because only one weight vector is adjusted for each input vector presentation. The Self-Organizing Maps (SOM) are trained by adjusting the weight vector of the winning neuron and the weight vectors of the artificial neurons in the region of the winning neuron (Kohonen, 2001). This creates groups of artificial neurons with similar weight vectors which respond to a range of similar input vectors.

The output junction of the self-organizing network processes all of the output signals from the artificial neurons in the layer and computes the largest value. It then produces an output vector of binary elements, with a dimension equal to the number of artificial neurons in the layer. One of the elements will be set to +1, to indicate the winning neuron, and all other element values are zero. The classification of the input data, and the number of classes that are created, is determined by the computations carried out during the training phase. It is important, however, that a sufficient quantity of training data is available to allow the important statistical trends to be established. The network weight adaptation for self-organizing and SOM networks is a simple process compared to the back-propagation algorithm which involves the evaluation of derivatives. Unsupervised training is not associated with quick training times however, because a SOM network will normally take longer to train than a FFBP network.

The requirement in this project was to design an ANN system that processes an input vector which represents three features of the section geometry and produces an output vector that predicts the number of forming stations that will be required to manufacture the section by roll-forming. A Learning Vector Quantization (LVQ) network was chosen, which is only partly self-organizing (Kohonen, 2001). The LVQ network architecture is shown in figure 7. For every input vector in the training data a "winning" artificial neuron is assigned in the hidden layer using the winner-take-all method. The competitive activation function produces a binary element vector that indicates the winning neuron. Therefore the hidden layer classifies the input vectors into categories, termed "subclasses", in a similar manner to a self-organizing network. The LVQ network is trained using target vectors, however, unlike the self-organizing network. All the artificial neurons in the output layer have linear activation functions and their network weights are adjusted until the network output vector is similar to the corresponding target vector. This is carried out in a similar manner to the training of a FFBP network. The LVQ network first places the input vectors into subclasses in the hidden layer then assigns each subclass into the corresponding target class in the output layer. On completion of

the training there may be several subclasses assigned to the same target class.

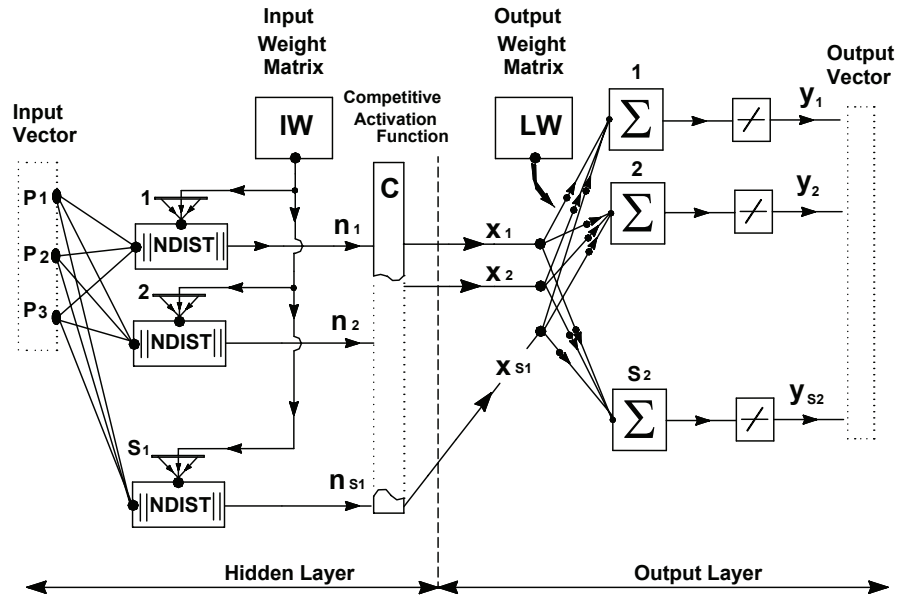


Fig. 7. LVQ Network Architecture

The industrial collaborator provided 40 Auto-CAD drawings of sections from previous roll-forming tool design projects. Four of the sections required only 6 forming stations and one section required 28 forming stations, therefore a range of 22 different forming machine sizes was represented. After evaluating the three section parameters for each section it was apparent that a simple re-scaling of the sheet edge displacement was required. The edge displacement values were divided by 10, resulting in all the element values in the input vectors representing the 40 sections falling in the range of 1 to 25. One element in the 22 element target vector was equal to unity to indicate the number of forming stations that was required and all remaining element values were equal to zero.

The LVQ network can be applied to any classification task and is not restricted to linearly separable problems. There must be a sufficient number of artificial neurons in the hidden layer, however, and the training data must have a suitable distribution of input vectors to ensure an adequate number of subclasses are created. Therefore a modification of the LVQ network architecture during the training phase may be necessary, but network design is unlikely to be as time-consuming as the FFBP network. If the performance of the trained ANN system is unsatisfactory the amendment of the training data is frequently a good strategy. This is not the case for a self-organizing network because the selection of additional input vectors that result in improving the network performance is usually a trial and error procedure. This was the reason why a LVQ network was used in this project in preference to a self-organizing or SOM network.

The performance of all ANN systems is dependent on the training data. If it is incomplete or there are inaccuracies the performance is likely to be poor. Generally, noisy data is tolerated but not if there is a significant proportion. In some cases the design of ANN systems will demand a resourceful approach to the generation of the training data. The industrial collaborator commissioned an experienced tool designer to supply the training data, based on previous design projects, and this was used to study the problem to be solved and select the most appropriate ANN system. They studied a wide range of section geometries, and from this information alone, predicted the number of forming stations that would be required. Constructing the training data in this manner is similar to the development of an expert system. Instead of defining a set of rules from the knowledge that is gleaned from the discussions with a human expert, the information is used to train the ANN system. Consequently, the LVQ network will predict the number of forming stations based solely on the section geometry. The industrial collaborator will include additional forming stations if this is required, for example due to variables such as sheet material, prior to carrying out the final cost estimation (Downes et al, 2006).

5. CONCLUSION

There are many different ANN systems which have been successfully applied to a wide range of applications. Prior to deciding on the most suitable ANN system to use it is essential that the problem and the required solution is carefully analyzed. Additionally, it is advisable to decide on the strategy to be applied if the testing phase results in a poor network performance. The training of ANN systems can be time-consuming, although this is not always the case because there are examples, such as the RBF network, that have rapid training times.

The performance of the network will always be dependent on the data used in the training phase. Conflicting information will be tolerated but a sufficient amount of accurate knowledge must be provided to allow the desired mapping relationships to be formulated. It is not always necessary to have a large quantity of training data. If the network performance is unsatisfactory a frequently used remedy is to change or increase the size of the training data, but for many applications this is not a feasible option. In some cases, resourceful tactics are required to obtain the required training data.

The features extracted from the problem domain and represented in the input vectors must provide sufficient information. When a FFBP network is used, however, it is helpful if the dimension of the input vector is relatively small because the training times will be reduced and it is easier to obtain a satisfactory network performance.

An ANN system is a feasible approach if the problem domain involves uncertainties and subjective knowledge. This is particularly true if the problem is a pattern-matching task and the categorization is largely unknown. If a solution to the problem can be obtained using a rule-based algorithm there is a good possibility that an ANN system will not be the best option.

ACKNOWLEDGEMENTS

The authors are grateful for the support and advice given by Tony Cotterill of Bradbury (UK) Ltd. and Michael Castellucci of Hadley Group Ltd.

REFERENCES

Anderson, J., Rosenfeld, E., 1988, *Neurocomputing: Foundations of research*, MIT Press, Cambridge, M.A., 18-27.

AutoCAD Reference Manuals, 1996, *AutoLISP programmers reference*, AutoDesk Ltd.

Bishop, C. M., 1995, *Neural networks for pattern recognition*, Oxford University Press, Oxford.

Bradbury (UK) Ltd., 2008, *www.bradburygroup.net*

Downes, A., Hartley, P., 2006, Using an artificial neural network to assist roll design in cold roll-forming processes, *J. Mat. Proc. Techn*., 177, 319-322.

Downes, A., Hartley, P., 2006, The use of an artificial neural network to estimate costs in cold roll-forming processes, *Computer Methods in Materials Science*, 6, 3-4, 203-212.

Downes, A., Hartley, P., Pillinger, I., 2004, A storage and retrieval system for roll-forming design data using a neural network, *Steel GRIPS: Journal of Steel and Related Materials*, 2, 235-239.

Fausett, L. V., 1994, *Fundamentals of neural networks: architecture, algorithms and applications*, Prentice-Hall.

Hadley Group (Smethwick), UK., 2008, *www.hadleygroup.co.uk*

John, J., Sikdar, S., Kumar Swamy, P., Das, S., Maity, B., 2008, Hybrid neural-GA model to predict and minimise flatness value of hot rolled strips, *J. Mat. Proc. Techn*., 195, 314-320.

Kim, H.S., Koç, M., Ni, J., 2007, A hybrid multi-fidelity approach to the optimal design of warm forming processes using a knowledge-based artificial neural network, *J. Mat. Proc. Techn*., 47, 211-222.

Kohonen, T., 2001, *Self-organizing maps*, Springer-Verlag, London.

Kurkova, V., 1992, Kolmogorov theorem and multilayer neural networks, *Neural Networks*, 5, 501-506.

MatLab Reference Manuals, 2000, *Neural network toolbox users guide*, The Mathworks Inc.

Ozerdem, M.S., Kolukisa, S., 2008, Artificial neural network approach to predict mechanical properties of hot rolled, nonresulfurized, AISI 10xx series carbon steel bars, *J. Mat. Proc. Techn*., 199, 437-439.

Peng, Y., Liu, H., Du, R., 2007, A neural network-based shape control system cold rolling operations, *J. Mat. Proc. Techn*., doi:10.1016/j.jmatprotec.2007.09.075.

Powel, M., 1987, *Radial basis function for multivariable interpolations*, Algorithms for Approximation, Clarendon Press, Oxford, 143-167.

Riedmiller, M., Braun, H., 1993, A direct adaptive method for faster back-propagation learning – the bprop algorithm, *Proc. of IEEE Int. Conf. on Neural Networks*, 1, 586-591.

Swingler, K., 1996, *Applying neural networks: a practical guide*, Academic Press

Wasserman, P., 1993, *Advanced methods in neural computing*, Van Nostrand Reinhold.

SZTUCZNA SIEĆ NEURONOWA W ZASTOSOWANIU DO PROJEKTOWANIA PROCESU GIĘCIA PROFILE Z BLACH

Streszczenie

Istnieją różne typy sztucznych sieci neuronowych mające różne możliwości i charakterystyki, które pozwalają na szeroki wachlarz ich zastosowań. Celem niniejszego artykułu jest analiza zastosowania trzech systemów opartych na sztucznych sieciach neuronowych, wykorzystywanych do projektowania procesu gięcia profile z blach. W pracy zamieszczony jest krótki opis każdego z systemów, który wyjaśnia istotność przyjętej architektury sieci neuronowej oraz zastosowanej metody uczenia. Szczególny nacisk położono na wykazanie ważności doboru odpowiedniego systemu do projektowanego procesu gięcia.

Submitted: February 6, 2008
Submitted in a revised form: February 13, 2008
Accepted: February 13, 2008