1. Non linear approaches in textiles: the Artificial Neural Networks example

by
M. Rangoussi and A. Çay

Department of Electronics Engineering, Technological Education Institute of Piraeus, Greece
Department of Textile Engineering, Ege University, Izmir, Turkey

1.1 Introduction

1.1.1 Engineering and Modeling

Engineering designs and constructs all kinds of devices, equipment, technical systems, large production units or public works aiming to improve the quality of human life and to raise the living standards. Engineering also designs and construct all tools, machinery and methods necessary for these tasks. To accomplish its mission, engineering makes use of all the great results of science and technology, along with the innovative thinking of engineers all over the world. The outcomes of all this effort comprise the so-called 'man-made' or 'artificial' component of the world that surrounds us, as opposed to the 'natural' component (earth, fauna, flora, human beings and climate). Now-a-days, engineering works cover all dimensions from micro- and nano- to giga- and terra- scales and expand their range of activities to the space and faraway planets and stars.

Textiles is one of the most ancient and most close to the human being engineering fields: it goes back to ancient Egypt, where the Pharaohs wore elaborate hand gloves made from cotton threads, to ancient China and probably even further back in human prehistory. For thousands of years, textiles have been clothing the human body for protection and survival, for distinction of hierarchy, role and responsibility, for celebration or mourning, for the joy of life and the sorrow of death.

Apart from clothing, yarns and fabrics have found millions of other uses, ranging from traditional investment of interiors (tents, furniture, buildings, cars, airplanes), sails for vessels or media for stocking goods to the most exciting modern uses (aesthetics and fashion, healthcare, military and safety) and further on to the smart, multi-functional textiles of the modern era, equipped with sensors and 'gifted' with artificial intelligence so as to respond to our needs or feelings!

A major tool in the effort of scientists and engineers to understand nature and its laws and exploit this knowledge to construct better artificial devices and systems has been the analysis and modeling of systems. This is achieved by means of mathematical and physical sciences, at various levels of abstraction and at various levels of approximation as well. Models of real life systems and of their functionality have evolved from early forms of architectural miniatures of landscapes, buildings, bridges, airports, factories, vehicles, etc. made of clay, cork, plastic or other materials, to the modern, electronic, three-dimensional models created by sophisticated computer graphics software.

It is important to keep in mind that it is the mathematics relations, simple or complex, lying behind all such software, that govern the drawing of the geometrical forms and shapes, texture and lighting effects that produce the exquisite, photo-realistic models on a computer screen. In turn, these mathematics relations have been formulated by scientists on the basis of analysis of the real system they had carried out, in an attempt to approximate its functionality by a set of relations of the minimum complexity possible; yet, it should adequately resemble the real system.

Although intuitively an accurate and detailed model is expected to be more useful, often an approximate, simplified model is to be preferred, as it lends itself to immediate use while it retains the key characteristics of the system it models. It is parsimonious, in the sense that it is not overloaded with details that cannot be appreciated by the user of the model. Still, it provides us with a clear-cut view of the real-world prototype it models.

What is the practical value of a model? It has to do with prediction. A model helps the designer predict the behavior, static or dynamic, of the device or system being designed, before taking the cost and dedicating the effort to actually construct it. This results in considerable savings of effort and cost. Loops of testing, corrections and changes for the improvement of the initial design are common practice in the design and construction of products or services. Fortunately, models allow us to loop through correction steps at a considerably low level of cost and take the construction cost for the real-world system only after the design has been finalized.

In textiles engineering, models built to describe the properties, characteristic measures and dynamic behavior of yarns, fabrics and final products have been valuable design tools. Of great practical value are models that predict the properties and behavior of the produced fabric based on the properties of the yarns and weaving pattern employed.

1.1.2 Is this a deterministic or a stochastic world?

In their strife to obtain 'good' models, scientists have gradually shifted from the deterministic to the so-called stochastic approach. The difference between the two terms is essential to the way the world around us is perceived and interpreted by humans; in fact, expressed under various forms, the dilemma whether this is a deterministic or a stochastic world has long been discussed and argued by science, philosophy and religion. Leaving that aspect of the discussion to the knowledgeable, engineering proceeds to exploit the best of the two approaches, selecting per case the one that produces adequately good models for the problem at hand.

- The deterministic approach heeds that the world around us, its natural and artificial components alike, can be fully and exactly described by mathematics relations. At an increased level of complexity – often prohibitively high for all practical purposes – equations and inequalities, linear or nonlinear, can describe in full detail all that happens around us, including laws of nature, behavior of beings and functionalities of constructed, artificial systems. Scientists formulate the sets of such relations by study and analysis of the real-world systems and phenomena; next, engineers simplify them enough to be manageable by contemporary mathematics and software – but not too much, so as to retain the essential features of the real-world prototype they model. The level of approximation in the description of systems and phenomena thus obtained is varying and it is decided per occasion by the engineers.

- The stochastic approach heeds that there exist factors affecting behaviors and functionalities of beings and systems that cannot be fully described by equations, as they are essentially random in nature. Noise, either acoustic or electronic, mechanical friction, the behavior of the atmosphere as transport media for wireless communications are but a few typical examples of factors that render a system random or stochastic. Laws of nature are no exception, either. The set of all possible outcomes or values of a random factor form an ensemble; the random factor is described by measures averaged over the ensemble of all its possible forms, rather than with exact equations per case. Stochastic equations are thus obtained to describe or model stochastic systems and the notion of probability (that one of all possible outcomes contained in the ensemble will eventually occur at a given time) comes into picture.

Major engineering tasks such as detection of events, pattern recognition and object classification receive stochastic answers under the stochastic approach. A certain fabric flaw probably (or with probability p %) is caused by a certain machinery fault, a certain fabric is more likely (higher probability) to belong to class A than to class B, or it will exhibit a given property with probability p %. In contrast, the deterministic approach provides 'binary' or 'crisp' answers of the type: belongs / does not belong, exists / does not exist, will exhibit / will not exhibit, etc. Both types of answers may be correct or wrong; the stochastic approach, however, is closer to the way a human expert would make and express decisions.

1.1.3 Is this a linear or a non-linear world?

This is clearly a non-linear world, all scientists answer in concordance. Non-linearity is the rule without exception in nature and all natural factors: materials, constructions, beings and behaviors. Linearity is an abstraction adopted by our perception in order to simplify nature at a level where we would be able to comprehend, describe and interpret it. A straight line or a perfect plane, as these are defined in Euclidean geometry, are not to be seen anywhere in nature; yet, they are successful simplifications or approximations of a tight string or a calm water or other liquid media level.

Inasmuch as they bear a correspondence to the real-world objects, such approximations are valuable help for common people and scientists alike: the former use linear approximations to cope with everyday life problems and calculations of distance, area, value, time, etc., while the later exploit them to express and test theories and communicate results. Scientists have another good reason to seek linear approximations: linear mathematics have traditionally been far more advanced than non-linear mathematics, the later having progressed to a level of practical interest only recently. Tools and methods at the avail of scientists and engineers have been linear in their vast majority, prompting them to attempt to 'linearize' essentially non-linear problems.

It can be argued that linearity is a matter of 'distance' one takes from the object or behavior under study. Indeed, if one 'zooms in' to the surface of an object made of a given material, irregularities, aberrations, flaws and fluctuations – inherent to all materials – will appear; upon 'zooming out' enough, flaws disappear and the ideal straight line or plane view prevails. In fact, there are areas or parts or aspects of the object under study where the linear approximation is 'reasonable', i.e. it lies at a smaller distance or 'leaves small error' to the real, non-linear nature of the object, and other parts where such condition does not hold. Different linear approximations may be required at different parts of the object. A sigmoid line, for example, may be crudely approximated by three different straight lines as in Figure 1.1; this is a piece-wise linear approximation. These three lines constitute a linear model of the actual sigmoid line.

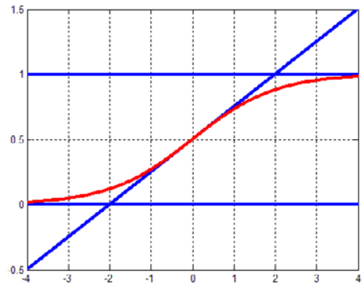

Figure 1.1: The non-linear sigmoid function (red curve) is approximated by three straight lines (blue lines). The horizontal line at y = 0 is a good approximation for x < -2; the 'diagonal' line for -2 < x < 2; the horizontal line at y = 1 for x > 2.

It may be argued that a linear approximation by five different lines would be preferable, as it would leave smaller error; however, this is a more complex model that requires more computations. In general, there is a trade-off between approximation (model) complexity and approximation error, calling for a balanced decision.

When a real-world problem is cast into a linear model via a linear approximation procedure, linear mathematics (typically coded into a software tool) are employed to yield the solution, which is tested at a simulation level and the model is corrected accordingly. These steps loop until some criterion is met. The final solution is then materialized by the construction of the actual object. Yarn, tissue, fabric and final textile product design and construction are no exception to this approach.

1.2 Artificial Neural Networks (ANNs)

1.2.1 Is it all hype?

A part of it is – was, rather – hype, but certainly not all, scientists reply today. There's true value in them; only, one has to know what to expect. Only a couple of decades ago, excitement over the merits of these new tools drove expectations too high: they were claimed to be universal problem solvers. Interestingly enough, now that the dust has cleared, they still hold a title of universality – this time by strict mathematical proof: they are universal function approximators. If only for this property, ANNs deserve a formal introduction. In light of the discussion held in the previous section,

Artificial Neural Networks are non-linear, stochastic mathematical models.

1.2.2 ANN types and structures

They are inspired by – and named after – the neural system of biological organizations, a network built from *neurons*, *axons*, *dendrites* and connection points know as *synapses*, as neuroscience explains. Through this network, information flows in the form of electric signals from the peripheral sensors to the brain (*sensing* direction) and control orders flow from the brain to the peripherals (*actuating* direction). Similarities do not hold any further, however; the nervous system and the brain are far too complex to be fully understood or modeled by science as yet, while ANNs are governed by simple – even if non-linear – relations.

Artificial, as opposed to biological, neural networks are built of nodes called *neurons* or *processing elements (PEs)* which are interconnected by links bearing *weights*. Each node receives a vector of inputs, processes them non-linearly in the general case and produces a single output. Figure 1.2 shows a simplified example of a node that accepts a vector of three inputs [x_0, x_1, x_2], weights them by [w_0, w_1, w_2], respectively, processes their sum z by the non-linear 'activation function' (a sigmoid, a hard-limiter or other) to produce a single output y.

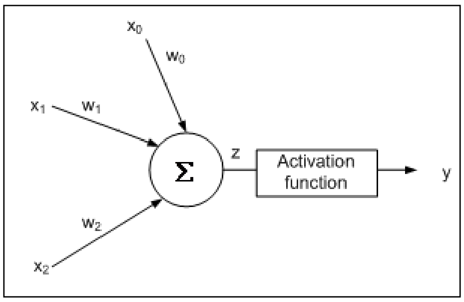

Figure 1.2: A simplified example of a node or PE or neuron of an ANN: input x are weighted by weights w. The weighted average z is processed by the non-linear activation function to produce output z.

Nodes are organized into *layers* arrayed into a sequence; output values of a layer serve as input values to the next layer. In general, an ANN is a *multi-layer* construction. In an ANN of L layers, the first (L-1) layers are called *hidden* while the last, L-th layer is the *output layer*. Figure 1.3 shows a simplified example of an ANN with three layers of nodes (input, hidden, output).

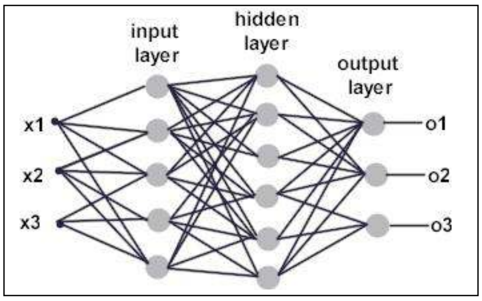

Figure 1.3: A simplified example of a three-layer ANN (input, hidden, output). Input vectors are three-dimensional ([x_1, x_2, x_3]). All inputs are 'fed' to all input layer nodes. All nodes of a layer are connected to all nodes of the next layer. Weights at connections are not shown for simplicity.

It is interesting that networks with as little as only two layers (one hidden and one output) can solve really complex problems. In fact, it has been proved that a two-layered network, appropriately structured and trained, can *approximate arbitrarily well any function* that has a finite number of discontinuities, thus gaining the title of universal approximators for the ANN family. How is this achieved? What other kinds of problems can ANNs solve?

Input data, typically in the form of vectors of measurements X, are introduced to the first layer. Data proceed through – while being processed by – the ANN, from layer to layer, to the *output layer*, where they form the vector of output values or (stochastic) decisions Y. Data flow between successive layers can be unidirectional (from a given layer to the next one in sequence) in a *feedforward network* or bidirectional (proceeding forward to the next layer *and* looping back to the previous one) in a *feedback network*.

Although more sophisticated forms exist, the *typical* processing relation performed by the nodes of layer L_i, in a feedforward network, in order to transforms the data vector X_i into the data vector X_{i+1}, is *a weighted average passed through a non-linear function*, formulated as

$$X_{i+1} = \boldsymbol{f}_i \,(a_i\, X_i + b_i). \tag{1.1}$$

Here $\boldsymbol{f}_i(\cdot)$ denotes the *non-linear transfer function* of layer L_i, common to all nodes in this layer but possibly varying across different layers. The log-sigmoid, the hard-limiter and the hyperbolic tangent are typical non-linear function examples. The linear option is retained for $\boldsymbol{f}_i(\cdot)$ if a *linear layer* is necessary for solving a given problem. $\{a_i\}$ denotes the vector of *weights* and $\{b_i\}$ the vectors of additive constants (offsets or *biases*) that render the linear combination affine.

Network 'architecture' (i.e., the number of layers, number of nodes per layer, possible connections among nodes and layers, weights and non-linear functions employed) complexity is commensurate with the complexity of the system the ANN is asked to model. A variety of different architectures have been proposed and successfully implemented so far. Perceptrons, multilayer perceptrons, feedforward and feedback, generalized regression, associative, hebbian learning, radial basis, linear vector quantizer and many other network types are available for testing and use. Selection of the best architecture is empirical; rules of thumb rather than closed form solutions are available.

1.2.3 ANN functionalities

In a feedforward network, input data vector X undergoes a series of L such transformations that change both the values and the number of the vector components, until it is handed out as output vector Y:

$$X=X_1 \rightarrow \boxed{L_1} \rightarrow X_2 \rightarrow \boxed{L_2} \rightarrow X_3 \rightarrow \cdots \rightarrow X_i \rightarrow \boxed{L_i} \rightarrow X_{i+1} \rightarrow \cdots \rightarrow X_L \rightarrow \boxed{L_L} \rightarrow Y$$

If viewed as one global system, the ANN structure proposes a relation between input vector X and output vector Y, of the form

$$Y = \boldsymbol{F}(X), \tag{1.2}$$

where $\boldsymbol{F}(\cdot)$ represents the *nested* application of the $\boldsymbol{f}_i(\cdot)$*s* across successive layers L_1 to L_L:

$$\boldsymbol{Y} = \boldsymbol{F}(X) = \boldsymbol{f}_L(X_L) = \boldsymbol{f}_L(\boldsymbol{f}_{L-1}(X_{L-1})) = \quad \cdots \quad = \boldsymbol{f}_L(\boldsymbol{f}_{L-1}(\boldsymbol{f}_{L-2}(\cdots(\boldsymbol{f}_2(\boldsymbol{f}_1(X_1)))\cdots))). \tag{1.3}$$

Although each $\boldsymbol{f}_i(\cdot)$ is a simple function or model, the ‘cumulative’ effect across all layers in a multi-layered network produces rather complex functions. What is the kind of problems that such functions – and, consequently, ANNs can address successfully? They can be grouped under three major categories:

1. *Function approximation*, including system identification, modeling and prediction,
2. *Pattern classification*, including pattern recognition and decision making, and
3. *Data processing*, including clustering, filtering and compression.

It is worth to note that under these three categories falls a considerable majority of engineering problems, either directly or after suitable manipulation. How are these demanding tasks accomplished by an ANN? It has to do with the *adaptation property* of ANNs. It would be a rather simple task to build up an ANN model and code it into software, if it weren’t for the fact that it is an adaptive model: weights $\{a_i\}$ and biases $\{b_i\}$ are repeatedly adjusted to best suit the data at hand, via an algorithm that is typically iterative, until an optimality criterion is met. A variety of iterative algorithms have been proposed so far; different algorithms are better suited to different types of problems. What is crucial here is the fact that these iterative algorithms have been proved to converge to a solution given an adequate set of sample data for training.

The development of an ANN approach in order to solve a given engineering problem proceeds in three phases: (a) initial selection of the ANN type, architecture and training algorithm, (b) training phase and (c) testing phase.

- During the training phase the ANN 'learns' the rules that govern the system under investigation through a set of examples (the training set) presented to its input; each input vector within the training set is associate with a correct output (answer). The iterative algorithm employed to 'train' the ANN adapts its weights iteratively, based on the difference between actual and correct output (error). Weights are adjusted until error is minimized; upon convergence the training phase ends.
- During the testing phase, which represents the actual ANN long-term functionality of interest, unknown examples are presented to its input; using the weight values adjusted through training, the ANN processes each unknown input and produces the corresponding output. This output value may represent things as different as class membership, probability of an event, estimated value of a parameter, etc. Correct outputs produced in response to unknown inputs prove the ANN's ability to 'generalize', i.e. to extract 'knowledge' or 'rules' from the set of examples that are then applied to unknown cases. The 'generalization' property ultimately shows that
  - the ANN type, architecture and training algorithm chosen are suitable for the problem at hand; and
  - the training set used was 'rich' (representative of all possible cases) enough to guarantee successful operation during testing phase.

What if 'generalization' is not achieved? This means that either the training set was not rich enough or the ANN selection was not successful (in total or in its parametrization). In the former case a different approach may be more suitable than ANNs since more data are not often easy to acquire; in the latter case the process loops back to redesign the net or its parametrization or the training algorithm and to go through the training phase once more.

All in all, the ANN approach is both complex and sensitive; it is worth taking the pain to resort to it only after straightforward, linear methods have failed to address the problem at hand or when there is strong evidence of non-linearity in the data, coming from prior information.

1.3 ANNs in textiles engineering

Artificial Neural Networks, in both their functions as approximators and as classifiers, have found use and successful application in a variety of problems arising in textiles engineering. They have been used to 'estimate' values of yarn-, fiber- or fabric-related properties on the basis of simple, measurable structural parameters, before the actual construction or fabrication step takes place. They have been used to detect failures, faults, or other events of interest from signals recorded or images taken during the yarn, fiber of fabric production process. It can be safely stated that within the textiles engineering area, ANNs have been exploited in all classic engineering problems, such as detection, classification, modeling, estimation and pattern analysis. In the following paragraphs, two such examples are outlined.

1.3.1 A function approximation problem example: prediction of fabric air permeability

Air permeability of fabrics is an important property in textiles because it determines both the comfort of the final product (garment) and the behavior of the fabric during the vacuum drying phase of its processing. Therefore, prediction of the air permeability of a fabric before its actual construction is a task of practical interest. Air permeability is known to depend on the material of the yarn and the micro-structural parameters of the fabric, through rather complex, non-linear relations. Porosity of the fabric offers a path to calculate air permeability; unfortunately, there is no standardized method to calculate or directly measure porosity, especially for dense fabrics. On the other hand, micro-structural parameters of the fabric such as warp and weft densities or mass per unit area of fabric can be measured with adequate accuracy.

Linear multiple regression analysis, carried out in order to investigate the degree of linearity of the relation among air permeability and the three aforementioned micro-structural parameters, reveals that an 85% of the variability in air permeability values can be linearly explained by the variability in the three parameters while the rest 15% calls for a non-linear approach. In that case an ANN can be designed and trained to predict air permeability values (output) from input vectors that contain micro-structural parameter measurement data.

A Generalized Regression Neural Network (GRNN) is employed for this task. This is a member of the Radial Basis Function ANNs that are known to be universal approximators appropriate for problems that present radial symmetry of the data space, as is the case at hand. Another advantage is that the GRNN training iterative algorithms converge rapidly. A GRNN contains two layers of neurons, each consisting of N neurons, where N is the cardinality of the training set (number of the input - output pairs available). The first (hidden) layer consists of radial basis function (RBF) neurons while the second (output) layer consists of linear neurons of special structure, allowing for real-valued outputs. A single, real-valued output value is employed here; it is the air permeability value predicted by the ANN on the basis of the three micro-structural parameters of the fabric under design. Indeed, after training with a set including various types of fabrics, the specific ANN exhibits satisfactory generalization, meaning that it can accurately predict air permeability values of fabrics not included in its training set – yet, of micro-structural parameters within the same range as those in the training set.

Figure 1.4 shows prediction results on a set of fabrics of six (6) different knitting patterns across five (5) different parameters yielding thirty (30) different fabric sample cases. Twenty four (24) samples are used for training and six (6) for testing. Air permeability values are on the vertical axis while fabric sample case index is on the horizontal axis. Upper plot shows excellent agreement between real (red stars) and ANN predicted (blue circles) air permeability values across the 24 cases of the training set, here used as the testing set. Lower plot shows the corresponding good agreement for the 6 cases of the testing set that are not used for training.

Calculation of the average error (difference between real and estimated air permeability output value) across different samples reveals that only 3.3% of the total variability in the air permeability value is not 'explained' by the non-linear, ANN approach, as compared to the 15% of variability left 'unexplained' by the linear method.

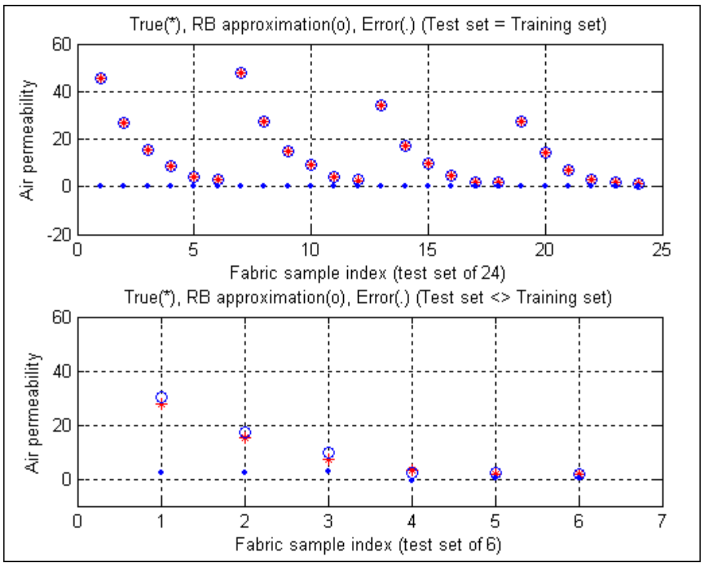

Figure 1.4: Air permeability real (red stars) and ANN predicted (blue circles) values for 30 fabric samples of different micro-structural parameters. Upper plot: training set used as testing set (24 cases). Lower plot: testing set unknown (6 cases).

In a feedforward network, input data vector X undergoes a series of L such transformations that change both the values and the number of the vector components, until it is handed out as output vector Y:

1.3.2 A classification problem example: classification of faults in circular knitting machines

The automated supervision of the knitting process is of high interest so as to avoid (i) the waste of material and (ii) the increase of production cost. Knitted fabrics produced by circular knitting machines that involve numerous moving parts may come out defective as a result of failure of the machine; depending on the type of failure, the product may be of reduced quality and price or altogether unsuitable for further use. Automated detection and classification of the various types of knitting machine failures is therefore of great practical interest. Indeed, if issued in real time, an alarm or call for technical support and repair will result in considerable time and cost savings. Yarn tension signal is a quantity that can be monitored for early machine failure detection. Figure 1.5 shows a yarn tension signal recording under normal (upper plot) and abnormal (lower plot) operating conditions of the knitting machine.

Correlation the different types of mechanical failures with the possible corresponding differences in the respective tension signals would allow for the classification of the machine fault type based on the classification of the event present in the recorded tension signal. This is a complex and demanding task for human experts; it is therefore expected to be demanding under automated performance as well. The non-linear ANN approach is investigated for the detection and the classification tasks.

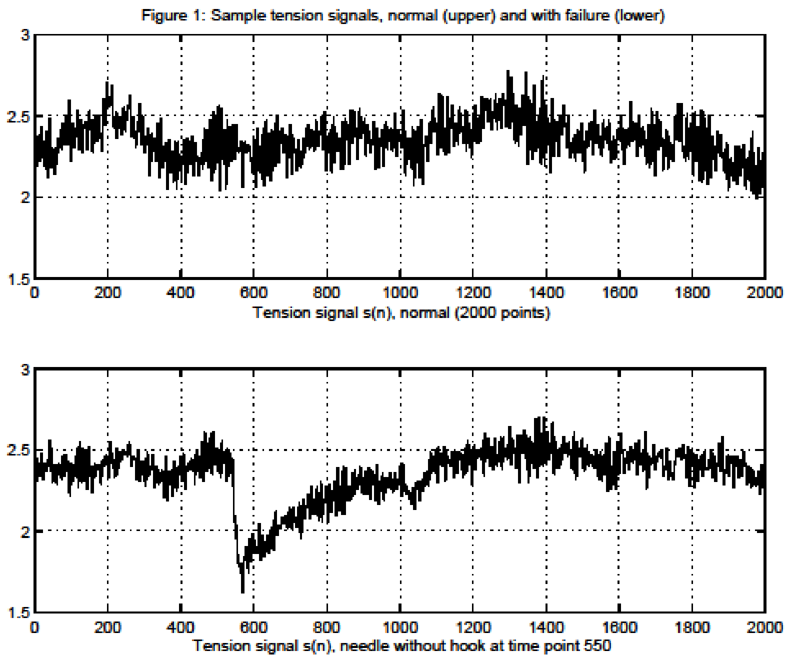

Figure 1.5: Yarn tension signal recording under normal (upper plot) and abnormal (lower plot – needle without a hook) knitting machine operating conditions.

As a first step, a set of 'features' or characteristic quantities have to be extracted from the signal. This step reduces the dimensionality of the problem from the dimension of the recorded signal length down to that of the number 'features' selected and extracted. These features, however, should retain and convey all information present in the signal that will subsequently allow for the classification of the signals into different classes. Taking into account the non-stationary nature of the yarn tension signal, a set of time-frequency analysis features are selected; these are based on the (pseudo-) Wigner-Ville Distribution (WVD) of the yarn tension signal. It is a two dimensional distribution of the signal power across time and frequency axes, that extends the notion of Fourier Transform spectrum of stationary signals to cover the case of non-stationary signals. Figure 1.6 shows this two dimensional WVD feature for the yarn tension signal cases of Figure 1.5 (left: normal, right: needle without a hook).

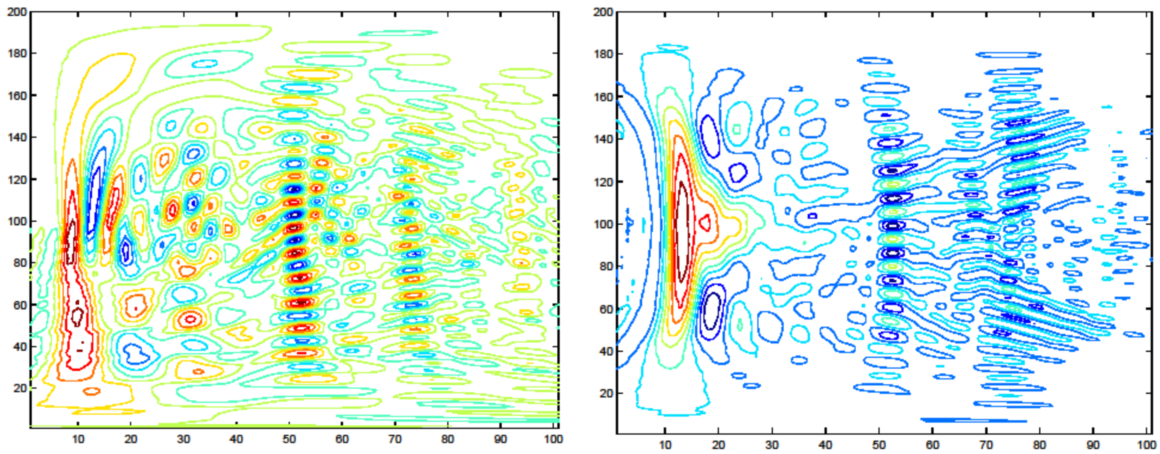

Figure 1.6: Two dimensional Wigner-Ville Distribution feature computed from the yarn tension signal of Figure 1.5 (left: normal operation, right: needle without a hook).

The plots in Figure 1.6 are contours obtained by cross-sectioning the two dimensional, landscape-like WVD characteristic quantities. Horizontal axis is frequency while vertical axis is time, centered on the failure time point of figure 5 (lower plot, n=550). Color depicts signal power at a given time-frequency neighborhood, scale increasing from blue to red. After suitable reduction back to one dimension, feature vectors are obtained and presented to a Learning Vector Quantizer (LVQ) type of ANN classifier. The LVQ ANN architecture is selected for its ability to handle input vectors of high dimensionality, as is the case at hand; yet, at the cost of longer training iterations. It is a two-layered architecture with a first, competitive layer that classifies inputs in sub-classes and a second, linear layer that groups sub-classes into target classes. The target class index is the single output.

The LVQ designed for this problem is trained to classify input vectors into (a) two and (b) three distinct classes of machine failures. Correct classification scores are varying between 75% and 90%, case-dependent. These are satisfactory results given the complexity of the task and the fact that they are obtained directly, without any ANN architectural parameter trimming. However, they reveal the sensitivity and the amount of computational effort required by the ANN approach.

1.4 Discussion

Non-linear methods have been attracting research interest in complex engineering problems where the linear approach is not adequate. Artificial Neural networks are but an example; Fuzzy Logic, Support Vector Machines, Genetic Algorithms, Soft Computing and many other alternatives are open for investigation as to their appropriateness to handle a given problem. These methods are valuable when tackling complex, demanding problems; yet, they are computationally demanding, they may converge to optimal or to suboptimal solutions while their performance is case-dependent. The right choice is possible only after the engineer has deeply studied and understood the nature of the problem at hand and has formed a clear view of the type of answer and the accuracy of answer sought. The cost of resorting to non-linear approaches has always to be taken into account and justified: there still exists the possibility that the linear method returns a solution whose quality and accuracy are satisfactory!

1.5 Literature

Araujo, M., Catarino, A., Hong, H., "Process Control in for Total Quality in Circular Knitting," AUTEX Research Journal, vol. 1, No 1, pp. 21–29, 1999.

Araujo, M., Catarino, A., Hong, H., "Quality Control in Circular Knitting by Monitoring Yarn Input Tension," Proceedings of the 79th World Conference of The Textile Institute, Chennai, India, vol. 1, pp. 167–182, February 1999.

Backer, S. (1951). The relationship between the structural geometry of a textile fabric and its physical properties, Part IV: Intercise geometry and air permeability, Textile Research J., vol. 2, pp. 703–714.

Bhattacharjee, D., Kothari, V.K. (2007). A neural network system for prediction of thermal resistance of textile fabrics. Textile Research Journal, 77 (1), pp 4–12.

Brasquet, C., LeCloirec, P. (2000). Pressure drop through textile fabrics-experimental data modeling using classical models and neural networks. Chemical Engineering Science, 55, pp. 2767–2778.

Cay, A., Vassiliadis, S., Rangoussi, M. & Tarakcioglu, I. (2007). Prediction of the air permeability of woven fabrics using neural networks. Intl. J. of Clothing Science and Technology, 19 (1), pp 18–35.

Chen, S., Cowan, C.F.N., and Grant, P.M. (1991). Orthogonal least-squares learning algorithm for radial basis function networks, IEEE Transactions on Neural Networks, vol. 2(2), pp. 302–309.

Claasen, T.A.C.M., Mecklenbrauker, W.F.G., (1980). "The Wigner Distribution – A tool for time-frequency analysis," Philips J. Res., vol. 35, Parts I, II, III.

Cohen, L., (1986). "Generalized Phase-Space Distribution Functions," Journal of Math. and Physics, vol. 7, pp. 781–786.

Crochiere, R.E., Rabiner, L.R., (1983). "Multirate Digital Signal Processing," Prentice-Hall, New Jersey, USA.

Cybenko, G. (1989). Approximations by superpositions of sigmoidal functions. Mathematics of Control, Signals, and Systems, no. 4, pp. 303–314.

Gurumurthy, B.R. (2007). Prediction of fabric compressive properties using artificial neural networks. Autex Research Journal, 7 (1), pp. 19–31.

Elman, J.L. (1990). Finding structure in time, Cognitive Science, vol. 14, pp. 179–211.

Ertugul, S. & Ucar, N. (2000). Predicting bursting strength of cotton plain knitted fabrics using intelligent techniques. Textile Research Journal, 70 (10), pp. 845–851.

Guruprasad, R. & Behera, B.K. (2010). Soft computing in Textiles. Indian Journal of Fibre and Textile Research, vol. 35, pp. 75–84.

Haykin, S. (1998). Neural Networks: A Comprehensive Foundation, Prentice Hall, ISBN 0132733501, New York.

Hertz, J., Krogh, A. & Palmer, R.G. (1991). Introduction to the Theory of Neural Computation, Addison-Wesley Longman Publishing Co., Boston, MA, USA.

Hornik, K., Stinchcombe, M. & White, H. (1989). Multilayer Feedforward Networks are universal approximators. Neural Networks, vol. 2, pp. 359–366.

Jain, A.K., (1997). "Fundamentals of Digital Image Processing," Prentice-Hall.

Janssen, A.J.E.M., (1982). "On the locus and spread of Time-Frequency Pseudo-Density Functions," Philips J. Res., vol. 37, pp. 79–110.

Keeler, J. (1992). Vision of Neural Networks and Fuzzy Logic for Prediction and Optimisation of Manufacturing Processes, In: Applications of Artificial Neural Networks III, vol. 1709, pp. 447–456.

Kohonen, T., (1990). "Self-Organization and Associative Memory," 2nd ed., Springer-Verlag, New York.

Kohonen, T., (1990). "Improved versions of LVQ," Proceedings of Intl. J. Conf. on Neural Networks '90, vol. 1, pp. 545–550.

Lin, D.-T. (1994). The Adaptive Time-Delay Neural Network: Characterization and Applications to Pattern Recognition, Prediction and Signal Processing. PhD thesis, University of Maryland, USA.

Lin, J.-J. (2007). Prediction of yarn shrinkage using neural nets. Textile Research Journal, 77(5), pp. 336–342.

Lippman, R.P. (1987). An introduction to computing with neural nets. IEEE ASSP Magazine, pp. 4–22.

Majumdar, P. K. (2004). Predicting the breaking elongation of ring spun cotton yarns using mathematical, statistical and artificial neural network models. Textile Research Journal, 74(7), pp. 652–655.

Matlab, (2005). MATLAB 7 R14, Neural Network Toolbox User's Guide, The MathWorks Inc., Natick, MA, USA.

Ramesh, M.C., Rjamanickam, R. & Jayaraman, S. (1995). The prediction of yarn tensile properties by using artificial neural networks. Journal of Text. Inst., vol. 86, no.3, pp. 459–469.

Rich, E. & Knight, K. (1991). Artificial Intelligence, McGraw-Hill, New York, USA, pp. 487–524.

Stylios, G. & Sotomi, J.O., (1996). Thinking sewing machines for intelligent garment manufacture. Intl. Journal of Clothing Science and Technology, vol. 8 (1/2), pp. 44–55.

Stylios, G. & Parsons-Moore, R. (1993). Seam pucker prediction using neural computing. Intl. Journal of Clothing Science and Technology, vol. 5, no. 5, pp. 24–27.

Vassiliadis, S., Rangoussi, M., Araujo, M. (2002). "Feature extraction and classification of faults in circular knitting machines based on time-frequency techniques," Proc. 2nd AUTEX World Textile Conference, pp. 203–213, Bruges, Belgium.

Zadeh, L. (1994). Soft Computing and Fuzzy Logic, IEEE Software, vol. 11, no. 1-6, pp. 48–56.