Rapid #: -21198959

CROSS REF ID: **2242429375600001811**

LENDER: **LJ1 (UNSW Sydney) :: UNSW Library**

BORROWER: **LE1 (University of Adelaide) :: Barr Smith Library**

TYPE:	Book Chapter
BOOK TITLE:	Industrial Tomography: Systems and Applications
USER BOOK TITLE:	Industrial Tomography: Systems and Applications
CHAPTER TITLE:	Machine learning process information from tomography data
BOOK AUTHOR:	Hoyle, Brian S. ; Machin, Thomas D. ; Mohamad-Sale
EDITION:	
VOLUME:	
PUBLISHER:	
YEAR:	2022
PAGES:	427 - 461
ISBN:	9780128233078
LCCN:	
OCLC #:	

Processed by RapidX: 8/31/2023 7:05:34 PM

Machine learning process information from tomography data

13

Brian S. Hoyle[1], *Thomas D. Machin*[2] *and Junita Mohamad-Saleh*[3]
[1]University of Leeds, Leeds, United Kingdom; [2]Stream Sensing, Manchester, United Kingdom; [3]Universiti Sains Malaysia, Penang, Malaysia

13.1 Introduction

Industrial Process Tomography (IPT) has become a well-established method of gaining a deep multidimensional insight into a wide range of industrial processes. *Machine Learning* (ML), also widely known by the term *Artificial Intelligence* (AI), offers significant advantages for the processing of IPT data. Due to the ill-posed nature of its intrinsic inverse process, and the typical sparse data due to sampling time constraints, results from classical IPT methods are often disappointing, even when sophisticated analytical or statistical modeling is used. In contrast, an ML approach does not rely upon modeling, but instead exploits actual (or simulated) process performance data through a learning network able to crystallize trends and relationships between measurement data and selected process state estimation parameters, from image pixels to interpretation data such as component fractions. Having learned from this, data networks can be implemented with major parallelism to deliver rolling state estimation for fast processes dynamics.

To contrast classical IPT and ML methods, we consider two examples: (i) a batch process to mix several reagents to form a product; and (ii) a continuous process to transport gas/liquid materials. IPT technology typically embodies three core tomographic steps, plus an optional step 4:

S1 – Sensing energy excitation of space/time material distribution of interest;

S2 – Detection of corresponding response;

S3 – Inversion of response data to estimate reconstructed material distribution, often as an image;

S4 – Interpretation of the distribution to identify relevant process characteristics.

Pilot process studies are useful for both optimization of plant design and in trials of new formulations prior to scale-up for mass production. Detailed image data of the internal process state may be valuable to plant and process designers. Many IPT applications have time constraints due to material and process dynamics. In engineering terms, the temporal performance of IPT instrumentation must ensure that Step 1 excitation and Step 2 response detection are completed within the limits of the process dynamics. If response data can be stored for later, Step 3 processing (and any required

Industrial Tomography. https://doi.org/10.1016/B978-0-12-823015-2.00033-9

Step 4) may be carried out after the test completion. Data are typically sparse and ill-posed. As explored in detail in other chapters, a wide range of iterative reconstruction techniques can be applied in the S3 inversion process, ideally exploiting a priori knowledge of process characteristics. Sophisticated techniques such as regularization and optimization methods require considerable computation and typically are more suited for use on stored data, following a pilot scale test.

Operating production processes are focused on efficiency and product quality. For a batch process, this may be on minimizing cycle-time to reduce costs and enhance plant utilization. For a continuous process, the key requirement may be on tracking and controlling the process state within desired levels. In such cases, the temporal performance of IPT instrumentation is critical to deliver timely data for all Steps for quality monitoring and control. For example, continuous in-line monitoring using IPT can remove the need for periodic sampling for laboratory quality checks, whose time lag may be costly in continued production of out of specification product.

In both application cases, S3 can be used to synthesize an image-based dataset with S4 providing further interpreted process information. In the pilot process case, the composite information may be used for design verification. In the operating process case, an image dataset may not be required, as direct estimation of process state for automated monitoring and control is achievable; in effect S3 and S4 are integrated to directly yield process information.

ML offers potential advantages in both pilot and production process cases, through the ability to amass and structure prior process experience data and interpret and generalize this to recognize both prior and newly presented data. ML and AI can be implemented by specific electronic devices, but also for design through software using a conventional digital computer.

In the pilot application case, this may provide enhanced performance; for example, under some conditions the process may exhibit a graded interface between materials, which can be distorted by a conventional IPT regularization algorithm, into a false hard boundary. In the operating production process, the prestored and structured data can, with appropriate hardware, offer fast direct interpretation for automatic process monitoring and control.

13.2 Machine learning methods for information needs

ML methods have developed strongly over the past 20 years, inspired originally of course by biological neural systems; and foundational concepts are now widely used. Early IPT research using ML methods, for example, Nooralahiyan, Hoyle, and Bailey (1993), demonstrated considerable promise, but was not developed further due to major skepticism that estimation results were not verifiable in analytic or statistical terms. As discussed briefly below, ML methods do rest on rigorous foundations, and now have wide ranging science and technology that spans many areas but remain challenging to verification by *reverse engineering*. Due to their impressive performance, they have more recently gained widespread trust, in some cases beyond

sensible expectation. Interestingly, recent IPT meetings have included several new ML developments, notably at the ISIPT ninth World Congress 2018, by: Johansen et al., Lähivaara et al., Machin et al., Yan and Mylvaganma, and by Zheng and Peng.

The following subsections present a short introduction to relevant ML theory and methods focusing primarily upon IPT realization of systems able to meet the demanding monitoring and control case discussed above. To provide a comparative link, the analytical basis of simple ML architecture is contrasted with a corresponding Linear Back-Projection (LBP) case, commonly used due to its implementation simplicity and speed. Although foundations and speed appear comparable, ML solutions have greater potential due to their ability to learn and generalize subtle data interactions, such as the *soft field* effect. These topics are further illustrated in depth using two contrasting design case studies: for complex multiphase flow measurement; and in the sophisticated application of multifaceted IPT sensors for the direct, in-process measurement of rheological properties of stream flows.

Finally, promising new approaches to ML IPT systems are considered. The term *Deep Learning* describes a newer range of ML techniques in which the analytics of data evaluation, organization, and preprocessing is also subject to ML creating a *deeper* learning process. Disordered data are preexamined for subtle relationships which can be exploited to speed and simplify later neural processing structures. The final section also provides a short overview of current design tools and implementation systems.

We include foundational references and others mainly focused on IPT focused ML work. Readers interested in more general ML science and technology are directed to the expansive general literature.

13.2.1 Outline of neural computing

Many texts on ML begin with foundations of neural simulation, based loosely upon a biological neuron, and commonly termed an *Artificial Neuron* (AN), in which the capability to perform a perceptive operation is determined in terms of settable coefficients in its structure. It must be stated for clarity that this is a minimal imitation of the major complexity of a biological neuron. A more detailed introduction centered upon instrumentation applications, coinciding with early IPT applications, is provided by Bishop (1994).

A collection of ANs forming an *Artificial Neural Network* (ANN) provides a variety of processing configurations, typically divided into the classes illustrated in Fig. 13.1, that includes their common name, sometimes based upon the name of their first proposer.

The primary classification separates configurations into two groups: Unsupervised ANNs, designed to self-learn without explicit training, and Supervised ANNs, which must be provided with an explicit training configuration procedure and data. Supervised Configurations are further divided into Static and Dynamic forms. In the Static case, a specific learning configuration uses experiential training data, representing key variations the network will be expected to recognize, and as a basis for generalization.

SUPERVISED		UNSUPERVISED
Feedforward	**Feedback**	
Radial Basis Function	Jordon Simple Recurrent	Kohonen
Perceptron	Mozer Partially Recurrent	Adaptive Resonance
Multilayer Perceptron	Hopfield Fully Recurrent	
Convolutional		
Time Delay		

Figure 13.1 Common configurations of artificial neural network.

13.2.2 Artificial neural network—perceptron

Here we focus upon a Static ANN: a single neuron; a single layer of neurons; and multiple cascaded layers. IPT application networks often exploit ANNs in a configuration commonly called a *Perceptron*, first proposed by Rosenblatt (1962). This combines detailed interpretive performance and straightforward (if extensive, one-time) training. A single Perceptron node is illustrated in Fig. 13.2 to define notation and abbreviations, based here upon biological neuron names for input dendrites (d), and output axon (a).

The Perceptron has a single bias input, d_0; and M measurement data inputs: $d_{1 \ldots M}$, abbreviated as the vector, $\boldsymbol{d}$. All inputs have an applied multiplicative *Weight* value, w_m. The bias weight, w_0, effectively provides a fixed offset value when the bias input is set (conventionally) to constant 1. The resulting weighted sum, s, is generated at the node. This sum is passed to a threshold (or *activation*) function, giving a composite output, a defined by the following:

$$a = f(s) \text{ where } s = \sum_{m=0}^{M} w_m \, d_m \tag{13.1}$$

The function $f(s)$ may have many forms. A classic threshold *step* function causes the neuron to trigger at the step level. For IPT applications, the Perceptron is often used

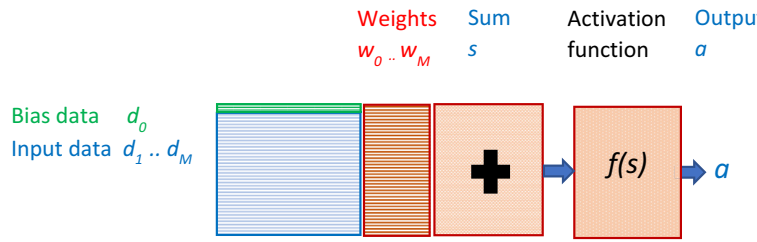

Figure 13.2 Perceptron artificial neuron node.

with a *sigmoid* function. This offers sensitivity to small signals and provides gain control of high values, and is defined as follows:

$$f(s) = \frac{1}{1 + e^{-K_s}} \tag{13.2}$$

where K is a *slope spread* constant, e.g., 0: linear; 0.5: shallow; 5: steep; infinity: step.

This function is nonlinear but is also continuously differentiable, an important property for training procedures discussed below. From Eqs. (13.1) and (13.2), the derivative is the following:

$$f'(s) = a' = \frac{Ke^{-K_s}}{(1 + e^{-K_s})^2} = Kf(s)\,(1 - f(s)) = Ka(1 - a) \tag{13.3}$$

A Network of Nodes, an ANN, consists of multiple nodes arranged in a sequential and parallel format. In an IPT context we may assume an ANN produces a corresponding output instantly, with respect to typical process dynamics. Fig. 13.3 illustrates a corresponding Perceptron ANN with N nodes and M measurement inputs.

Bias and measurement inputs to each node from the *first* to the final Mth input are shown in abbreviated *databus* form. As illustrated, each of the 1 … N nodes has a bias input and, as is usually the case, all M measurement inputs are *fully connected* to each Node, meaning that each input is connected to every node in the layer. For any given task, ANN node and connection structure are determined by design; and the internal settings of the bias and input $(1+M)$ weight values are determined by a learning process using training data. Optimal design can be complex. Obvious key starting points are the number of inputs, and the required outputs. It may also be advantageous to preprocess training set data, to enhance orthogonality and reduce correlated data,

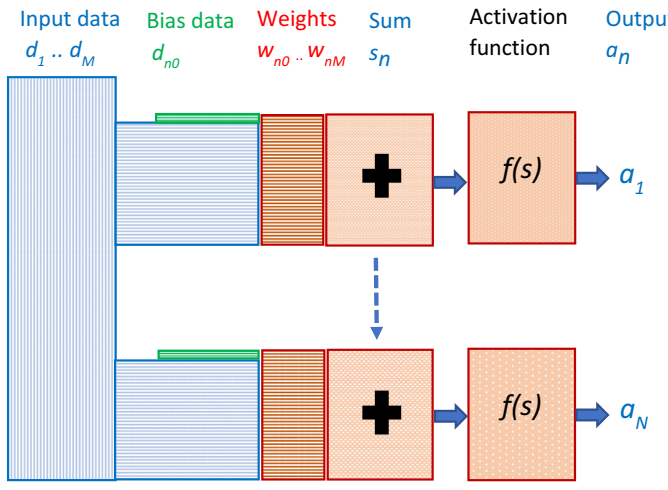

Figure 13.3 Schematic perceptron ANN.

providing a simpler and more efficient ANN; training data are then processed in advance of network design.

For tomographic applications, we infer that an IPT sensor is specified to measure appropriate raw data. Early chapters present a complete range of single and multiple sensing modes. A sensor typically provides a *projection* of all unique measurements at each sample interval. For example, a 12-electrode ECT sensor (Chapter 1, Eq. 1.1) provides a vector of 66 (M) measurement inputs, which is combined with the bias input. Various interpretations could be estimated in parallel from raw IPT data. For example, a scalar value of solids fraction in a liquid and binary status flags for quality control limits and fault alarms. It may also be useful to encode possible output values, even to the extent of creating an image matrix synthesized using an output node for each estimated pixel. A useful design starting point is therefore to specify ANN output nodes to suit these requirements.

Diagrams involving ANNs where each neuron is depicted quickly become complex, even for small networks. A corresponding compact form, depicting essential design information in *databus* and *process block* form, is shown in Fig. 13.4, and is used in all following sections.

Here data and process flows are from Left to Right. Bias blocks are outlined in Green; Data blocks are outlined in Blue; and Operator blocks are outlined in Red. Names or descriptions of data items in blocks are shown in Blue text. Symbols or values are included at the top and bottom of a block in Black text: the relevant variable at the top; and the data name, or number of data at the bottom. All layers are assumed to be fully connected. Here the 1 bias input and M measurement inputs are combined in a weighted summation followed by the sigmoid activation function with output, a.

Having defined the basic ANN Perceptron, we may consider its operation and limitations. If it is possible to separately classify data presented as a sequence of inputs, Rosenblatt (1962) proved that a Perceptron can identify any specific classification. ANN weights embody interaction data, allowing the network to recognize previously presented datasets, and importantly to generalize when presented with new data, based upon the learning experience. This provides the capability to interpret complex object behavior, for example, electrical tomography *soft field* distortion, which arises from interaction between sensing energy and the object field. As significant effects feature during learning, the ANN will compute a result that embodies this effect. Hence, the ANN is a powerful estimation engine.

For comparison, a classical process estimation usually involves an initial (one-time) *model-driven* selection of a physical model and for a dynamic process, a rolling

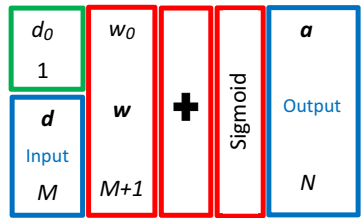

Figure 13.4 Compact representation of single layer ANN.

temporal cycle of measuring process data, followed by a conventional digital computed algorithm to estimate process *output* values. For example, LBP inversion of tomographic measurements to create an image, which may be used to further estimate component fraction values in a mixture.

The ANN method also involves an initial (one-time) *data-driven* stage but does not employ any assumed analytic or statistical model. Instead it exploits the alternate computing paradigm that *learns* process characteristics in a *training* process in terms of the network structure and the set of network bias and weight values, from data sampled from wide ranging actual plant measurements (or from simulations). Rolling estimation in a dynamic process again proceeds in a time cycle of measuring process data, but the trained ANN then directly transforms measurements to component fraction values.

13.2.3 Artificial network training

The foundational Perceptron *back-propagation* learning process attributed to Rumelhart is based upon gradient descent optimization. A major review is beyond the scope of this IPT contextual overview, and for more details, readers are referred to Bishop (1994), and a collection of key works by Anderson and Rosenfeld (1988, 1990). In summary, Rumelhart describes the M weight values at each ANN node (including the bias weight) in terms of an M-dimension hyperspace. For any linearly separable set of input *patterns*, a single-input, single-output layer ANN can extract a corresponding output solution pattern as an $M-1$ dimensional hyperplane. Where the set of input patterns are not linearly separable, Rumelhart found that one or more intermediate *hidden* layers must be used, forming a *multilayer* ANN, to enable a solution.

The availability of adequate training data for the learning process is critical for a successful search for an optimum set of weights and biases for all layers and nodes, in which ANN data are input for feed forward estimation, followed by the back-propagation weight update step. During training, each IPT measurement vector must be accompanied by a corresponding synchronized and independent measurement of the *target* process output, usually measured using temporary supplementary instrumentation fitted to the process plant. An integrated process and sensing model−based simulation may also be used to generate process (measurement with target) data.

One, or several combined, sources may be used to create a *Design Dataset*. A training strategy will typically randomly select three specific datasets from the collected Design Dataset: (i) a *Training Dataset* to be *learned* by the ANN; (ii) a *Test Dataset* to assess error levels during training; and (iii) a *Validation Dataset* of wide-ranging data used to assess if training is complete (and hence avoid *over-training*). The Training set is the only data for which the specific *Training Mode* which features back-propagation is deployed. *Estimation Mode* is the normal application use for a trained network. During training, the Test and Validation datasets are applied to the network in Estimation Mode, but with an added operation that error values are still computed using the included target data. At this stage, network training is incomplete as network structure and weight values are not finalized. For clarity, the Training dataset is the sole *learned* data; the Test and Validation datasets consist of *unlearned* data

but is not used in training. This is important to assess generalization and interpolation capabilities.

It is also common to *preprocess* input data to enhance orthogonality and remove duplication, increasing ANN efficiency to learn and generalize. This is explored in detailed in later sections, particularly via a major example in Section 13.4.3.

To illustrate the back-propagation learning mode, we consider a single-layer ANN having M-inputs vector $\boldsymbol{d}$, and N nodes, with output vector $\boldsymbol{a}$, and corresponding target vector $\boldsymbol{a_t}$. This presents an optimization surface having a singular minimum global Mean Square Error (MSE), E. The error value E_n, at each node, from Eq. (13.1) and resulting expression for E are as follows:

$$E_n = a_{nt} - a_n = a_{nt} - f\left(\sum_{m=1}^{M}(w_{nm}d_{nm})\right) \tag{13.4}$$

$$E = \frac{1}{N}\left\{\sum_{n=1}^{N}(a_{nt} - f(s_n))^2\right\} = \frac{1}{N}\left\{\sum_{n=1}^{N}\left(a_{nt} - f\left(\sum_{m=1}^{M}(w_{nm}d_{nm})\right)\right)^2\right\} \tag{13.5}$$

To seek the minimum MSE point in terms of design weight values, we consider the partial derivative of input pattern error with each weight. Analysis of this derivative (Anderson & Rosenfeld, 1988) for change in error for each training pattern leads to the following expression:

$$\delta_n = -\frac{1}{a_n}\frac{\partial E_d}{\partial w_{nm}} \tag{13.6}$$

where δ represents a weight change coefficient.

Eq. (13.6) indicates that making weight changes proportional to δ will reduce error. Rationalizing from Eq. (13.3) shows that the change in node output error may then be stated as follows:

$$\delta_n = f_n'(s_n)(a_{nt} - a_n) = K\, a_n(1 - a_n)(a_{nt} - a_n) \tag{13.7}$$

This result is used to update the M weight (t) values for each node for the next $(t + 1)$ iteration:

$$w_{nm}(t+1) = w_{nm}(t) + R\,\delta_n\,a_n \tag{13.8}$$

where R is a *Learning Rate* constant, for control of search stability and convergence.

Initialization for training begins by setting all bias threshold weights, w_{0n}, and all input weights, w_{nm} (m:1 … M, n:1 … N) to small random values in the interval [−0.5 … +0.5]. The iteration should then use a randomly ordered sequence of data

patterns from the Training Dataset. Training is normally paused at regular intervals to apply Test Datasets to assess error levels. When errors are acceptable, the Validation Dataset is used to assess when training is complete, to avoid overtraining.

Step 1. Apply input pattern vector, $\boldsymbol{d}$, with bias $d_0 = 1$ and measurements d_1, d_2, … d_M.

Step 2. Compute node output vector $\boldsymbol{a}$, for target vector $\boldsymbol{a_t}$ from Eqs. (13.1) and (13.2).

Step 3. Compute error change vector, $\boldsymbol{\delta}$ from Eq. (13.7).

Step 4. Back-propagate node for each weight: get next vector: $\boldsymbol{w}$ $(t + 1)$ from Eq. (13.8).

Step 5. IF [Test/Validate needed?] THEN [Execute]; ELSE [Return to Step 1].

This simple gradient decent process is used to locate the expected single minimum error point. It is extended to multilayer ANNs by addition of update steps for hidden layers, where back-propagation is executed beginning from the network output. The optimization process for a multilayer ANN is expected to feature local minima and an appropriate search strategy is then required to minimize a global MSE criterion. For brevity, this is not covered in this conceptual introduction; Anderson and Rosenfeld (1988) offer a detailed review. Other training strategies may also provide faster and more efficient design optimization for multilayer networks. For example, an approximation of Newton's method by Levenberg−Marquardt can increase speed but requires large memory. The Momentum method is shown to ignore small local minima in the error surface; related methods also exploit an adaptive Learning Rate.

ANN design and training typically use a computer workstation which features a digital network simulation. In Training Mode, long compute times may be required, but this is a one-time process, and has little relation to the speed of the trained ANN when later used in Estimation Mode, where feedback training features are not required.

IPT instrumentation applications for both pilot plants and operating processes have been outlined in Section 13.1. A design could retain a digital computer implementation if incorporated into an experimental pilot plant configuration, where new ANN configurations can be trained and trialled. In contrast, an operating process configuration could deploy specific purpose-designed hardware to increase speed, reduce instrument costs, and increase plant reliability. Modern ANN design tools (exemplified below in Section 13.6.2) include sophisticated ANN simulation and training facilities. When the design is complete, they include support for the transfer of the ANN structure including all bias and weight values to dedicated Estimation Mode hardware for forming a rolling process estimation instrument.

13.3 Artificial neural networks for IPT applications

To interpret IPT data, we first consider an ANN design to estimate an object image, in effect realizing steps S1, S2, and S3 of Section 13.1 above. As discussed, image information is primarily suited to pilot applications, or in plant diagnostics, where a human

observer has an interest in the current process state. However, it is also useful here for direct comparison with the classical IPT method outlined in Section 13.1. The ANN will examine an input vector of M normalized tomographic measurement values that make up a complete projection, corresponding to a specific time-slice within the dynamics of the process of interest.

13.3.1 ECT example

To illustrate principles, we take the specific IPT example of an Electrical Capacitance Tomography (ECT) sensor having 12-electrodes (Chapter 1, Eq. 1.1). This provides a unique 66-value normalized capacitance measurement vector, $\boldsymbol{c}$, for each tomographic projection of a circular section object space, such as a pipeline transporting liquid/gas. We wish to transform these data to a grayscale image pixel vector, $\boldsymbol{g}$, where pixel positions are assigned using an object space grid, often based upon a Finite Element Model (FEM). We do not require any electro-physical model, or sensitivity distribution. Instead, we simply require training data, sampled from the process; or from a simulation, delineated using the same grid positions.

To estimate an image, we deploy a simple ANN with a single layer for the direct processing of input data to yield the required image output. For simplicity, and recognizing the limited spatial resolution of ECT, we specify an image of 100 pixels. Our Perceptron ANN design thus has 100 nodes, each having a bias input, 66 measurement inputs, and $(1 + 66 = 67)$ weights as shown in Fig. 13.5.

As described above, the realization of ANN performance is critically dependent upon an iterative training process. This ECT example features 100 bias values plus (66×100) measurement input values, a total of 6700 input/bias weight values. Training data will typically include a variety of subtle behavior, for example, including soft field effects.

For later comparison, in terms of processing load during the feed-forward estimation operation, this includes 6700 multiplications; plus 100 summations; and activation computations.

13.3.2 ANN versus linear back-projection

It is interesting to compare the above ANN design with a classical LBP approach, where an electro-physical model is used to generate sensitivity distribution data,

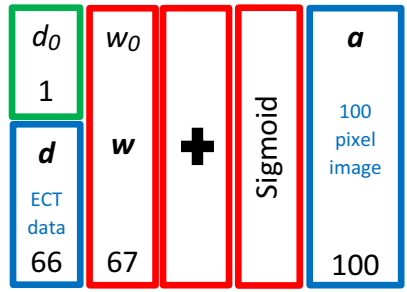

Figure 13.5 ANN for ECT 66 measurement direct 100-pixel image generation.

$s(p)$ which, using the same capacitance measurements described above, enables computation of each positioned grayscale pixel in an image:

$$g(p) = W(p) \sum_{m=1}^{M} c_m \, \mathrm{s}_m(p), \text{ where } W(p) = \left[\sum_{m=1}^{M} s_m(p) \right]^{-1} \tag{13.9}$$

This is a simple process to implement and therefore offers practical advantages of speed, but produces results which are nonlinear, and due to soft field and dynamic range issues, typically includes artifacts and distortion. A postprocessing dynamic thresholding operation is also usually required.

Taking the previous ECT example case data, each image estimate requires 100×66 multiplications (6600); plus 100 scaling additions each of 66 values (6600), plus processing for typical thresholding.

In simple terms, the LBT processing load (without an allowance for threshold processing) is almost double that required to execute the ANN Estimation Mode solution.

To gain speed, both approaches will benefit from parallelized processing hardware. In terms of the result quality, the ANN approach is very likely to out-perform an LBP estimation, as it incorporates learned properties of soft field distortions and thresholding and has faster direct computation speed. Of course, the ML data-driven method requires a major investment in its one-time training.

13.3.3 Multilayer example

For simplicity, we add a further stage to the ANN shown above in Fig. 13.5. The additional hidden layer has inputs from the previous image output layer; and a new final layer now provides an *interpretation* of the image data, for example, to identify the flow configuration. In practice, if an image of the pipeline flow is not required, the interpretation output may be achievable directly by a single-layer ANN with only 66 inputs and outputs for each interpretation state. A design addition to the above network is used to identify which of four common flow states is present:

Stratified flow - Bubble flow - Core flow - Annular flow

Using the simple design rule, an additional output stage will have four nodes, one for each state, each fully connected and having 100 inputs. In this revised model, the image layer now becomes a *hidden* layer to compute the *internal image*. If the revised ANN has twin purposes, it is possible to simply train the added layer using a training set incorporating image configuration data. The revised Multi-Layer Perceptron (MLP) ANN network is shown in Fig. 13.6:

The revised ANN includes the previous 100 (bias) + 100×66 (inputs), totaling 6700 weight values, with the addition of a further 4 (bias) + 4×100 (inputs), totaling 404 weight values; totaling 7104 weight values. It could be deployed for this dual function of providing an image plus a flow regime interpretation. If the only requirement was flow regime identification, we would be likely to find a simpler ANN would suffice, as we next explore.

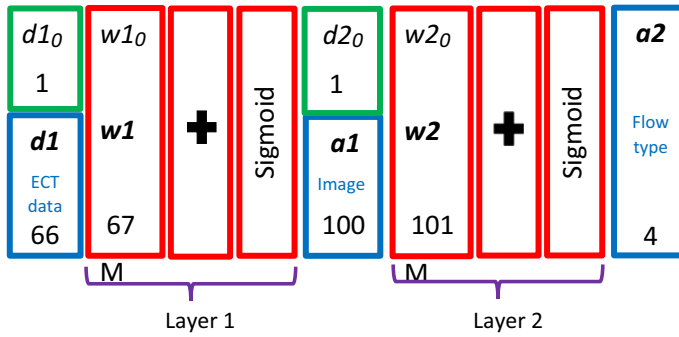

Figure 13.6 ANN for ECT 66 measurement direct image and flow configuration.

Part 3, Chapter 16, also provides a following supplementary application review of AI in process control.

13.4 Case study A—estimating multiphase flow parameters

Multiphase flows arise in several industry sectors, for example in Oil and Gas processing, where volume rate measurements are important. This Case Study addresses an ML approach to the key enabling measurement of component fraction, defined as the area occupied by gas, oil, or water, relative to a total normalized unit area.

Direct flow parameter estimation methods, which are not reliant on reconstructed tomographic images, have been proposed for component fraction and concentration estimation. Early pioneering proposals were reported to be flow regime dependent; more recently proposals by Yu, Yang, Wang, and Yu (2019) are limited to a subset of flow states. Many instruments are limited to two component flows. A direct flow measurement capability that does not require extensive image reconstruction processing would be valuable for instrumentation (Zainal-Mokhtar & Mohamad-Saleh, 2013).

13.4.1 Overview and measurement requirements

IPT ECT sensors have often been deployed for estimating flow parameters within a pipe cross-section based on image reconstructions, as detailed in Chapter 1. Their accuracy is critically affected by the nonuniform sensitivity of the sensing area and the *soft field* effect. Enhancements gained through extensive processing can improve accuracy, but with limitations to certain flow conditions. They are also expensive in processing cost and time further limiting application in dynamic processes such as transport pipelines.

Hence, ANN-based solutions should be flow regime independent and able to directly and speedily transform ECT measurements directly to cross-sectional, component fraction estimates for two-component gas−oil flows, and three-component gas−oil−water

flows. Fig. 13.7 shows the schematic flow regimes investigated, categorized as Stratified, Annular, Bubble, or Homogeneous, where 100% content is assumed be of air, oil, or water.

13.4.2 ECT sensor and simulated ECT measurements

The core sensing unit deployed in this simulation case study is a 12-electrode ECT sensor, as in the introductory example of Section 13.3, for which capacitance measurements were simulated. The simulated pipe has normalized inner, outer pipe, and screen wall radii of 1, 1.2, and 1.4 units, respectively, to facilitate scaled future adaptation. Sensor and guard electrodes have subtended angles of 22 degrees and 2.5 degrees, respectively, resulting in gaps of 8 degrees between pairs of sensor electrodes. Capacitance values between electrode pairs are annotated as c_{ij} (where, i,j: 1 … 12).

For convenience, measurements are normalized with respect to the value when the pipe is filled with the low permittivity material, and scaled with respect to the value range when filled with high and low permittivity material, defined as follows:

$$c_{ijn} = \frac{c_{ij} - c_{ij}\text{low}}{c_{ij}\text{high} - c_{ij}\text{low}} \tag{13.10}$$

The fraction for a component is then calculated simply as follows:

$$f_{\text{component}} = \frac{A_{\text{component}}}{A_{\text{pipe}}} \tag{13.11}$$

where $A_{\text{component}}$ is the area covered by the component of interest (either oil or water), and A_{pipe} is the area of the entire pipe cross-section. Hence, component fraction ranges from 0 (empty pipe) to 1 (pipe full of oil or water).

A two-dimensional FEM ECT simulator by Spink (1996) was used to obtain target flow regime data from electric field lines for corresponding permittivity distributions. Its *decomposed geometry* feature was used to generate sets of 66 ECT projection vectors, $\boldsymbol{c}$ (Mohamad-Saleh & Hoyle, 2002). Design Datasets were generated for all flow patterns of interest for network training and evaluation and included relevant Target parameter values:

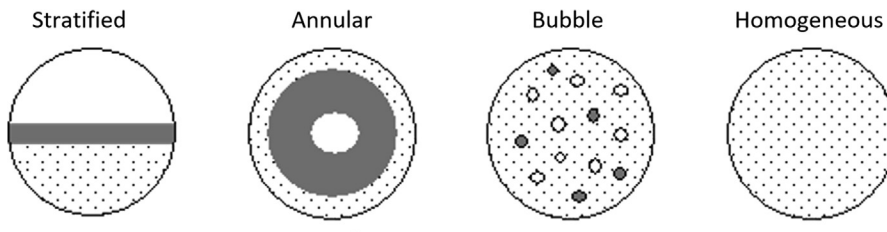

Figure 13.7 Schematic model of the target flow regimes.

Oil−Gas: 3185 flow patterns each with calculated component fractions: 2 simple patterns, 1 empty pipe, 1 full pipe flow, 2520 stratified flows, 636 bubble flows, and 27 annular flows.

Oil−Gas−Water: 4932 flow patterns with calculated oil and water fractions: 3 simple patterns, 1 full gas, 1 full oil and 1 full water, 2730 stratified flows, 2028 bubble flows, and 171 annular flows.

13.4.3 Network design and training for flow estimation

Network design centered upon a computer-based ANN Design Toolsuite, as outlined in Section 13.2.3, based upon the **MATLAB** Neural Network Toolbox (Mohamad-Saleh & Hoyle 2008). Execution speed was not critical as trial data were stored for later evaluation. This supported switching between Training Mode and (simulated) Estimation Mode and was used to design optimum networks for the noted flow regimes for two and three-component flows for parameters: number of hidden layers, all weight and bias values. The aim was to facilitate application deployment, either for a pilot-plant demonstration, or for embedding into purpose-designed hardware forming part of on-line instrumentation for a process monitoring.

The design process used the above combined Dataset of 3185 patterns for Oil/Gas flows and 4932 patterns for Oil/Gas/Water flows. This was randomly divided into 67% Training, 13% Test and 20% Validation datasets. As outlined in Section 13.2.3, Training Datasets form the possible *learned* data used only in Training Mode; Test Datasets were used in simulated Estimation Mode to assess error levels in training; and wide-ranging *Validation* datasets were used in to assess when training was complete.

Preprocessing of input data is commonly used to scale data values and refine data to enhance network efficiency. Here datasets are first preprocessed to normalize values to a convenient range. Next, a further valuable preprocessing operation is used to enhance orthogonality, by removing highly correlated and repeated data. Training Datasets are processed using the Singular Value Decomposition method to realize a Principal Component Analysis (PCA), which ranks orthogonality of data elements. The percentage value of total variation in the transformed Principal Component (PC) dataset is referred to as the PC-Variance. Fig. 13.8 illustrates *Training Mode Preprocessing Operations* for a candidate input Training Dataset.

As illustrated, the M elements of every capacitance measurement vector, $\boldsymbol{c}$, are normalized to produce vector $\boldsymbol{c_n}$, based upon zero mean and unity variance across the Training Dataset. The PCA transformation eliminates measurements which contribute less than a specified fraction (here 5%) of the total Training Dataset variation. The operations produce two outputs: (i) a composite PCA transformation matrix, $\boldsymbol{t}$, generated for the Training Dataset and stored for later corresponding Estimation Mode use; and (ii) the resulting PCA preprocessed Training Dataset, $\boldsymbol{c_{np}}$, of L ($L \leq M$) orthogonal, uncorrelated measurement vectors available for Training Mode operations, in which each vector also includes a time-synchronized target parameter value for error computation.

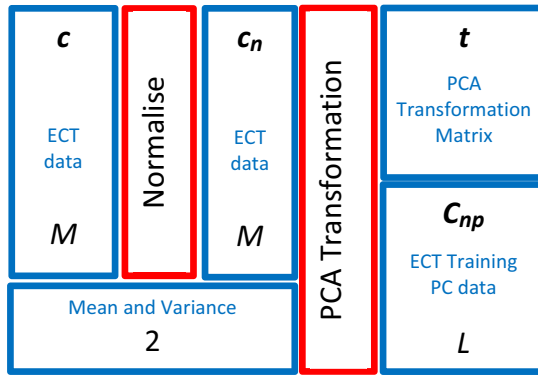

Figure 13.8 Network training mode preprocessing operations.

The Test Dataset is used periodically as training progresses; and the Validation Dataset is used to assess when to cease training. All datasets must embody corresponding preprocessing operations. We also require the same efficiency features to be applied to in-line application networks. Hence, a further preprocessing operation is needed for two roles: (1) to allow the transformation of both Test and Validation Datasets within the network design process and (2) to facilitate preprocessing of in-line process data in pure Estimation Mode when the trained network is fed with process plant ECT vector data. For both preprocessing needs, optimum PCA values have been determined, and for an application network, the design is complete. Hence, this preprocessing operation need only normalize (against preset norms), and then apply the prestored PCA transformation matrix, $\boldsymbol{t}$. The resulting preprocessing operations are illustrated in Fig. 13.9, which has similar nomenclature to Fig. 13.8, but where input vector $\boldsymbol{c_n}$ and preprocessed result vector, $\boldsymbol{c_{np}}$, refer to the specific role, as noted above as (1) and (2).

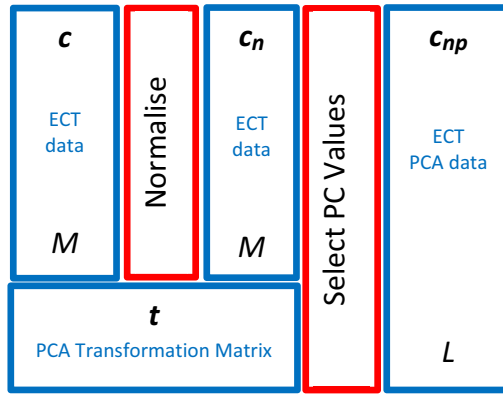

Figure 13.9 Network test/validate and estimation mode preprocessing operations.

For role (1) Training Mode deployment, both preprocessing operations of Figs. 13.8 and 13.9 are single-use operations executed on a one-time basis to transform a complete Dataset. For role (2), Fig. 13.9 operations must be integrated into the front-end of the Estimation Mode ANN design, as illustrated later.

An important ANN design stage is the determination of the number of hidden neurons and the optimum PC variance value in a training process. This can begin firstly with the MLP trained with zero hidden neurons and using the normal practice of initializing weights with small random values, as described in Section 13.2.3.

An *inner iteration* using a fixed PC variance value of preprocessed pattern data is used with a Bayesian Regularization search from the initial weights. Validation patterns are used to assess performance, and if this improves, weights are updated and used in the next training cycle. Training is suspended when there is no improvement. A single hidden neuron having a sigmoid activation function is then added and training is resumed. The iteration of adding a further hidden neuron at each stage continues if performance improves. This *network-growing* process is terminated when there is no further improvement, when the optimum number of hidden neurons is assumed to be attained. The corresponding state information of all current training weights is stored for use in the testing stage. 30 training trials (each with different starting locations on the network error surface) are performed to ensure a global minimum is located.

An *outer iteration* of the network growing process is repeated over a range of PC variance values, using a Mean Absolute Error (MAE) criterion, to locate the global optimum design of PC variance and number of hidden neurons. Here, the MAE criterion is preferred to the MSE criterion of Section 13.2.3.

For this flow estimation application, separate MLP designs are used for each of three component fraction estimation tasks. A linear activation function provides component estimation outputs in the range [0 … 1]. The variants, typically with differing PCA and hidden neuron configurations, must be optimized for their selected flow estimation application role. Hence, the overall system features three subunits each with Estimation Mode Preprocessing Operations and MLP for a specific role:

F1 - Oil fraction from Gas/Oil flows;

F2 - Oil fraction from Gas/Oil/Water flows;

F3 - Water fraction from Gas/Oil/Water flows.

To explore the optimal design of the F1 ANN, measurement data for Gas/Oil Flows MAE results for a range of PC variance values are shown in Fig. 13.10.

This figure presents data for hidden neuron configurations from 2 to 8. A PC variance of 0.05% having the lowest MAE corresponds to an optimum 27 inputs. Use of all 66 measurements (0% PC variance) is shown to produce the highest MAE; likely to arise from correlated components in the full measurement set. This also demonstrates that the network generalizes better when input data are optimal for learning. The figure reveals that the optimum network configuration with five hidden neurons has a test MAE of about 0.32%, which increases when either six or four hidden neurons are used, demonstrating overfitting and underfitting, respectively.

To explore the optimal design of the related F2 and F3 ANNs, measurement data for Gas/Oil/Water Flows MAE results for a range of PC variance values are shown in Figs. 13.11 and 13.12, respectively.

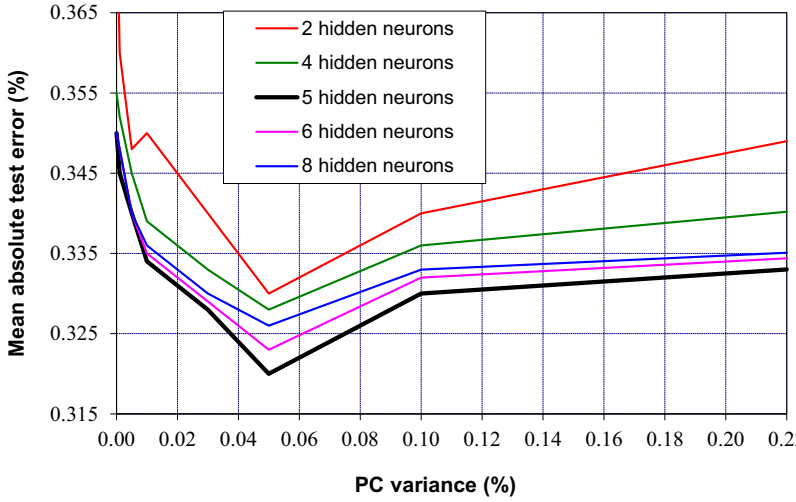

Figure 13.10 Performance with various hidden neurons trained using different PC variance values to estimate oil fractions in Gas/Oil flows.

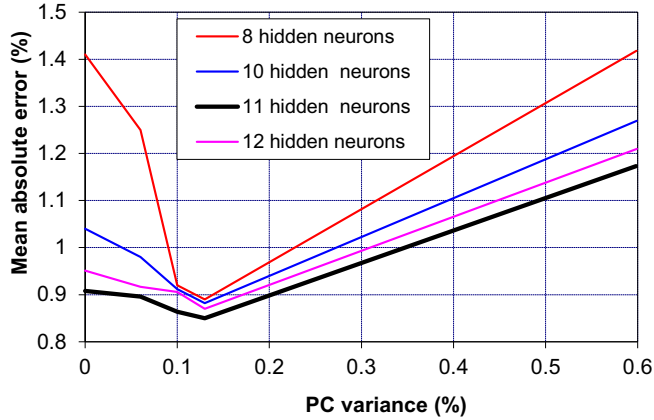

Figure 13.11 Performance with various hidden neurons trained using different PC variance values to estimate oil fractions in Gas/Oil/Water flows.

The F2 and F3 networks are separately trained to estimate oil fraction, and water fraction, from 3-component gas−oil−water flows. Figs. 13.11 and 13.12 show their performance at various PC variance values in the estimation of oil and water fractions from Gas/Oil/Water flows, respectively. In both cases, the least MAE occurs at a PC variance of about 0.13, corresponding to 29 inputs. The F2 oil fraction estimator performs best with 11 hidden neurons, with an MAE of 0.85%. The F3 water fraction estimator performs best with five hidden neurons with MAE of 0.72%. In both cases, MAE increases when overfitting or underfitting occurs.

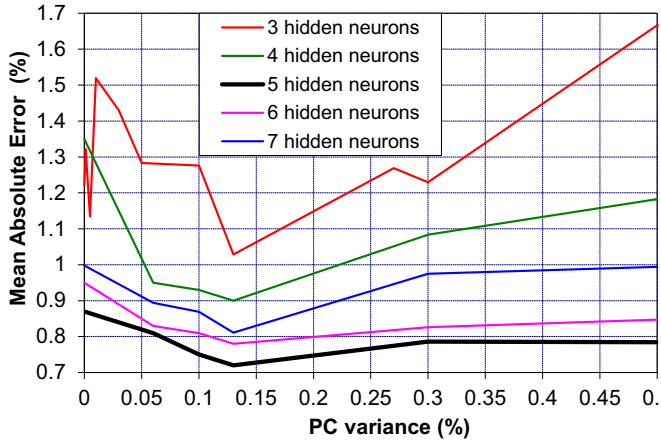

Figure 13.12 Performance with various hidden neurons trained using different PC variance values to estimate water fractions in Gas/Oil/Water flows.

Schematic block diagrams for each of the three optimal ANN designs are illustrated below, including the Estimation Mode Pre-Processing Operations represented here as a single operation in each part of Fig. 13.13.

These clearly indicate the common forms of the networks and the implicit potential to standardize upon a hardware approach to realize instrumentation that fulfills these related application requirements.

13.4.4 Network trials

Trials were carried out on both target flow types: Oil/Gas (F1) and on Oil/Gas/Water (F2 and F3).

Trial performance results for the F1 2-component case, using the ANN of Fig. 13.13a, are classified by flow regimes in Table 13.1.

Results indicate the largest average absolute error occur for annular flow patterns, expected for gas bubbles located in the pipe center, where ECT sensitivity is low. Stratified flows present a simpler estimation of the location of the gas−oil interface, which presents at the ECT sensor boundary where sensitivity is high. The absolute difference in error levels and standard deviations is small, indicating good overall performance. To assess worst case annular data, Fig. 13.14a provides example calibration data of actual and estimated values for 33 annular test patterns, providing a regression fit coefficient of 0.998. These results offer a high confidence level that oil fraction estimation in Oil/Gas flows using the ANN estimation is accurate and not flow regime dependent.

Trials performance results for the F2 3-component case, using the ANN of Fig. 13.13b, are classified by flow regimes in Table 13.2.

Results follow the trends for F1 for the same reasons. Overall errors are slightly higher, but 3-component flows are more challenging. To assess worst case annular

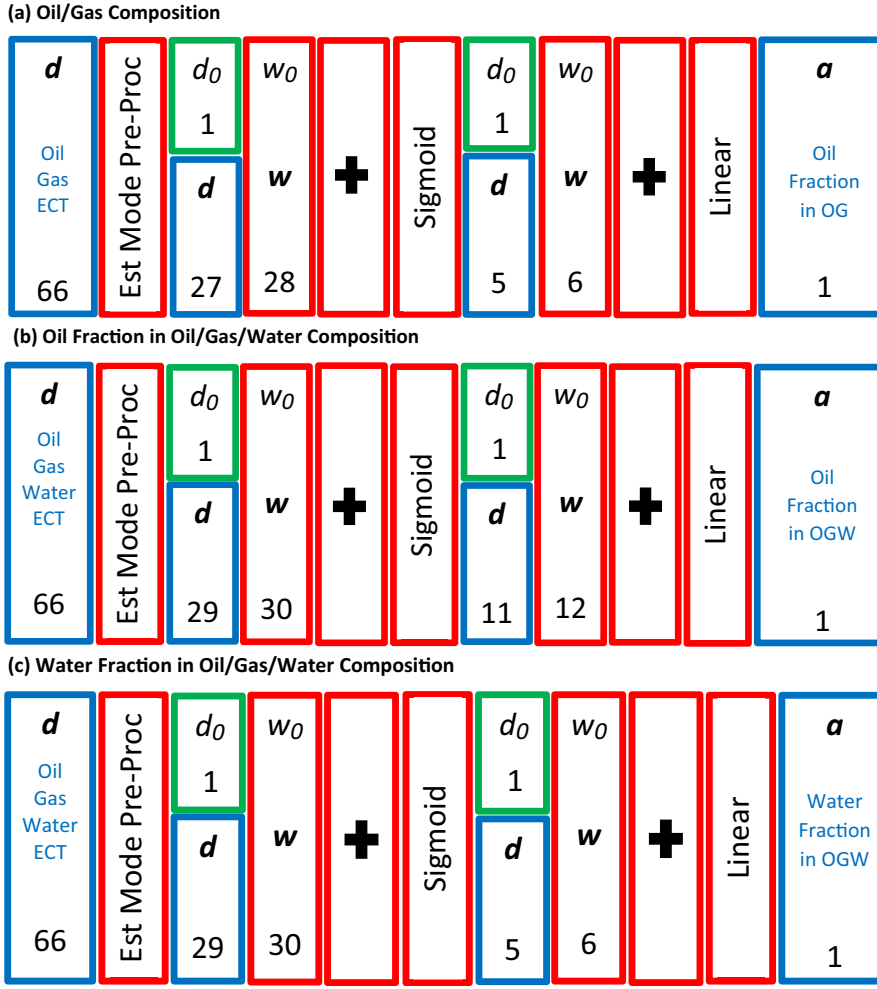

Figure 13.13 MLP ANN - Direct fraction estimations — (A) F1- Oil Fraction in Gas/Oil Composition; (B) F2 - Oil Fraction in Gas/Oil/Water Composition; (C) F3 - Water Fraction in Gas/Oil/Water Composition.

Table 13.1 Oil fraction errors for annular, bubble, and stratified Oil/Gas flows.

	Annular	**Bubble**	**Stratified**
Mean absolute test error (%)	0.20%	0.15%	0.026%
Standard deviation of test error (%)	±0.15%	±0.12%	±0.023%

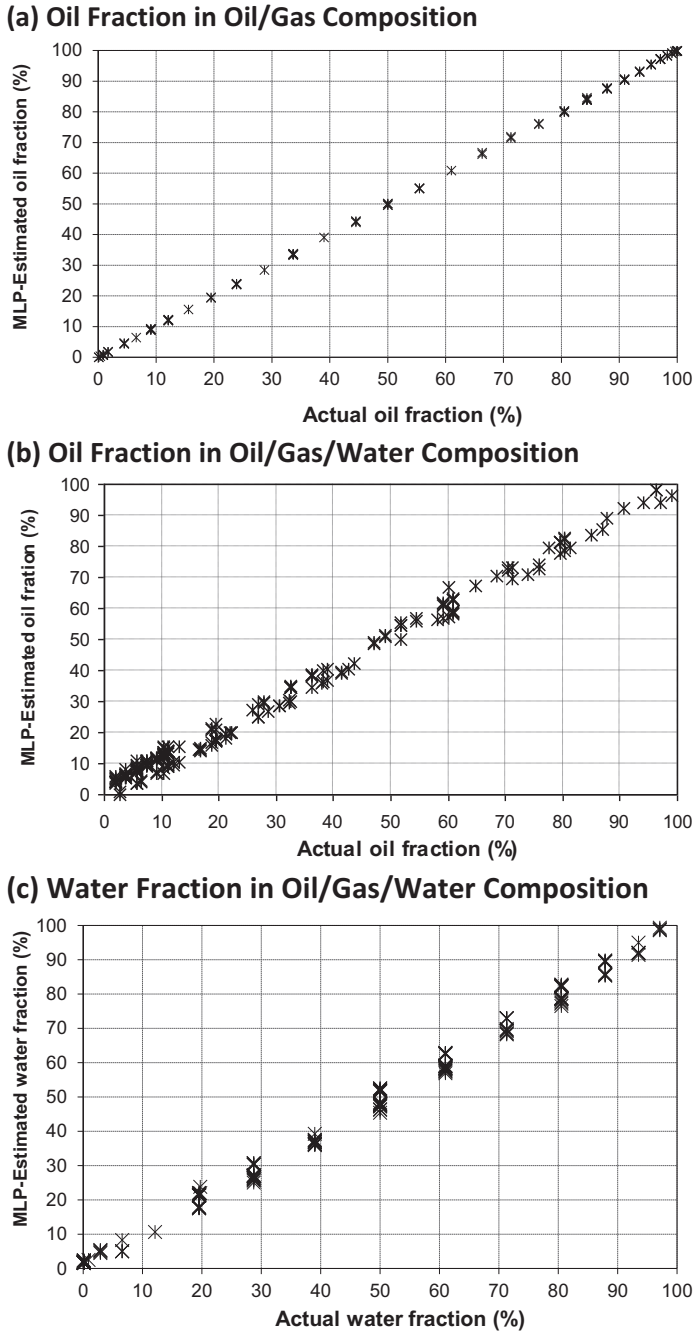

Figure 13.14 Performance calibrations – (A) F1 - oil fraction in gas/oil composition; (B) F2 - oil fraction in Oil/Gas/Water composition; (C) F3 - water fraction in gas/oil/water composition.

Table 13.2 Oil fraction errors for annular, bubble, and stratified regimes of Oil/Gas/Water flows.

	Annular	**Bubble**	**Stratified**
Mean absolute test error (%)	2.3%	0.85%	0.15%
Standard deviation of test error (%)	±0.75%	±0.37%	±0.085%

data, Fig. 13.14b provides example calibration data of actual and estimated values and provide a regression fit coefficient of 0.982. These results offer a high confidence level that oil fraction estimation in Oil/Gas/Water flows using an ANN estimation system is accurate and not flow regime dependent.

Trials performance results for the F3 3-component case, using the ANN of Fig. 13.13c, are classified by flow regimes in Table 13.3.

Trial results again follow the general trends of F1 and F2 results for the same reasons. Overall errors are slightly lower than those for F2. Again, to assess worst case annular data, Fig. 13.14c provides example calibration data of actual and estimated values and provide a regression fit coefficient of 0.980. These results offer a high confidence level that water fraction estimation in OGW flows using an ANN estimation system is accurate and not flow regime dependent.

Further details of these ML flow estimation performance results are available (Mohamad-Saleh & Hoyle 2002).

13.4.5 Evaluation of ANN performance of flow component sensing

This case study has shown that it is feasible to use MLP ANNs to estimate component fractions from 2-component Gas/Oil flows and 3-component Gas/Oil/Water flows. Performance and calibration trial results indicate the success of the method through very low MAE for 2-component, oil−gas flows, with more complex 3-component Gas/Oil/Water flows only slightly increased MAE. The 3-component water fraction estimations have lower error compared to oil fractions. This probably arises as capacitance measurements are calibrated based on materials with highest and lowest permittivity values; in this case, water and gas, respectively. Hence, measurement values are more dominant when the sensing area is filled with water, reinforcing learning associated to water content. However, oil fraction estimation errors are

Table 13.3 Water fraction errors for annular, bubble, and stratified regimes of Oil/Gas/Water flows.

	Annular	**Bubble**	**Stratified**
Mean absolute test error (%)	2.0%	0.7%	0.13%
Standard deviation of test error (%)	±0.61%	±0.31%	±0.1%

slightly higher; they are small in absolute terms offering high confidence that ML estimation is feasible in 3-component flows. Importantly, results indicate that the trained ANNs are robust to different flow regimes and measurements meet the primary aim of flow regime independence.

13.5 Case study B—estimating inline rheology properties of product flow

Rheology, the way a material flows, impacts upon every aspect of our daily lives from the texture of food to the paint on the walls. Not only does rheology affect the final product quality, but it also governs in-process efficiency and is critical in all chemical and physical processing (Barnes, Hutton, &Walters, 2001).

13.5.1 Overview and measurement requirements

The measurement of rheology, termed *rheometry*, is typically conducted off-line through careful sampling of a process stream. Rheological properties from off-line measurements are often considered, with assumptions, as directly applicable to real process flows. However, they can only provide a retrospective characterization of a fluid sample, which may not be representative of the structure as a function of the time-shear history received during processing (Machin, Wei, Greenwood, & Simmons, 2018a,b), and is therefore often considered unsatisfactory.

In situ measurements in a flow environment overcome this deficiency, and also remove the possibility of sample degradation. Due to the critical nature of rheology in processing, the development of an in-line rheometer can advance rheometry from a process end-point, quality control aid to a capability for control, and optimize processes and material structures. Hence, there is an increasing interest in the development of in-line rheometry, as the majority of industrial fluids exhibit complex non-Newtonian behavior.

To address this demand, Machin et al. (2018a,b) developed a novel, in-pipe, tomographic measurement which obtains real-time rheological information of process fluids, termed *Electrical Resistance Rheometry* (ERR). The ability of ERR to estimate rheological parameters is enhanced by the use of ML algorithms and has been demonstrated and validated at an industrial pilot plant of a multinational manufacturer. The trial focused upon the characterization of a wide range of industrial personal and home-care products including shampoos, fabric washes, conditioners, and body washes. Its aim was to compare ERR results against off-line rotational rheometry to determine suitability for in-line quality control.

ERR enables the characterization of fluid rheological properties within a pipe under steady state, incompressible, laminar flow conditions. To achieve such conditions, the Reynolds number of the flow must be less than 2000 (Paul, Atiemo-Obeng, & Kresta, 2004). By cross-correlating fluctuations of computed conductivity pixels across and along a pipe, using noninvasive, microelectrical tomography sensors, rheometric

data are obtained through the direct measurement of velocity profile. The use of microelectrical tomography sensors means that the fluid must be relatively conductive, in the range of 0.5−50 mS/cm. The approach is extremely robust, noninvasive, and able to interrogate process fluids, without limitation due to optical opacity. The wide-ranging applicability and simple implementation of microelectrical tomography sensors to industrial processes is an ideal platform for the development of an in-line rheometer. ERR is derived from Electrical Impedance Tomography (EIT) and is capable of simultaneously sensing mixing performance using the cross-sectional impedance map. Thus, ERR combines two significant engineering quality and process control concepts in a powerful characterization instrument and provides an understanding of the inhomogeneity and nonuniformity of a process.

13.5.2 Sensing and processing system

The ERR instrument used is supplied by Stream Sensing Ltd. and utilizes four sensor arrays of eight electrodes, consisting of two linear and two circular arrays, as shown in Fig. 13.15. This novel configuration provides a complete velocity profile with high sensitivity data near to the pipe-wall boundary provided by the linear arrays, and data from center of the pipe by the circular arrays.

This sensor is linked to a v5r EIT system, supplied by Stream Sensing Ltd., capable of operating a sensor with a maximum of 32 electrodes. Here, the arrays consist of eight electrodes to provide high temporal resolution for velocity precision. The EIT instrument is controlled using v5r software, in Rheology Mode, and to capture a measurement every 30 s. The modified sensitivity Back-Projection algorithm from Wang, Mann, and Dickin (1996) is utilized to reconstruct both linear and circular sensor

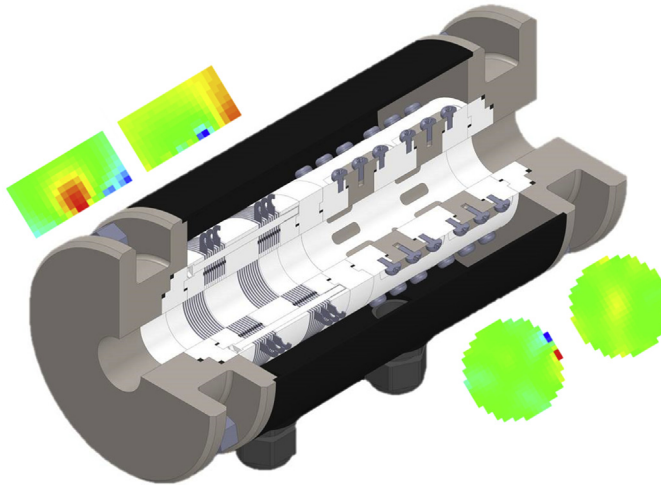

Figure 13.15 Electrical resistance rheometer electrode configuration.
Courtesy of Stream Sensing.

tomograms. These are segmented into zones of interest to assign several radial velocity measurement positions in the pipe cross-section.

A heat pulse is generated within the sensor which causes a minor increase in temperature, in turn producing a conductivity perturbation. This pulse generates uniform heating of the cross-section; is 95% efficient; removes the requirement for a tracer; provides negligible degradation of thermosensitive products; and is automated (Knirsch et al., 2010). Thermal degradation is reduced further as the required temperature rise is only 0.7°C for 0.5 s.

For each zone, the sensed conductivity is averaged and normalized. A cross-correlation algorithm is employed to tag the fluid motion across and within multiple measurement arrays. As the distance between the arrays is fixed and known, the extracted time delay can be converted into two- and three-dimensional velocity profiles. However, due to the axisymmetric nature of velocity profiles in laminar flow, there is only a requirement for a two-dimensional velocity profile in the determination of rheology (Bergman, Lavine, Incropera, & DeWitt, 2019), which reduces computational requirements.

Rheological properties strongly influence velocity profile shape in laminar pipe flow, due to a shear rate response of the fluid. Raw velocity measurements provide a *fingerprint* aid to predict rheological parameters and can be utilized to infer rheological behavior exhibited by complex fluids systems exhibiting shear-banding, wall depletion, and shear-induced phase migration. The velocity profile may be coupled with measurement of differential pressure to obtain the linear shear stress profile in steady state, incompressible laminar flow. If the rheological behavior is known to be described by conventional rheological models, a parametric fitting of the velocity profile can be utilized to directly output the desired rheological properties (Machin et al., 2018a,b). This can be used to extract rheological parameters of fluids modeled by simple, specific constitutive equations; however, as their complexity is increased, parametric fitting becomes increasingly ill-conditioned with large discrepancies. A number of industrial fluids do not comply with such models.

ML methods are preferred in order to provide robust sensing for all materials. To validate this novel approach, several industrial fluids have been investigated at a pilot plant of a multinational manufacturer of homecare and personal care products. The industrial ERR sensor utilized within this study is depicted in Fig. 13.16.

The 25.4 mm Dia. ERR sensor was installed in the simple recirculation flow loop pipeline, shown in Fig. 13.17. This contains a 200 L jacketed vessel, agitated by an anchor impeller and a controlled centrifugal pump. Differential pressure was measured across a 1 m distance using an Endress + Hauser PMD55 meter, with logged flow rate and pressure.

The jacket heater was used to maintain the fluid temperature at 25°C, monitored by a probe at the vessel outlet, before the fluid was circulated at three selected flow rates of 500, 750, and 1000 kg/h. The experiment was repeated at 30°C and again at 35°C. Three ERR measurements were performed for each experimental condition.

A wide range of personal and homecare products were selected for the industrial trial at the pilot scale: shampoos; body washes; fabric washes; and hand washes.

Figure 13.16 Electrical resistance rheometer sensor.
Courtesy of Stream Sensing.

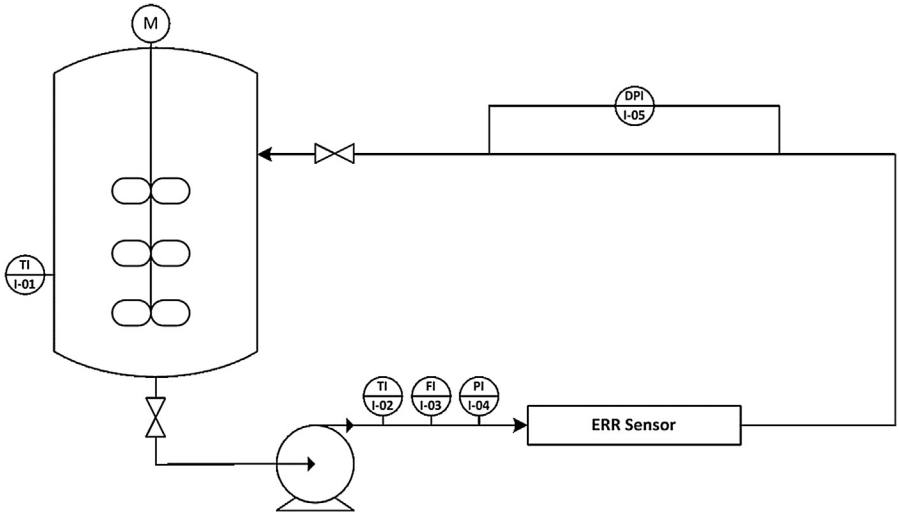

Figure 13.17 Rheology trial flow loop.

To provide comparison of off-line measurements, an HAAKE Rheostress 1 Rotational Rheometer (Thermo Fischer Scientific, USA) equipped with a smooth-walled, 1 degree stainless steel cone and plate geometry was used. A Peltier plate was utilized to maintain each sample at $25 \pm 0.1°C$, $30 \pm 0.1°C$, and $35 \pm 0.1°C$. To assess the impact on structure from the time-shear history during processing, samples were collected, and off-line measurements performed, at each experimental flow rate. To minimize such effects further, a new sample was loaded onto the rheometer for each experimental condition. Upon loading, a logarithmic ramp was applied with a shear rate range of $0.1-1000\ s^{-1}$, with five points per decade over 5 min. The resulting data were analyzed using a nonlinear least squares regression, using the HAAKE

RheoWin software, to extract rheological parameters. This analysis was performed across the shear rate range of 0.1−100 s^{-1}, reflective of the subsequent flow experiments. Two specific shear rates of 4.83 and 20.7 s^{-1} were then selected for comparison with viscosities obtained from both in-line and off-line measurements.

13.5.3 Machine learning approach

A supervised two-layer ANN was selected to estimate the viscosity at the noted shear rates. Inputs include 34 velocity data points from the ERR sensor, plus the differential pressure across the sensor. Input values are acquired over approximately 30 s to ensure that dynamic changes in the process are captured. A sigmoidal transfer function was employed for the 30-neuron hidden layer. The output layer is a single neuron with a linear transfer function to provide an estimated viscosity output. The ANN is shown in Fig. 13.18.

For each shear rate, an independent model was applied with target data and viscosity obtained from off-line rotational rheometry. Considering all test material types, viscosity varied by up to three orders of magnitude. To ensure measurement sensitivity for small viscosity values, two models were trained for each shear rate. Due to the importance of rheology measurements in processing, there is a requirement to obtain a high level of accuracy and thus Bayesian Regularization was selected as the training algorithm. Training was performed using the MATLAB ML toolbox, with the data split into 75% training and 25% test groups to verify the performance.

13.5.4 Network trials

Rheological properties of a fluid strongly influence the velocity profile shape in laminar pipe flow, due to a shear rate fluid response, as highlighted in Fig. 13.20 below. The displayed velocity profiles were obtained using the method outlined by Machin et al. (2018a,b), utilizing a parametric fit of ERR data to a power law rheological model. It is evident that velocity profiles of shampoo, Fig. 13.19d, feature increased blunting, compared to the hand dish wash, Fig. 13.19b, due to increase in shear thinning behavior. This is also reflected in the rheology data obtained from off-line measurements, in Fig. 13.19a and c.

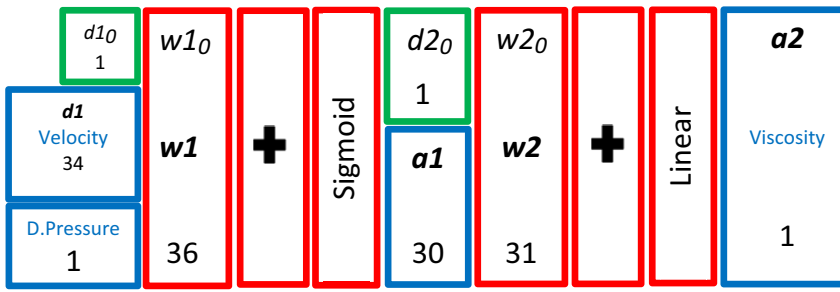

Figure 13.18 Two-layer feed forward ANN to estimate rheological properties.

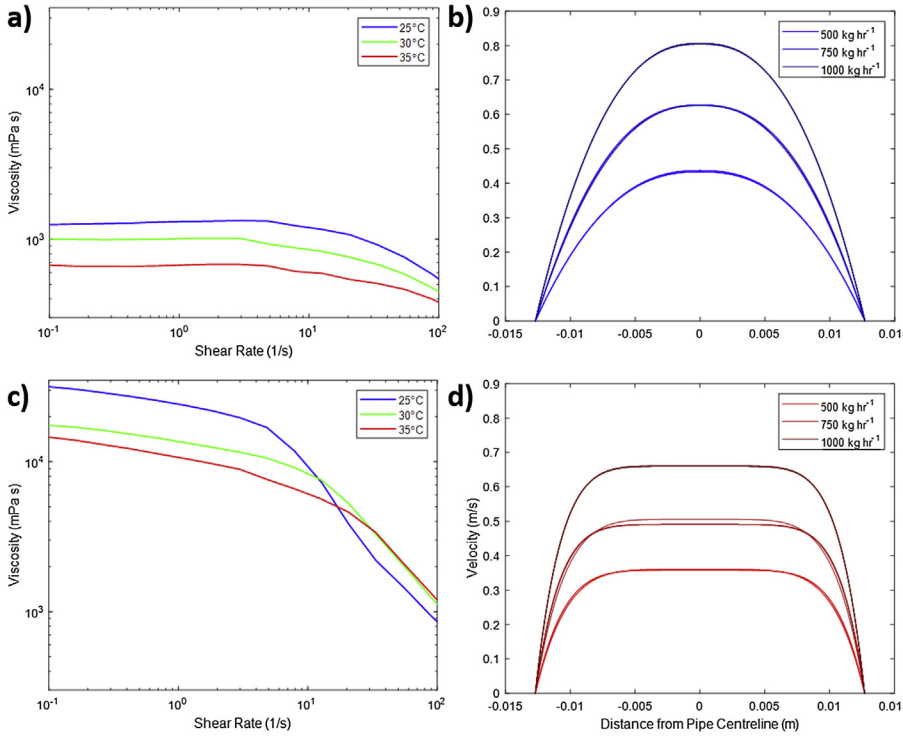

Figure 13.19 Experimental data: (A) hand dish wash − rheology; (B) hand dish wash − ERR velocity profiles; (C) shampoo − rheology; (D) shampoo − ERR velocity profiles.

It is acknowledged that the parametric fitting approach is not ideal, as selected fluids do not adhere to the power law rheological model. Therefore, to capture the increased complexity of the fluid structure, an ML algorithm was developed for in-line quality control. The estimation of in-line viscosity was seen to correspond to that of the off-line rheometer, with results outlined in Fig. 13.20. When considering low viscosity fluids, at 4 s^{-1}, the correlation coefficients between the measured and estimated viscosity were 0.999, 0.998, and 0.999 for the training, test, and complete datasets, respectively. High correlation is also found in the corresponding root-mean squared error (RMSE) of 2.50, 30.6, and 11.87 mPa s. This ensures that the application of an ANN to ERR provides a highly accurate estimation of in-line viscosity.

The high performance has been attained using a relatively small dataset and is likely to be improved further, with the performance of algorithms trained using Bayesian Regularization enhanced monotonically with increasing size of the dataset, as discussed by Neal (1996). The ability of the high viscosity model to predict viscosity was also seen to have high correlation with correlation coefficients of 0.999, 0.997, and 0.999 obtained for the training, test, and complete dataset, respectively.

To estimate rheological behavior, the viscosity of the system must be estimated at multiple shear rates and hence a second shear rate of 20.7 s^{-1} was selected. As the

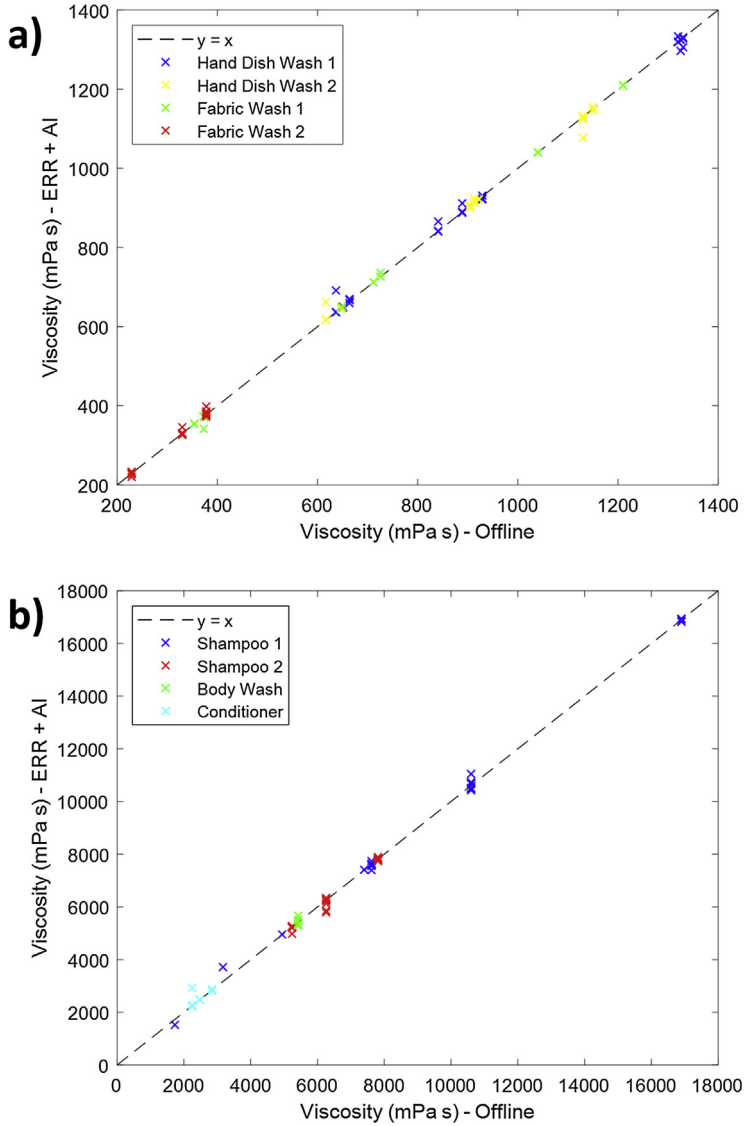

Figure 13.20 ERR viscosity estimates at 4 s^{-1}: (A) low viscosity fluid model; (B) high viscosity fluid model.

shear rate was increased, a minor decrease in the viscosity estimate error was observed. The RMSE for the low viscosity fluid model reduced from 30.6 to 21.8 mPa s; the test data for all conditions are displayed in Fig. 13.21. Similarly, the RMSE in the high viscosity model decreased by 59% with ERR able to accurately estimate the viscosity at multiple shear rates and ultimately rheological properties of a wide range complex industrial fluids. The RMSE error obtained for all test data conditions is below the desired error for an in-line rheology measurement ensuring high in-plant applicability.

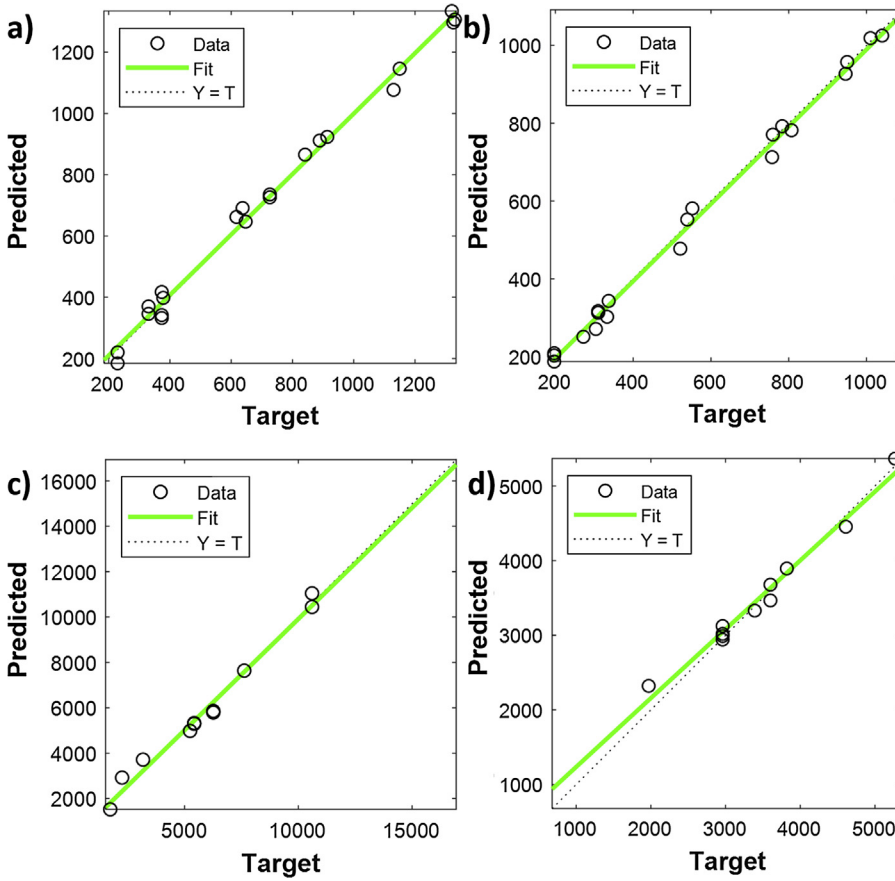

Figure 13.21 Test data: (A) low viscosity fluid model − 4 s^{-1}, (B) high viscosity fluid model − 4 s^{-1}, (C) low viscosity fluid model − 21 s^{-1}, (D) high viscosity fluid model − 21 s^{-1}.

Although not demonstrated in this study, if the fluid adheres to a specific constitutive equation, AI may be employed to estimate parameters of the model and determine the rheological behavior across the entire shear rate ranges of Fig. 13.20. The use of an ANN to estimate parameters of complex rheological models reduces the extent of ill-conditioning when compared to the parametric fitting approach.

13.5.5 Evaluation of ML performance for rheology sensing

This study has demonstrated that ML algorithms can be successfully applied to ERR measurements to accurately estimate in-line rheological properties across a range of industrial personal and home care products. AI methods are also able extend the capability of ERR to capture complex fluid behaviors often observed in industrial fluids, such as shear-banding and wall depletion, greatly enhancing in-plant applicability.

The application of AI algorithms to ERR transforms rheometry from a critical, off-line control tool, to a powerful on-line capability to support control and optimization of processes.

13.6 Future trends

This chapter has introduced ML from first principles for applications to IPT, in both pilot scale and embedded instrumentation; and illustrated this with two contrasting, detailed Case Studies providing detailed insights into design. They demonstrate that sampled experiential process data can be transformed into effective and accurate ANN designs, which are then able to transform IPT measurement data directly into process information.

It is interesting to project potential contributions of ML-IPT to industrial development for a cleaner, greener, and more efficient and productive world. Industrial strategies such as Industry 4.0 (Schwab, 2017) seek to exploit major in-depth information, from individual processes to complete manufacturing plants, to enhance performance, impacting business, economic, and environmental goals. ML methods hold great promise in the direct conversion of spatial measurement data to meaningful information, based upon prior deep knowledge of the process. This requires compatible in-line sensors and diagnostic methods for high response process analytics. Tomographic measurements can interrogate major internal process information. Accurate timely and smart information, typically in digital form, such as that available from IPT, is an essential prerequisite to the success of Industry 4.0.

It may be said that IPT is currently hampered in wider application, by an apparent *manual need* to find models and *tune* them into a design approach for any new candidate process. As demonstrated above this technically demanding and time-consuming process can be largely replaced by a simpler exercise of sampling and training data selection. In Section 13.6.1 next section, we overview the future reality of also removing much of this process through new trends in deep learning.

Tools for investigation and design continue to develop to further enable the lifecycle process of instrumentation design from concept to products. Section 13.6.2 reviews leading products and systems through to the deep learning stage noted above.

Final conclusions are noted in Section 13.6.3.

13.6.1 Deep learning futures for ANNs for IPT

A key theme of this chapter is data-driven design, represented by the essential training requirements for many ML system architectures, particularly the supervised forms noted in Fig. 13.1. An essential element is the selection and formation of training data. Section 13.2.3 introduced the possibility of preprocessing training data for efficiency in representing an optimal variety of the process experience. Section 13.4.3 has provided a detailed exemplar in preprocessing ECT measurement data for ANN Training Mode to select, and then in Estimation Mode deploy, optimum

orthogonalized and uncorrelated data. This avoids overtraining which is both inefficient and importantly reduces generalization capabilities. In these examples ANN Training Mode is presented as a pseudomanual design process, where evidence is obtained for network design decisions.

More recent deep learning methods take further lessons from biological systems, in their ability to prestructure complex information, inspired initially from vision processing. For example, we learn to recognize familiar shapes even when they are oriented differently (or *shifted*). These powerful capabilities are modeled in the Convolutional Neural Network (CNN), which like a conventional MLP has input, output, and multiple hidden layers. Its structure was originally proposed by Fukushima (1980) for pattern recognition. Its capabilities are provided by convolutional layers that convolve multiple data with a multiplicative, or other dot product operation, to reveal hidden spatial patterns, such as repeated patterns from nearby measurements, identified by intrinsic shift and compare operations. The activation function is commonly a Rectifier-Linear unit which features a dead-band that responds linearly from a significant feature detection level. Additional convolutions such as data pooling layers, which share selected convolution results; fully connected layers; and normalization layers, are all referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

The shift and compare operations can also be temporal, realized through a further CNN variant, the Time Delay Neural Network, which provides temporal shift invariance identification. This variant was first proposed by Waibel et al. (1989, p. pp328) and has recently been deployed by Johansen, Østby, Dupré, and Mylvaganam (2018) for flow regime estimation from ECT data.

In general terms, a composite raw data training set may be considered as a tensor that can be operated upon jointly by a set of spatial and/or temporal convolution operations, to automatically reveal repeated or correlated data, and to then encode it in compressed form for processing by later layers. We see *deep learning* commonly applied in many application areas where copious data exist. IPT offers raw data. Clear steps and research are needed to join-up these developments, so that, perhaps based upon temporary supplementary instrumentation, deep learning strategies can automate ANN learning in industrial processes delivering fast application realization.

13.6.2 Practical future steps forward in machine learning for IPT

A systematic approach to IPT instrumentation must consider both design and implementation. In principle, a classical approach is *model driven*, whereas the neural approach is *data driven*. Although any project can usefully utilize known process information, a neural computing approach must primarily consider from the outset if, and how, adequate operational data for training can be obtained. Where a classical design will focus upon the system hardware and software needs, a neural design will focus on ANN architecture and its training, with major process investigation and investment to acquire training data. Based upon wider examples, this one-time investment in process data is worthwhile in a major enhancement of performance.

There are currently several tools for ANN design, which typically include ANN simulation and training support useable on any ubiquitous design workstation. All major computing systems manufacturers offer major support to link application design to fast highly parallel devices and subsystems that can be integrated into process instrumentation design. This technology area is fast changing.

Here we offer a brief summary of current provision from supplier through five examples.

Major global microsystem manufacturer ARM (2021) currently offers CMSIS-NN, a library of efficient kernels able to maximize performance and minimize memory requirements for ANN applications using ARM Cortex-M processors. A trained prototype can be optimized using ARM tools for compatibility with the CMSIS-NN library. High-level functions from the kernels can then be simply integrated into high-level application code.

As deployed in case studies in this chapter, MathWorks (2021) offer a range of mathematical tools which include specific support for ML and may be suited especially for early pilot scale investigations. Automatic machine learning includes feature selection, model selection, and hyperparameter tuning. Automated generation of code for embedded and high-performance applications is provided. Integration with Simulink as native or MATLAB Function blocks and supports embedded deployment or simulations and popular classification, regression, and clustering algorithms for supervised and unsupervised learning with faster execution than open source on most ML computations.

Multichannel Graphics Processing Unit microsystem manufacturer NVIDIA (2021) is the world leading supplier of products used for high-performance graphics applications. Due to their featured major parallelization, these devices have been deployed for ANN applications. The TensorRT software development toolset for application development may be used to optimize ANN models trained in all major frameworks and also supports deployment to embedded hardware products such as process instrumentation.

Systems solutions provider NXP (2021) offers its eIQ ML development environment for NXP MCUs, i.MX RT crossover processors, and i.MX family *System on Chip* (SoC) devices. The environment also supports ARM CMSIS-NN and ARM NN SDK, TensorFlow Lite, and OpenCV inference engines.

Major configurable systems provider Xilinx (2021) offers its Deep Neural Network Development Kit (DNNDK) for porting ANN configurations to Xilinx devices. The Deep Learning Processing Unit (DPU) is a configurable computation engine optimized for CNNs (see Section 13.6). The degree of parallelism utilized in the engine is a design parameter and can be selected according to the target device and application. It includes a set of highly optimized instructions and supports most CNNs. The DPU, a hardware platform for Xilinx FPGAs, is scalable for use on to fit various Xilinx Zynq-7000 and Zynq UltraScale+ MPSoC devices. The DNNDK accepts models from Caffe and/or Tensorflow and maps them into DPU instructions. The DPU has a runtime engine supporting generic software and APIs to facilitate efficient deployment and testing of ANNs.

13.6.3 Predicting the future of ML in process sensing

This chapter has presented encouraging steps that promise to liberate the exploitation of IPT technology which has been severely limited by the current need to follow a complex path of modeling, trial, and individual interpretation design in which every application must be designed as an expensive and time consuming special case.

We know that the optical data presented to the retina of a human eye are a blur of color and intensity. The human brain learns the massively complex capability we call *sight* to make sense of these data to create powerful and complex visual information purely by *practicing*.

We can expect deep learning CNN-based interpretation to directly learn to design and train ANN structures to solve the much simpler task of processing raw IPT measurement data, to gain major high-value, multidimensional process information to realize many facets of Industry 4.0.

ML is waiting to supercharge IPT!

References

Anderson, J. A., & Rosenfeld, E. (Eds.). (1988). *Neurocomputing: Foundations of research* (Vol. 1). MIT Press, ISBN 978 0 26251 0486.

Anderson, J. A., & Rosenfeld, E. (Eds.). (1990). *Neurocomputing: Foundations of research* (Vol. 2). MIT Press, ISBN 978 0 26251 0752.

ARM. (2021). https://www.arm.com/solutions/artificial-intelligence/machine-learning. (Accessed 29 January 2021).

Barnes, H. A., Hutton, J. F., & Walters, K. (2001). *An introduction to rheology* (3rd ed.). Amsterdam: Elsevier, ISBN 978 1 49330 2611.

Bergman, T. L., Lavine, A. S., Incropera, F. P., & DeWitt, D. P. (2019). *Fundamentals of heat and mass transfer* (8th ed.). New York: Wiley, ISBN 978 1 119-53734 2.

Bishop, C. (1994). Neural networks and their applications. *Review of Scientific Instruments, 65*(6), 1803−1831. https://doi.org/10.1063/1.1144830

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics, 36*(4), 193−202. https://doi.org/10.1007/BF00344251

Johansen, R., Østby, T. G., Dupré, A., & Mylvaganam, S. (2018). Long short-term memory neural networks for flow regime identification using ECT. In *ISIPT 9th world congress in industrial process tomography, Bath UK* (pp. 135−142), ISBN 978 0 85316 3497.

Knirsch, M., dos Santos, C., de Oliveira Soares Vicente, A., & Vessoni Penna, C. (2010). Ohmic heating, a review. *Trends Food Science, 21*(9), 436−441.

Machin, T., Wei, K., Greenwood, R. W., & Simmons, M. J. H. (2018a). Electrical resistance rheometry − the application of multi-scale tomography sensors to provide in-pipe rheology in complex processes. In *ISIPT world congress in IPT-9, Bath UK* (pp. 143−154), ISBN 978 0 85316 3497.

Machin, T. D., Wei, K., Greenwood, R. W., & Simmons, M. J. H. (2018b). In-pipe rheology and mixing characterisation using electrical resistance sensing. *Chemical Engineering Science, 187*, 327−341. https://doi.org/10.1016/j.ces.2018.05.017

MathWorks. (2021). https://www.mathworks.com/solutions/machine-learning.html. (Accessed 29 January 2021).

Mohamad-Saleh, J., & Hoyle, B. S. (2002). Determination of multi-component flow process parameters based on electrical capacitance tomography data using artificial neural networks. *Measurement Science and Technology, 13*(12), 1815−1821. https://doi.org/10.1088/0957-0233/13/12/303

Mohamad-Saleh, J., & Hoyle, B. S. (2008). Improved neural network performance using principal component analysis on matlab. *International Journal of Computer Integrated Manufacturing, 16*(2), 1−8. ISSN 0858-7027.

Neal, R. (1996). *Bayesian learning for neural networks*. New York: Springer, ISBN 978 0 387 94724 2.

Nooralahiyan, A. Y., Hoyle, B. S., & Bailey, N. J. (1993). Application of a neural network in image reconstruction for capacitance tomography. In *Proc. European concerted action on process tomography* (pp. 50−53).

NVIDIA. (2021). https://www.nvidia.com/en-us/ai-data-science. (Accessed 29 January 2021).

NXP. (2021). https://www.nxp.com/applications/enabling-technologies/ai-and-machine-learning:MACHINE-LEARNING. (Accessed 29 January 2021).

Paul, T. D., Atiemo-Obeng, V., & Kresta, S. (2004). *Handbook of industrial mixing: Science and practice*. New York: Wiley-Interscience, ISBN 978 0 47126 9199.

Rosenblatt, F. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. Washington DC: Spartan Books, ISBN 978 3 642 70913 5.

Schwab, K. (2017). *The fourth industrial revolution*. New York: Crown Publishing Group, ISBN 978 1 52475 8875.

Spink, D. M. (1996). Direct finite element solution for the capacitance, conductance or inductance, and force in linear electrostatic and magnetostatic problems. *International Journal for Computation and Mathematics in Electrical and Electronic Engineering, 15*(3), 70−84.

Waibel, A., et al. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, & Signal Processing, 37*(3), 328−339.

Wang, M., Mann, R., & Dickin, F. J. (1996). A large scale tomographic sensing system for mixing processes. In *Proc. third international workshop on image and signal processing* (pp. 647−650), ISBN 978 0 44482 5872.

XILINX. (2021). https://www.xilinx.com/applications/megatrends/machine-learning.html. (Accessed 29 January 2021).

Yu, H., Yang, G., Wang, Y., & Yu, C. (2019). Design of capacitive sensor for phase concentration measurement in two-phase flow. In *31st Chinese control and decision conference (2019 CCDC), Nanchang, China, China* (pp. 3221−3226).

Zainal Mokhtar, K., & Mohamad-Saleh, J. (2013). An oil fraction neural sensor developed using electrical capacitance tomography sensor data. *Sensors, 13*(9), 11385−11406.

Further reading

Lähivaara, T., Kärkkäinen, L., Huttunen, J. M. J., & Hesthaven, S. (2018). Estimation of porous material parameters using ultrasound tomography and deep learning. In *ISIPT 9th world congress in industrial process tomography, Bath UK* (pp. 283−288), ISBN 978 0 85316 3497.

Wang, J., Liang, J., Cheng, J., Guo, Y., & Zeng, L. (2020). Deep learning based image reconstruction algorithm for limited-angle translational computed tomography. *PLoS One, 15*(1), 1−20. e0226963.

Wang, G., Zhang, Y., Ye, X., & Mou, X. (2019). *Machine learning for tomographic imaging*. IOP Publishing, ISBN 978 0 7503 2214 0.

Yan, R., & Mylvaganma, S. (2018). Flow regime identification with single plane ECT using deep learning. In *ISIPT 9th world congress in industrial process tomography, Bath UK* (pp. 289−298), ISBN 978 0 85316 3497.

Zheng, J., & Peng, L. (2018). Deep learning based image reconstruction for electrical capacitance tomography. In *ISIPT 9th world congress in industrial process tomography, Bath UK* (pp. 337−346), ISBN 978 0 85316 3497.