

## SMART ARCHIVE FOR ON-LINE LEARNING SYSTEMS

Kauko Väinämö, Juha Röning, Perttu Laurinen  
Machine Vision and Media Processing Group, Infotech Oulu,  
Department of Electrical Engineering, University of Oulu, FIN-90570 Oulu, Finland  
email: [verneri@ee.oulu.fi](mailto:verneri@ee.oulu.fi), [jjr@ee.oulu.fi](mailto:jjr@ee.oulu.fi), [perttu@ee.oulu.fi](mailto:perttu@ee.oulu.fi)

### ABSTRACT

*The amount of information grows at an exponential speed, posing increasing demands for the systems processing that information. The focus in future intelligent systems will be to find and store the information significant for the overall system. This involves data warehousing or data mining techniques as well as machine learning. The meaningful information affects the ability of systems to learn and adapt to the ever-changing reality.*

*In this paper, we present a concept of a smart archive, to be used as a memory and information processor for on-line learning systems. The overall concept includes three logical parts: a smart archive, a static classifier (static model) and a neural network structure (dynamic model). The smart archive consists of four sub-objects: a data pre-processor, a smart data storage, a training set generator and a control object. The smart archive involves on-line and off-line algorithms. The on-line algorithms are used for filtering, data validation, data annotation and the creation and maintenance of a history buffer. The off-line algorithms are used for sampling control, filtering control and network structure re-creation and training.*

*The smart archive is presented in the case of an industrial classifier where visual inspection is used for production quality control. A smart application on a production line must be robust, accurate and reliable, it must be able to cope with changing situations and it must have a short start-up and installation period. The approach proposed meets these demands; the presented system has an ability to filter outliers from the measurement data, and it has a short start-up time, because a static model can be used from the beginning. The dynamic model learns gradually and improves the accuracy of the system. The concept is efficient because real-time algorithms are used and memory consumption is minimised by using smart archiving algorithms.*

**Keywords:** smart archive, artificial neural networks, on-line learning, machine learning

### 1 INTRODUCTION

Today's information society generates huge amounts of information that need to be processed, not to mention industrial production systems. The amount of information poses challenging requirements for information models or systems, which must also be able to cope with changing situations. The key point is to find the relevant information from the data and to realise that relevance is quite often a function of time. This involves data mining and data warehousing

techniques as well as machine learning. In that way the meaningful information affects the ways in which the system learns and adapts to the ever-changing reality.

Smart applications learn from data samples or derive knowledge using specific information obtained from experts. However, the problem in many cases is that the intelligent model or application is static in nature. The approach often adopted to cope with a changing situation is to use adaptive methods to adapt to changes in the measurement signal trends, for example. Such methods are very usable in non-stationary situations. However, the memory they include is quite limited; the system adapts itself to the data based on the recent history, but is not able to recognise a rapidly changing situation or to recall information learned in the remote past. Another, and the most important, lack of functionality is the inability to recognise novel input patterns.

Problems of this kind are typical in industrial classifier systems. The product materials change, and it is impossible to create a static classifier model capable of coping with the changing situations. And even when a static classifier model is created, it will produce erroneous results after some time has elapsed.

In this paper, we present a concept of a *smart archive* (SA), which is an essential component of on-line learning systems. The overall system consists of a static classifier model, a trainable neural network structure and a smart archive. The concept of smart archive is a solution that enables the on-line learning system to archive very relevant information in order to respond to changing situations. The present solution is usable in industrial applications. The key issues include highly automated data processing, robustness, effectiveness, reliability and a short start-up time.

The reason why there is both a static classifier model and a dynamic neural network structure is to ensure the reliability of the system even at the beginning of overall system usage. In many cases there is neither enough time nor other possibilities to record enough information to create a basic dynamic model at start-up. This problem is eliminated by using a static classifier model, which gives results that are reliable, though not optimal in accuracy at the beginning, but which cannot cope with samples not seen earlier and not included in the classes of the classifier. When the system is used and more history information is gathered, the dynamic model will be trained to give better accuracy for the overall classification system.

The role of the SA is to be a memory, a data pre-processor and a teacher. The smart archive includes the following methods and algorithms:

- Data mining and filtering methods for the preparation of archived data
- Storage minimisation; removal of redundant information
- Temporal information weighting for re-training of the on-line learning system
- Algorithms to detect new situations or concepts of input variables
- Control of on-line learning system re-training

The SA involves mining data for information. Traditionally, the data mining process is a manual operation and very time-consuming, as especially data preparation is very hard to automate. However, in order to recognise new input classes, for example, automatic data mining algorithms

must be used. Pyle has presented automated methods for data preparation. These automated techniques can cut the preparation time by up to 90%, depending on the quality of the original data [Pyle, 1999]. Such automation is essential for on-line learning systems, because it is not possible to have human interaction for every instance of re-training.

This paper presents a framework for an industrial intelligent system, which includes several known techniques for information processing. The focus is on the smart archive concept. The paper presents an idea of how the smart archive concept can be used in the framework of an intelligent system. The concept is presented using an industrial classifier as a case. A quality control system based on the machine vision technology has a critical component, the classifier.

The paper is organised as follows: the nature of on-line learning systems is described in Chapter 2. It includes basic theory and references to relevant recent research. The principle of the smart archive method is described in Chapter 3. The main contribution of the paper is in that chapter. Chapter 4 describes a case where the Smart Archive concept will be used, and the conclusions are presented in Chapter 5.

## 2 NATURE OF ON-LINE LEARNING SYSTEMS

The present concept includes a dynamic model, which has a capability to learn from samples and to adapt to the operational environment. To achieve such goals, the selection of the model technique is critical. There are issues in machine learning which should be considered carefully in order to have successful application implementation [Mitchell, 1997]:

- What algorithms are available for learning general target functions from specific training examples?
- How much training data is sufficient?
- When and how can prior knowledge held by the learner guide the process of generalising from examples?
- What is the best strategy for choosing a useful next training experience, and how does the choice of this strategy alter the complexity of the learning problem?
- What is the best way to reduce the learning task to one or more function approximation problems?
- How can the learner automatically alter its representation to improve its ability to represent and learn the target function?

When considering these key issues in the smart archive concept, it is clearly easy to choose a neural computing technique for the dynamic model presented here. According to Haykin, a neural network can be viewed as an adaptive machine [Haykin 1994, Aleksander & Norton 1990]

*A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:*

1. *Knowledge is acquired by the network through a learning process.*
2. *Interneuron connection strengths known as synaptic weights are used to store the knowledge*

Neural network techniques constitute a robust machine learning approach to complex real-world problems that involve huge amounts of information. That claim and a *back-propagation* learning algorithm have proven successful in many problems, such as learning to recognise hand-written characters [LeCun et al. 1989], learning to recognise spoken words [Lang et al. 1990], and learning to recognise faces from digital images [Cottrell 1990]. See Haykin (1994), Mitchell (1997) and Väinämö (1996) for further information about neural computing and the back-propagation algorithm.

Even though neural computing has proved to be an efficient intelligent application, it needs a functional framework to control itself in order to be fit for use in practical applications. The neural networks generally used have been static in nature. The information needed for training is obtained and pre-processed. The neural structure is created and trained by using the processed training material. With this approach, the architecture is ready for use in many complex real-world systems. This approach is called *off-line* use and training of neural networks.

In our case, however, there was a strict prior requirement to be adaptive to changing situations. This cannot be achieved using an off-line approach. A static neural structure can be re-trained after certain intervals with the newly collected and pre-processed data. This, however, requires human interaction and supervision. In order to automate this process, we need an on-line approach with automated functionality. This is the clear focus in our approach.

Contrary to static neural network models, an on-line learning system has a capability to adapt to the new samples. The training can be incremental with different approaches. A typical way to do the job is to collect training material and to re-train the network at regular intervals. With this approach, the basic model remains static, but adapts to the changing situations. The structure has a memory, because re-training is started using the previous static network structure. The previous structure has a static model stored in the network parameters. The new archived data adapts the network parameters in order to adapt the model to the changed situation. The problem in this approach is to decide when re-training must be carried out.

Applications of on-line learning systems can be found in many areas. These include pattern recognition [Park et al. 1997], controlling of non-linear plants [Zufiria et al. 1999], sensor validation [Napolitani et al. 1998] and robotics [Yaolong et al. 1998], [Xianyi et al. 1999]. An example of bad generalisation and adaptation that a statically trained network can bring to a system is presented in a comparison by Wilkinson (1996). The task of the remote sensing system was to segment and classify images from different geographical locations. When the system was trained statically to classify images in one location and was then applied to a new location, the classification accuracy dropped down to as low as 25%. However, the use of an on-line approach in the training of the system resulted in 80% accuracy of classification.

The distinction between on-line and off-line definitions is not clear. The distinctions overlap and may be confusing, and the terminology is used very inconsistently. The following algorithm descriptions try to illustrate the difference between the approaches [Sarle, 2000].

- Batch learning algorithm:

1. Initialise the weights.
2. Loop until a good performance level is found
  - 2.1. Loop: Train the structure using all the training data.
  - 2.2. Update the weights after each sample.

- Incremental learning algorithm:

1. Initialise the weights.
2. Loop once or a few predefined times:
  - 2.1. Process one training case.
  - 2.2. Update the weights.

- Intermediate learning algorithm (mini-batch):

1. Initialise the weights.
2. Loop once or a few predefined times:
  - 2.1. Process two or more, but not all training samples.
  - 2.2. Update the weights.

The term "batch learning algorithm" is used quite consistently in the neural network literature, but "incremental learning" is often used for on-line, constructive, or sequential learning. Bertsekas and Tsitsiklis (1996) have used these definitions in a systematic way. The main difference between the off-line and on-line approaches is naturally the training algorithms. In an off-line case, there are no critical requirements for training time or processor power. Off-line training (batch) algorithms, such as back-propagation, are related to conventional numerical optimisation techniques [Haykin, 1994]. In an on-line case, however, the incremental algorithms involve stochastic approximation techniques [Saad, 1998].

For off-line learning, it is characteristic that all the data is stored and can be accessed repeatedly. Batch learning is always off-line. In on-line learning, each case is discarded after it has been processed and the weights are updated immediately, so that the history information is stored into the weights of the network structure. On-line training is always incremental. Incremental learning can be done either on-line or off-line. With the off-line learning approach, supervision is needed for the training. The person who creates the model can measure and observe the progress made in training. Different learning algorithms can be selected interactively, if necessary, and the designer of the model has the experience and instinctive ability to make good selections. The generalisation of the model can be easily verified in different ways, using stored data samples [Haykin, 1994]. The model can be tested using an independent test data set, and the test gives quite a good estimate of how the model will perform in a real situation.

On-line learning is more problematic. You cannot calculate the magnitudes mentioned in the previous paragraph, because you cannot compute the objective function of the training set or the error function on the validation set for a fixed set of weights, since these data sets are not stored. This also makes the reliability of the on-line trained model questionable. This is why we have not used a pure on-line approach in our concept. The present method is a hybrid method that includes features from the on-line and incremental approaches. Training the model incrementally helps to eliminate part of the problems of pure on-line learning. It is therefore important to differentiate between the concepts of incremental and on-line learning.

Our hybrid on-line learning concept involves network growing as well. Network growing is often called constructive learning. The network growing algorithm adds units or connections to the network during the training. In most cases, network growing begins with a network with one hidden unit, which is trained to perform as optimally as possible with the given training set. The reason why it is relevant to use constructive learning in the present concept is that there is not much data available for training at the beginning. The main points of the concept presented in this paper for training a feed-forward back-propagation neural network are:

- network growing
- incremental on-line learning
- sequential re-training using a sample selection algorithm
- a model-balancing algorithm for the gating results given by classifier models (the models are defined in Chapter 3)

In the concept, network growing is started with one hidden unit and additional units are added in proportion to the size of the history buffer. The training is simplified in the sense that the existing structure or weights are not frozen after a new unit has been added. On the contrary, the whole structure is kept dynamic, which means that all weights are adjusted during the incremental training.

Incremental learning is done for every sample, with feedback from the output control. However, the learning rate is kept very small, to avoid significant changes in performance. Only slow adaptation takes place. Sequential training is controlled by a smart archive. It triggers a period of re-training whenever there is a significant change in the history data. SA is a training set builder as well. The sequential training sets do not include all the samples, but they include a subset of the history data such that the oldest samples have been dropped out and the latest samples have higher entropy in the training set. The entropy distribution is achieved using multiple copies of samples according to their relative age. The latest data samples have the largest amount of instances in the sequential training set.

### 3 PRINCIPLE OF A SMART ARCHIVE

#### 3.1 Overview of the presented concept

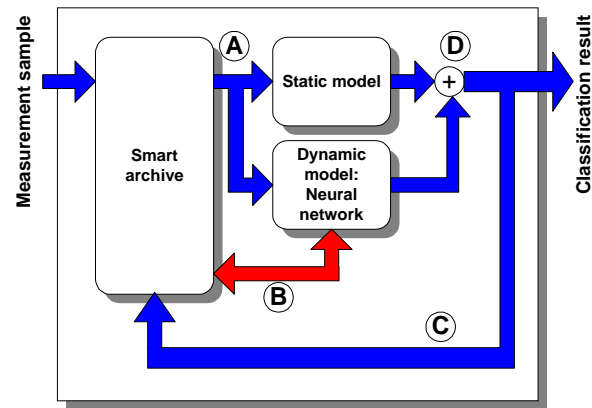
The overall system consists of a static model, a trainable dynamic model (neural network) and a smart archive. The concept of smart archive is a solution for the on-line learning system to archive very relevant information in order to handle the constantly changing situations. SA takes measurement signals as input and forwards them to classifiers. Figure 1 presents how the smart archive is related to the frame of the overall system. The input presentation is in a sample vector format.

The smart archive filters the input sample vectors and forwards them to the static model and the dynamic model. These models present a classification mapping as an output. The output presentation is the same in the static and dynamic classifiers, and that makes it possible to use voting or averaging algorithms for gating.

In the generation phase, the basic idea of the present on-line learning system with SA is first to create a basic static model, which gives as good an output as is optimally possible. In this case, the static information model is a fuzzy classifier, which is implemented using expert knowledge. Effectiveness is a very important issue in an industrial system, which is why a fuzzy classifier is used. Another reason is model integration. The static and dynamic models should have similar output presentations in order to use gating algorithms to produce the final classification result.

When using the present concept is used in an industrial case, for example, the static classifier must be implemented using prior information obtained from experts and from previous installation cases. The principle is that the static model is designed and implemented using human supervision and the best possible prior information for the application case. After an installation and start-up period, the dynamic model will have more responsibility to improve the accuracy of the overall classifier system. This does not exclude the re-creation of the static model, but the re-creation is only done at relative long intervals or when there are such radical changes in the environment that the adaptive system is unable to cope with them. This again involves human supervision and expert knowledge.

When the model is used, the history information accumulates in the SA. At regular intervals the neural network model is retrained and re-structured if necessary. Re-structuring is based on the network growing algorithms. The parallel neural network's role is to dynamically correct errors or inaccuracies, because the static model cannot adapt to changing situations.



*Figure 1. Logical model of the smart archive concept with an on-line learning system.*

Figure 1 includes the labels:

- A: On-line data stream to the classifier models
- B: On-line and sequential training data and control stream
- C: Feedback from the output; the correct result of the classification to be stored
- D: Gating of the results obtained from the classifier models

### 3.2 Structure of the Smart Archive

SA itself consists of four logical objects: on-line data preparation, data storage, training set generation and system control (Figure 2). Each object makes its own contribution to the overall functionality. On-line data preparation is used to filter outliers from the samples. This is achieved by using multiple measurement samples from the observed object. The filtering parameters are controlled by an archive control object. Smart data storage includes data validation, annotation processes and a history buffer. The function of the storage is to maintain history information about the system and to minimise the storage needed by removing outliers and redundant samples from the data. In the annotation process, a temporal attribute is added as well as a reliability attribute for the measurement of the data sample.

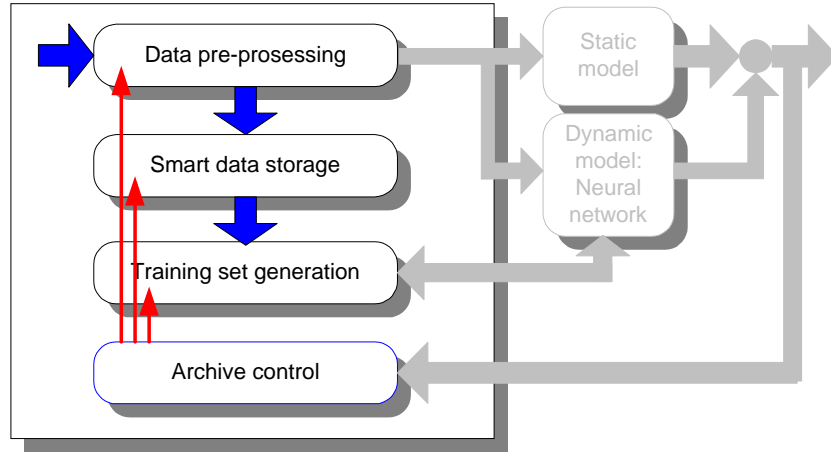


Figure 2. Smart archive architecture.

### 3.3 Information pre-processing

In on-line processing the data is filtered, pre-processed and forwarded to the static and dynamic information models (Figure 3). In this classification case, filtering is implemented using multiple measurement samples from the object being tested. Filtering involves calculating a representative measurement vector from multiple measurements. The filtering process includes vector scaling and normalisation as well.

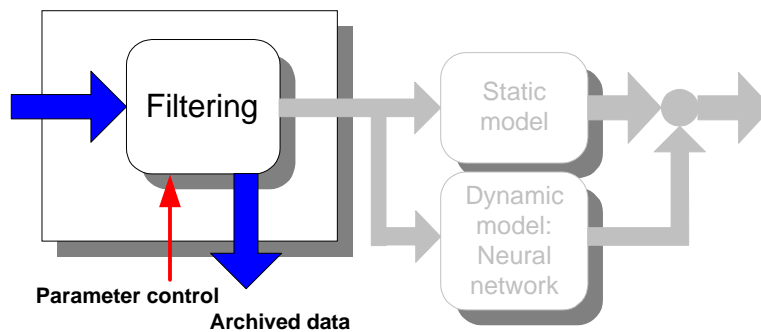
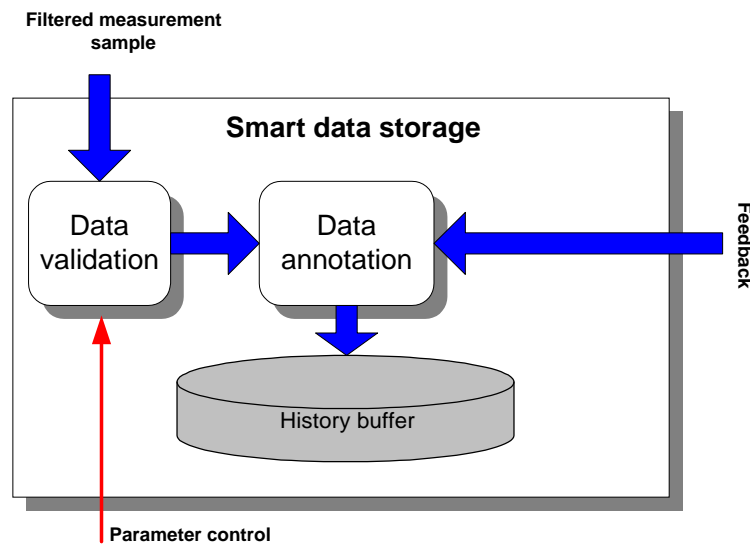


Figure 3. Input pre-processing.

### 3.4 History buffer

Data validation assesses the reliability and relevance of the data sample, to determine if it is a typical sample or a new one not seen earlier. The validation task decides if the sample is included in the history buffer (Figure 4). In validation, a reliability estimate is calculated for the sample. The annotation task encloses even more information in the sample: timestamp, basic target information (including surface information about the product under visual quality control) and, in the case of a classifier, the correct classification result if it is available. Re-training of the

dynamic information model is triggered by using this collected information and the dynamic changes in the data, for example, the trend detection algorithms.



*Figure 4. Information storage.*

This information selection is an important part of system generation. The input selection involves phases of problem and solution space exploration. The basic model is a fuzzy classifier. It is not optimal in accuracy when used in non-linear and complex applications, because it is implemented using insufficient measurement information. The information is insufficient, because the application environment keeps constantly changing. The benefit of the basic model is a fast start when installing on-line learning systems, such as industrial control systems. The system is usable immediately after the installation, but its accuracy will improve after the history data have been collected into a smart archive.

The problem with a history buffer, as more time elapses, is that the storage size grows, even when minimisation is optimised as much as possible. This leads to the concept of active forgetting. The oldest information is deleted, and the size of the storage is defined by the smart archive's control object. Figure 5 visualises how a classification cluster develops over time. The shape of the cluster changes and some re-training of the dynamical classifier is needed, because the static model is built based on old history data. When the situation changes and the oldest information is removed, the new classification accuracy is based on the re-trained dynamic classifier.

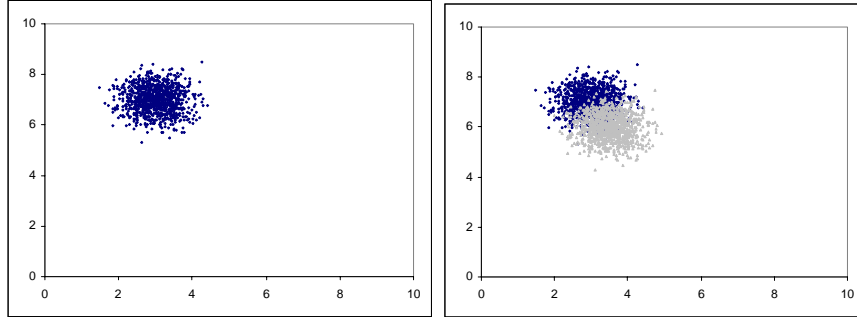


Figure 5 .a) and b) Cluster development over time.

The classifier history buffer includes, or should include, clusters of classification samples to obtain correct classifier results. It is important to measure the development of clusters to trigger sequential training and to build a training set for this purpose. This is done by tracking the centre of gravity of each class cluster. If this centre starts to move significantly, a decision on re-training is made. In the training set building phase, the oldest samples are excluded from the set with the same entropy as the latest measurement samples.

The classifiers are based on  $n$ -dimensional clusters, where  $n$  refers to the dimension of the measurement sample vector. Over time, more and more history data is gathered and there will be a significant amount of redundant information in the history buffer. In the stable state, the entropy will concentrate near the cluster core. The redundant information is a problem in two ways. It impairs the re-training efficiency and storage usage. A large number of redundant samples make the neural network re-training process slow.

The problem of redundancy minimisation is solved using *edge samples*. The redundant information, i.e. the core samples in the  $n$ -dimensional class cluster, is removed partially or totally. The role of the smart archive's control object is to define the percentage reduction of data attained by removing the core samples. At the same time, it is also necessary to remove the possible outliers that have accumulated after the filtering and validation process. Figure 6 visualises the principle of core sample removal, where the edge samples define the cluster properties. The figure is presented as a simplified two-dimensional diagram.

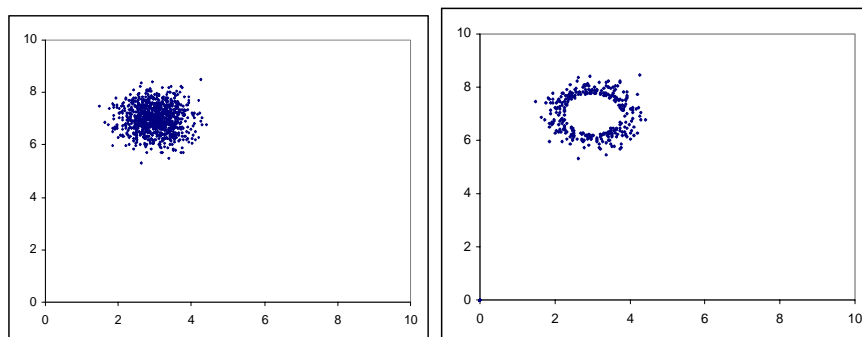


Figure 6. a) and b) Removal of redundant information.

### 3.5 Training set generation and re-training of the dynamic model

The role of the training set generation process is to, when triggered, create a subset from the samples in the history buffer. The temporal gaining is implemented using an entropy function. Figure 7 describes the architecture of the training set builder. The training set includes copies of samples in proportion to their relative age in the buffer. The newest samples have more copies than the oldest. This temporal gaining and reliability gaining (when available) could be implemented using only such factor attributes for each sample, and the learning rate coefficient is slightly scaled according to the given temporal and reliability factor. But it is out of the scope of this paper to present a learning algorithm derived from the back-propagation algorithm using momentum.

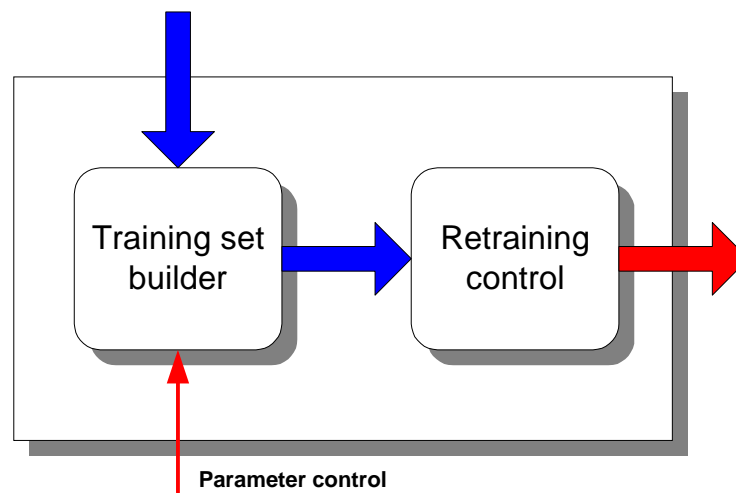


Figure 7. Training set builder.

The constructive learning part is implemented in such a way that only one hidden unit - neuron - is trained at first. At the beginning of the training, one unit is added during each sequential training session. There are very different opinions in the literature about the optimum size of the neural structure. The relation to the training set size is, however, quite clear. You cannot construct a very large neural structure if you have a small training set size. In our concept, we decided to limit the maximum hidden unit count to  $2n$ , where  $n$  is the size of the sample vector.

Incremental learning is done for every sample, with feedback from the output control. However, the learning rate is kept very small, to prevent significant changes in performance.

### 3.6 Smart archive control

The responsibility of the control object is to calculate, store and forward control parameters to other objects. The triggering of sequential training is generated in the control object as well. Figure 8 presents the internal structure of the control object. The object is divided into several sub-objects with their own responsibilities. This architecture has been designed aiming at reasonable and efficient software implementation.

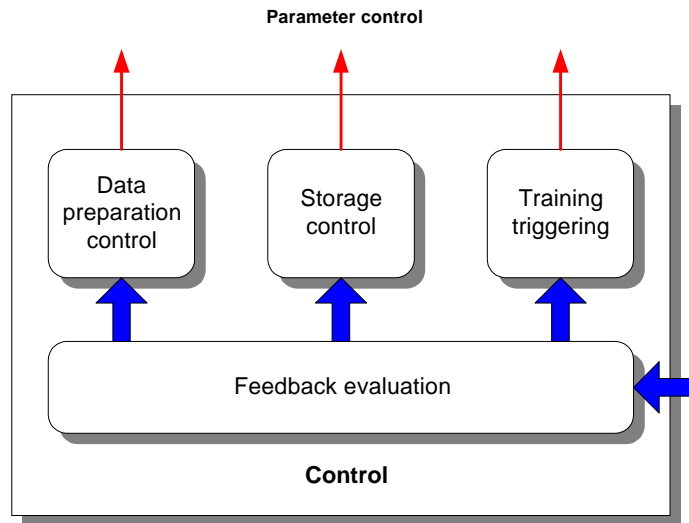


Figure 8. Control object architecture of the smart archive.

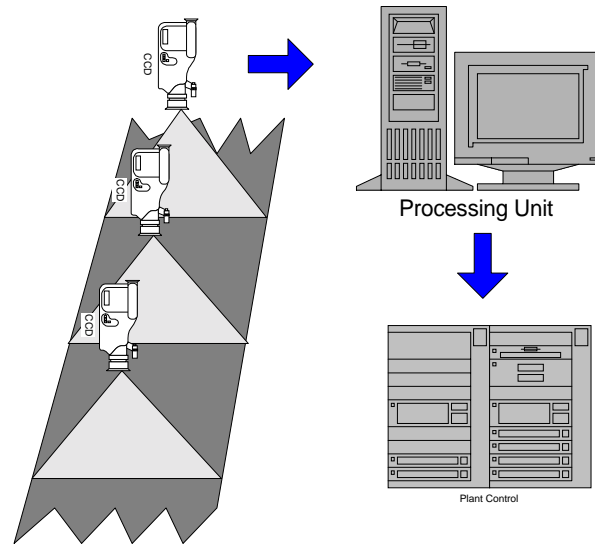
The result is interpreted using a gating algorithm. At the start-up phase, the system is based on a static fuzzy classifier, which simply gives an estimate of defect class membership for each class in a sample. The vector has a dimension of  $k$ , where  $k$  is the number of known defect classes. The result interpretation is implemented using a weighted sum with the results of the dynamic model. Both models have the same classification result representation: a vector including membership for a certain defect class. The final result is obtained using weighted sums and scaling. At the beginning, the overall model is based on the fuzzy classifier's accuracy. Hence, the fuzzy classifier has a factor of  $2n$  for the weighting sum and the neural network 1. After a certain period and when the number of training sequences and the size of the network have grown, the fuzzy classifier will have a smaller and the network a greater weight. In this way the overall framework balances the knowledge between the models.

#### 4 QUALITY CONTROL CLASSIFIER WITH A SMART ARCHIVE

The present concept is to be implemented in a real-life environment, i.e. an industrial quality control system using machine vision (Figure 9). The goal of the system is to find defects in the material strip under inspection. Another goal is to classify the detected defects into different classes as accurately as possible in order to give feedback to the overall production control system. The system includes an automation interface, which is used for communication with the plant production IT system. Another interface is available for a vision interface, where the measurement information is obtained for classification.

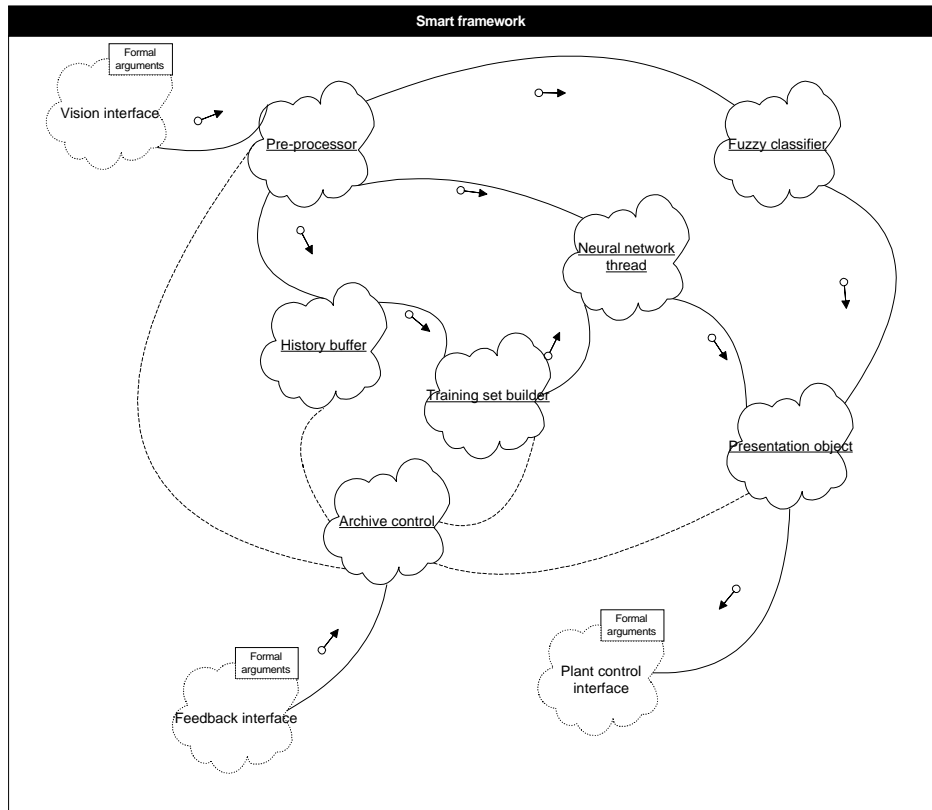
The system is implemented using a fuzzy classifier as a static model. The classifier is created using old inspection material and expert knowledge. The vision system controls the imaging process and, by using multiple cameras, it can provide multiple samples of the defects found. The

defects are classified according to their shape features as well as the grey level colour information. The imaging cannot, however, normalise the measurement completely, because there may be sudden changes in lighting due, for example, to welding, when service procedures are done near the cameras. Another source of disturbance is the material itself: there may be changes in the material when there are different production qualities.



*Figure 9. Example setup for the system.*

The overall software architecture is presented in Figure 10. The imaging system interface provides as normalised measurement samples from the found defects as possible. The sample vector consists of geometrical features of the defect. The pre-processor filters the sample vectors, selecting the median value from the multiple samples. The sample is forwarded to the classifier threads for modelling. This is done on-line and real-time.



*Figure 10. Software architecture of the concept.*

The off-line processing or, in this case, background processing with a lower priority consists of smart archive processing. The filtered data is forwarded to the history buffer for validation and annotation. The re-training of the dynamic model is implemented in such a way that it will not interfere with the real-time processing. The training of the network is done using a copy of the on-going processing structure, and the re-trained structure is copied to replace the active model between the real-time modelling periods.

The presentation object is responsible for presenting the results of the classifier models. At the beginning, the weighting is focused on the results of the static model. Later, when the dynamic model is totally constructed after a number of sequential training periods, the weighting is smaller for the static model.

In this case, there is no possibility get direct feedback on the classification result. The feedback problem is solved by storing both the image information and the measurement sample information. Both human supervision and automatic clustering tools for giving to the system the correct classification off-line are needed. With these limitations, the presented system can be used as an intelligent core of the industrial visual inspection system.

## 5 DISCUSSION AND CONCLUSIONS

A novel concept of *smart archive* is presented. The contribution of the paper is to organise different knowledge methods and algorithms into an automatic and efficient concept for an industrial on-line learning system. The basic idea of the concept is to provide an automatic on-line learning platform for expert applications in the industry.

For initialising the smart archive, a static model is needed. The better the initial model is, the faster the dynamic model will converge towards the optimal situation. There are several possibilities to set up the initial model. The first is to rely exclusively on expert knowledge using fuzzy techniques. Another approach is to implement a statistical classifier based on data obtained from the some time interval. Hence, the generation of the static model is a classic model generation problem, which has several solutions in the literature.

One of the challenges in a smart archive is to find methods to balance the history and the new data. In on-line learning, data is discarded after it has been processed. In a smart archive, however, part of it is stored. Even if this data accounts for merely a fraction of the total information flow, it will ultimately accumulate, and parts of it must be actively discarded at regular intervals. Discarding redundant information and using edge samples is a solution to minimise the size needed in training sets. However, it is important to give temporal gaining to the sample set information. This is done by using entropy as a function of time. The training set includes copies of samples according to their relative age in the history buffer. Another approach to temporal gaining is to use the relative age of the sample as an inverse factor for the learning rate in the network training phase. The traditional approach is that the older the sample is, the smaller is its effect on weight changes in the network.

A further problem with the on-line approach is that it is difficult, or downright impossible, to calculate the error function on the validation set for the fixed part of model, since these data sets are not stored. However, in our approach this can be solved. The smart archive contains a significant amount of information stored, and, during the sequential training phase, the validation sample set can be used for validating learning status. The training material is a sub-set of the overall history buffer information, and the validation set is a sub-set. Hence, it is important that these sub-sets do not include or only include minimally shared samples. There is naturally no need for temporal gaining in the validation set.

Rapidly changing data makes the previously discovered patterns invalid. The question to be solved is how quickly we allow the model to adapt to a changing situation. This depends on the choice of training parameters (e.g. the learning rate) and is highly application-sensitive. On-line adaptation is achieved using incremental training after each sample. This cannot adapt to rapid changes, because the learning rate in incremental training should be kept so slow that the performance is not impaired by noisy samples. Rapid changes are monitored in the archive by measuring the shift of the centre of gravity of class clusters in  $n$ -dimensional space. The temporal gaining function is derived from the core movement in such a way that, after a rapid change, the gaining is focused on the latest samples.

This paper presents an overall framework for an intelligent application. The focus is on the *smart archive* concept, which is based on a new approach of selecting the good features of the existing machine learning methods and presenting hybrid solutions to problems. The key idea is automation, efficiency, memory minimisation and a short start-up time, which make it useful for industrial environments. The work is still in progress, and the next phase will be to finish and prove that the concept works in an industrial environment and cases. Another theoretical on-going work aims to find a solution to automatically adapt new patterns to the overall system. This problem is frequently encountered in reality, for example, in a quality control classifier. When new a defect class appears, it is either misclassified or dumped into a class labelled 'unknown'. This problem also involves sub-problems, such as structural adaptation of the static and the dynamic models, when the output presentation vector size may grow, as well as problems in the Smart Archive of how and when to cluster the new patterns.

## 6 REFERENCES

Aleksander, I., And H. Morton, "*An introduction to Neural Computing*", Chapman & Hall, London, 1990.

Bertsekas, D. P. and Tsitsiklis, J. N., "*Neuro-Dynamic Programming*", Belmont, MA: Athena Scientific, ISBN 1-886529-10-8, 1996.

Cottrell, G. W., "*Extracting features from faces using compression networks: Face, identity, emotion and gender recognition using holons*", In D. Touretzky (Ed.), *Connection Models: Proceedings of the 1990 Summer School*, Morgan Kaufmann, San Mateo, CA, 1990.

Fahlman, S.E., and Lebiere, C., "*The Cascade-Correlation Learning Architecture*", in Touretzky, D. S. (ed.), *Advances in Neural Information Processing Systems*, Los Altos, CA: Morgan Kaufmann Publishers, pp. 524-532, 1990.

Haykin, S., "*Neural Networks; A Comprehensive Foundation*", MacMillan Collage Publishing Company, New York, 1994.

Lang, K. J., Waibel, A. H., & Hinton, G. E., "*A time-delay neural network architecture for isolated word recognition*", *Neural networks*, 3, p.33-43, 1990.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D., "*Backpropagation applied to handwritten zip code recognition*", *Neural computing*, 1(4), 1989.

Littmann, E., and Ritter, H., "*Learning and generalization in cascade network architectures*", *Neural Computation*, 8, 1521-1539, 1996.

Mitchell, T.M., "*Machine Learning*", The McGraw-Hill Companies Inc., New York, 1997.

Napolitano, M.R., Silvestri, G., Windon, D.A., II, Casanova, J.L. and Innocenti, M.,

“*Sensor validation using hardware-based on-line learning neural networks*” IEEE Transactions on Aerospace and Electronic Systems, vol 34 2, p. 456–468, 1998.

Park, J-M. and Hu Y-H., “*On-line learning in pattern classification using active sampling*” IEEE International Conference on Acoustics, Speech, and Signal Processing. ICASSP-97., vol 4, p. 3217 –3220, 1997.

Pyle, D., “*Data Preparation for Data Mining*”, Morgan Kaufmann Publishers Inc., San Francisco, 1999.

Saad, D., “*On-Line Learning in Neural Networks*”, Cambridge: Cambridge University Press, 1998.

Sarle, W.S., ed. (2000), Neural Network FAQ, part 2 of 7: Learning, periodic posting to the Usenet newsgroup comp.ai.neural-nets, URL: <ftp://ftp.sas.com/pub/neural/FAQ.html>. Referenced 27.3.2000.

Väinämö, K., “*Neural Networks for Human Aerobic Fitness Approximation*”, Department of Electrical Engineering, University of Oulu, Oulu, Finland. Diploma thesis, 1996.

Wilkinson, G.G., “*Open questions in neurocomputing for earth observation*”, Proc. 1st “COMPARES” Workshop, York, U.K., July 17–19, 1996.

Zufiria, P.J., Fraile-Ardanuy, J., Rianza, R. and Alonso, J.I., “*Neural Adaptive Control of Non-linear Plants via a Multiple Inverse Model Approach*”, International Journal of Adaptive Control and Signal Processing, vol 13, p 219-239, 1999.

Xianyi Y. and Meng, M., “*Real-time tracking control of robot manipulators with online learning based approach*”, IEEE Canadian Conference on Electrical and Computer Engineering, vol 2, p. 1035 –1040, 1999.

Yaolong L., Holtz, J., Lee, T.H., “*Critical implementation issues in compensation for nonlinearities in industrial robot manipulators by adaptive multilayer neural networks*”, Proceedings of the 1998 American Control Conference, vol 4, p. 2200 –2202, 1998.