Data Driven Prognostics using a Kalman Filter Ensemble of Neural Network Models

Leto Peel, *Member, IEEE (GOLD)*

***Abstract*—This paper details the winning method in the IEEE GOLD category of the PHM '08 Data Challenge. The task was to estimate the remaining useable life left of an unspecified complex system using a purely data driven approach. The method involves the construction of Multi-Layer Perceptron and Radial Basis Function networks for regression. A suitable selection of these networks has been successfully combined in an ensemble using a Kalman filter. The Kalman filter provides a mechanism for fusing multiple neural network model predictions over time. The essential initial stages of pre-processing and data exploration are also discussed.**

***Index Terms*—Kalman Filter, Ensemble, Radial Basis Functions, Multi-Layer Perceptron, Prognosis.**

I. Introduction

THIS paper details the author's method in response to the Prognostics and Health Management (PHM) '08 Data Analysis Challenge [1].

A. PHM '08 Data Challenge

The goal of the PHM '08 Data Challenge was to estimate the remaining useable life left in a set of unspecified components (referred to as units). The challenge was focused on data-driven techniques for PHM; no system or domain specific information was given. The data provided consisted of a multivariate time series for each unit, each of which starts with the unit in normal condition. The variables in each series corresponded to 3 operational settings (defining modes or operational tasks the unit was undertaking) and 21 sensor measurements. At an unspecified point in each series a fault occurs causing the unit to degrade, resulting in the eventual failure at the end of the series. The data provided was split into training and test sets, where each series in the test set was terminated some time before the system failure. Assessment of the algorithm performance was based on a function of the error in predicting the number of remaining time cycles. The scoring function, shown in (1), was asymmetric and exponential, such that late predictions were penalized more heavily than early predictions.

$$\begin{aligned} score(s<0) &= \exp(-s/13)-1 \\ score(s>0) &= \exp(s/10)-1 \\ score &= sum(score) \end{aligned} \tag{1}$$

The value s is the difference between the estimated and actual remaining life.

B. General Approach

The challenge was approached using an ensemble of regression models. The target of the regression was the number of remaining cycles before failure. The available training set was split into two sets; the first set contained the data from 145 randomly selected units and was used to train the models. The set of the remaining 73 units was used as a validation set to assess the trained models using (1) in order to select the best data representation and model architecture. The neural network architectures Multi-Layer Perceptron and Radial Basis Functions were used as the regression models and were trained using the Netlab toolbox [2].

The first steps in tackling this challenge were to use visualization techniques to explore and gain an understanding of the data; this is discussed in Section II. Section III considers the issues concerning the representation of the data as regression inputs. In Section IV the neural network architectures for performing the regression are presented. The use of the Kalman Filter for fusion of the ensemble models and filtering over time is covered in Section V. Finally conclusions are stated in Section VI.

II. Data Exploration

A purely data-driven approach takes no account of any prior system knowledge and therefore effective visualization of the data is a key first step in gaining an understanding of an unknown dataset. However, due to human perception

Manuscript received July 21, 2008. This work was supported by BAE SYSTEMS Advanced Technology Centre.

L. Peel is with the Advanced Information Processing Department, BAE SYSTEMS Advanced Technology Centre, Bristol, UK (phone: +44 (0)117-302-8159; fax: +44 (0)117-302-8007; e-mail: leto.peel@baesystems.com).

9778-1-4244-1936-4/08/$25.00 © 2008 IEEE

being limited to three dimensions, problems arise when trying to visualize more than three features at a time. Aside from plotting two or three features at a time, a number of visualization techniques exist to tackle this issue such as Principal Component Analysis, Self Organized Mapping [3], and Sammon Mapping [4].

Sammon Mapping is a topographic projection technique, which, for a given dataset, projects high dimensional $(\Re^d)$ data points into a lower dimension $(\Re^q)$ while attempting to preserve the relative distance between data points. A significant drawback of the Sammon Mapping approach is one of scalability, as the computational demands increase with the square of the number of data points [2]. A way around this is to use the Neuroscale model [5]. While the Neuroscale model also suffers the same computational disadvantage as Sammon Mapping, it does have the advantage of defining a non-linear mapping between $\Re^d \rightarrow \Re^q$. This means that a mapping can be learnt from a manageably sized sub-sample of the data, after which the mapping can be used to project the complete dataset into $\Re^d$ space.

The Neuroscale mapping is generated by training a Radial Basis Function neural network with d inputs and q outputs which minimizes the Sammon stress metric:

$$E_{sam} = \sum_{i=1}^{N} \sum_{j>i}^{N} \left(m_{ij} - m_{ij}^{*} \right) \quad . \tag{2}$$

Equation (2) is based on the (typically Euclidian) distance m_{ij} between points x_i and x_j in the original space $\Re^d$ and the distance m_{ij}^{*} between the mapped points y_i and y_j in the projected space $\Re^q$.

Fig. 1 shows the Neuroscale plot of the PHM '08 training set. The data points have been shaded according to the remaining life left in the unit, where black signifies normal and light grey is close to failure. The first point of interest to be noted from this figure is that there are 6 distinct clusters present in the data. It is known from the description of the data challenge that different modes of operation exist and so seems reasonable to assume that these correspond to 6 distinct modes of operation. This was confirmed by plotting the three operational settings for all the instances in a 3D plot (Fig. 2). It can be seen that each of the clusters in Fig. 1 contains the full spectrum from black to light grey, indicating that current mode on its own is not an indicator of remaining life left. Another observation from Fig. 1 is that the deviation of the grey data points from the black cluster centers indicate that a change occurs in the data values as the unit degrades. From this it can be inferred that the deviation from the cluster centers (in black) could provide some indication of remaining life.

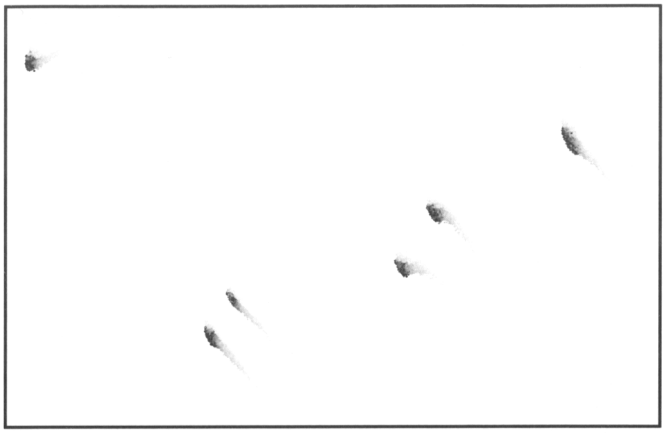


Fig. 1: Neuroscale visualization of the dataset. Lighter colored points indicate unit is closer to failure. It can be seen that there are 6 clusters in the data and deviation from the black centers occurs as the remaining life diminishes. NB: Axes values of a neuroscale plot are irrelevant and therefore have not been included.

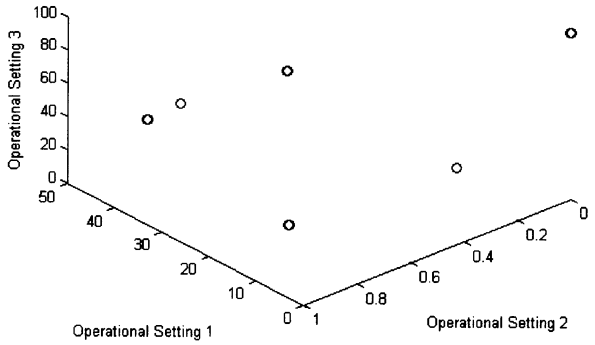


Fig. 2: Plot of the three operational setting variables of the 45918 training set instances confirms that there are 6 operational modes.

III. Data Representation

In order to set up the problem as a regression task, a number of considerations as to how the data would be best represented had to be taken into account. To address this, a selection of possible representations was chosen, over which a search using specific regression models was performed. This was in order to find the best performing representation for the given model. This section will discuss the features of data representation that were considered for this search.

A. Normalization

It is a common pre-processing step to perform component-wise normalization on a dataset before analysis in order to provide a common scale across all the features of a dataset. A standard method to achieve this is to use the equation:

$$N\left(x^d\right)=\frac{x^d-\mu^d}{\sigma^d},\forall d \qquad (3)$$

where x^d are the original data values for feature *d*, and μ^d,σ^d are the mean and standard deviation respectively for that feature. However, while this method does provide a uniform scale, it was noted that there were 6 modes of operation in the data and that the operation mode was independent of the number of cycles left; consequently it was considered that determining the mode from the sensor measurements was not required, and that instead maximizing the variance within each mode would be more useful for the given task. To facilitate this (3) can be modified to normalize by mode as well as feature;

$$N\left(x^{(m,d)}\right)=\frac{x^{(m,d)}-\mu^{(m,d)}}{\sigma^{(m,d)}},\forall m,d \qquad (4)$$

where *m* refers to one of the six possible modes. This latter method of normalization was found to yield the best performing models overall.

B. Time Representation

A significant characteristic of the PHM Challenge dataset is that it contains time series and therefore the representation of time or at least how to represent and take account of previous temporal observations is an important consideration.

The quickest and simplest method for representing time is to consider each time point independently and create a prediction at each time step. An alternative representation would be to consider using a phase space embedding representation, in which a sequence of instances is generated using a fixed length sliding window. However phase space embedding has the significant drawback of increasing the number of dimensions proportionally with each time step represented, giving rise to the problems associated with the "curse of dimensionality". From preliminary experiments it was found that the prediction performance did not improve significantly using the embedding space representation given the increase in computational demands. Therefore the chosen representation was to predict remaining life using single time points.

Based on the decision to make the predictions based on single time point instances, it seemed appropriate to incorporate the unit's history into the instances in some other manner. It is given in the data challenge guidelines [1] that the operational settings have a significant impact on the unit's performance; therefore it was assumed that operational mode history would be a good feature to include in the prediction model. This was achieved by adding an extra six features containing a total of the number of cycles spent in that mode since the beginning of the series.

IV. Neural Network Architectures

Artificial neural networks were used to perform regression on the data. The architectures implemented were multi-layer perceptron (MLP) and radial basis function (RBF) networks [6].

MLP is a widely used neural network architecture. The form of MLP used in this work was a two layer network, consisting of a hidden layer and an output layer, a layout which is known to be able to model any smooth function [7]. The number of hidden nodes was a parameter over which the models were searched.

RBF is an alternative neural network architecture, which again has a two layer setup in which models have to be searched over the number of hidden nodes. There are some practical advantages to using RBF networks over MLP networks, such as the two-stage training procedure which is somewhat faster than MLP training methods; however the reason for choosing this architecture was primarily to examine a contrasting architecture.

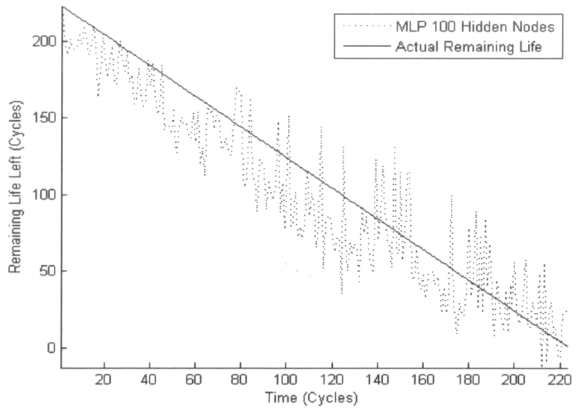
The predictions of two of the best models compared with the actual remaining life of a single unit from the training set can be seen in Fig. 3. These graphs highlight two important issues about the learnt regression models:

- The models contain random errors,
- The random errors vary between models.

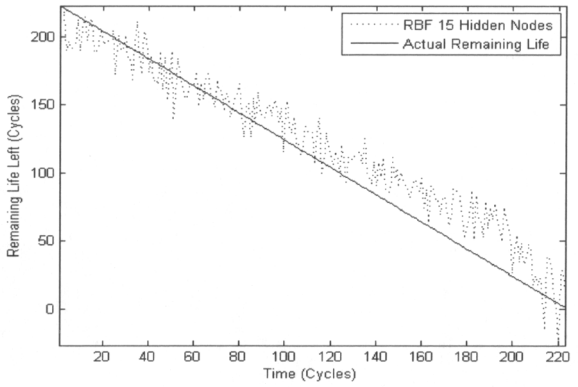
The first issue means that the predictions of these models appear noisy and maybe due to sensor noise. Note that each prediction did not take into account any of the previous predictions, i.e. each prediction in the series is independent of the others. As the only prediction of interest is the final one for each unit, this meant that the error could change by a large amount if the series terminated just a single time step earlier or later. Therefore to improve the reliability and consistency of the predictions as the series progresses, the output of the models needed to be filtered so that they took into account all of the previous predictions.

As for the second issue, the two graphs show that, for the given unit, one of the models is usually an overestimate, while the other is often an underestimate. This suggested a combination of these predictions could improve the performance of the prediction.

The concept of a combination of models is known as an ensemble and is a common way to improve the performance of a regression model [8]. It was deemed that a suitable mechanism for both containing the ensemble and filtering predictions over time was a Kalman Filter.



(a)



(b)

Fig. 3: The actual remaining life (solid line) and the predictions (dotted line) from two of the best models over the life of a single unit; (a) shows the performance of an MLP model with 100 hidden nodes, (b) shows an RBF model with 15 hidden nodes. Predictions are noisy and do not take account of previous predictions.

V. Kalman Filter Ensemble

A. The Linear Kalman Filter

A discrete linear Kalman Filter [9] is a recursive method for estimating the state of a process. It consists of two stages: a predict stage and an update stage. The predict stage is governed by the following equations:

$$x_k^- = Ax_{k-1} \tag{5}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{6}$$

where x_k is the state vector at time k, A is the transition matrix, P is the error covariance matrix and Q is the process noise covariance. The superscript minuses denote *a priori* estimates of the state and covariance variables given the knowledge prior to that time step. The equivalent variables without superscripts refer to *a posteriori* estimates given the observations at that time step. The predict stage updates the state estimate with the transition of time. This stage of the Kalman Filter allows the recursive incorporation of all points in the series and results in any given state estimate having some dependence on all previous state estimates. The process noise covariance matrix in (6) affects the smoothness of the resultant series, where low covariance values increase the amount of smoothing.

The second stage is the update stage which updates the state estimate with the observations. The update stage is given by:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{7}$$

$$x_k = x_k^- + K_k(z_k - Hx_k) \tag{8}$$

$$P_k = (I - K_k H)P_k^- \tag{9}$$

where K_k is the Kalman gain, H is the observation matrix which provides a mapping from state to observation space, R is the observation noise covariance and z_k is an observation at time k.

Any number of update steps can be made per prediction step, which facilitates the combining of a number of models in an ensemble fashion. The Kalman gain calculated in (7) can be considered as providing a weighting between the observations and the previous state estimates.

Regarding the application of the filter to this case, the state vector, x was chosen to contain the number of remaining cycles and the rate of change of cycles left (rate of degradation). The rate of change of cycles remaining was initialized to -1 based on the reasoning that with each time step transition, the unit would be one cycle closer to its eventual failure. Considering the outputs of the regression models as observations (z) allows the non-linearity of the problem to be dealt with by the regression models. Thereby the neural networks provide a non-linear mapping from the sensor measurements to give observations in the same space as the state. This not only allows a simple implementation of a linear Kalman Filter to be applied, but it also simplifies the task of deriving the mapping from state space to sensor space, which is essentially the inverse of the challenge problem. Finally, the R matrix for a specific model was set to its mean squared prediction error, σ^2, over the whole training set.

This means that for a Kalman Filter ensemble of n models z_k, H, and R take the form:

$$\mathbf{z}_k = \begin{bmatrix} z_k^1 \\ z_k^2 \\ \vdots \\ z_k^n \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \end{bmatrix},$$

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & & 0 \\ \vdots & & \ddots & \\ 0 & 0 & & \sigma_n^2 \end{bmatrix}.$$

B. Ensemble Selection

A selection of the best models found during the earlier model searches were used to conduct a final model search to find a good subset of models to be contained in the ensemble. A tournament style heuristic was developed to undertake this final stage of model selection.

A heuristic is required because an exhaustive search soon becomes intractable as the number of candidate models increases. For example, 16 models were selected to search over; an exhaustive search would require evaluating 2^{16} = 65536 possible model combinations. A possible heuristic is a greedy hill-climbing search. This involves incrementally finding the best model to add to the ensemble until the performance ceases to improve. However, this method is constrained to a very specific search path, e.g. any ensemble evaluated would have to contain the best single model, which is not necessarily a good ensemble candidate. The tournament heuristic aims to reduce the search space while still providing a broad search path.

The tournament heuristic randomly divided the set of models into equal subsets, e.g. four divisions of 4 models and exhaustively evaluated all possible ensembles in each subset ($4*2^4 = 2^6$). Each ensemble evaluated was scored and ranked using (1). The four most frequently occurring models in the n best ensembles overall are selected for the next round of the tournament, where n is chosen such that a distinction can be made between the forth and fifth most frequent model. An example of this is shown in Fig. 4 where n = 7 best ensembles were required to find the four most frequently occurring models. An exhaustive search was then carried out over these succeeding 4 models and the best generated ensemble was selected. This reduced the total search space down to $2^6 + 2^4 = 80$ ensembles.

This selection method works on the assumption that a good ensemble member is one that consistently performs well in an ensemble. It is possible that a better performing ensemble exists containing models which otherwise perform poorly, however these limitations were accepted given the intractability of an exhaustive search.

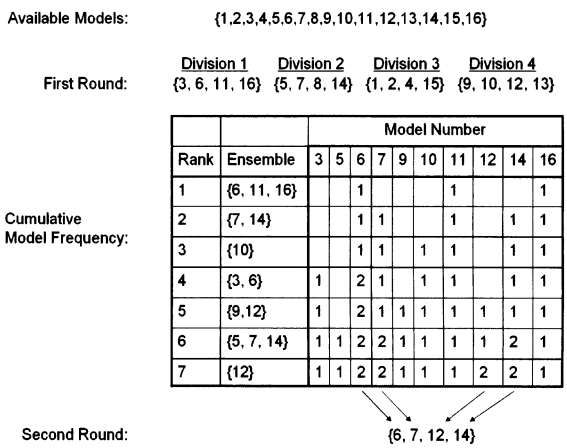


Fig. 4: Tournament heuristic search example over 16 candidate models using a division size of 4 models

The final ensemble selected consisted of three models: an RBF model with 15 hidden nodes and two MLP models, one with 75 hidden nodes and one with 100 hidden nodes. A graphical representation of this ensemble is shown in Fig. 5.

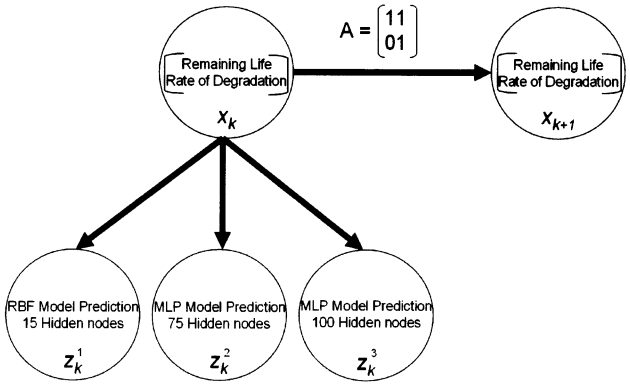


Fig. 5: Graphical representation of final Kalman filter ensemble

C. Results

Fig. 6 shows a graph of the final predictions compared to the actual remaining life left against time for a number of the units. Although the graphs of the units are shown as a continuous plot, it is important to note that the Kalman filter was re-initialized at the start of each unit's series. A smoother estimate can be seen in this graph as a result of applying the Kalman filter and it is noticeable that, within a single unit, the prediction improves over time.

It can also be seen in Fig. 6 that the starting predictions for each unit tend to be similar, regardless of the eventual length of the unit's life. This results in the general trend of overestimating short lived units and underestimating the longer ones. This apparent convergence of the initial predictions for each unit is likely to correspond to the mean or expected useable life of a unit. The variation between these initial predictions supports the notion that there are

"*different degrees of initial wear and manufacturing variation*" [1] between the individual units. The improvement of the prediction for each model over time would occur then as the model adjusts to the usage of the specific unit. This correlates with the "*operational settings that have a substantial effect on unit performance*" as stated in the data challenge guidelines [1].

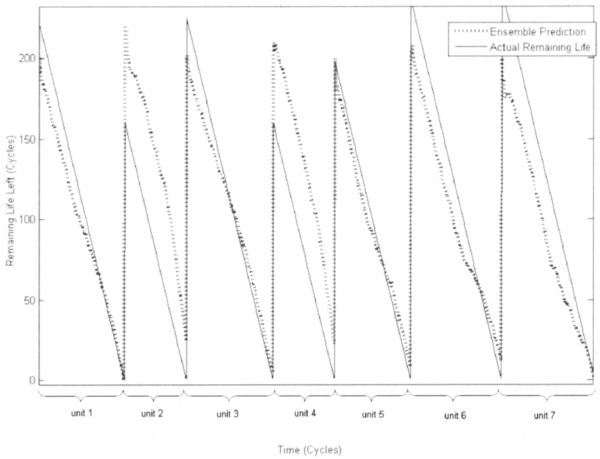


Fig. 6: Predicted and actual remaining life left for a selection of units. A smoother prediction which tends towards the ground truth over time can be observed after the application of the Kalman filter.

Table 1 shows the mean squared prediction error for each of the individual models with and without filtering compared with the error for the final ensemble model. It can be seen that filtering the individual models improves the prediction, with the greatest improvement being the final ensemble model.

TABLE 1
MEAN SQUARED ERROR FOR VARIOUS PREDICTION MODELS

Model	Without Filtering	With Filtering
RBF 15	1107	1105
MLP 75	1075	1008
MLP 100	1443	1210
Final Ensemble		984

VI. CONCLUSION

This paper has covered the winning algorithm (IEEE GOLD category) for the PHM '08 Data Challenge. It has been shown how a linear Kalman filter can be used to filter and contain an ensemble of mixed artificial neural networks for the purposes of determining remaining life left of a complex system in a purely data driven approach. The results have shown that the combination and filtering of models can yield a marked reduction in the prediction error.

Analysis of the model prediction suggests that the model determines the expected useable life of an average working unit and adjusts this prediction over time based on the usage data. There is also some evidence that the initial wear and variation is detected by the model.

Suggestions for further work would be to investigate feature selection methods, Kalman filter tuning, the use of other regression models, and learning of the residuals. In this work all parameters were used as inputs into the model, but it is possible that selecting a reduced set of features would improve the model. More work could be done on the tuning of the Kalman filter parameters to investigate how they affect the performance of the ensemble. This work has focused on the use of neural network models for regression, however this method could easily be extended to include any regression model. By looking at the results in Fig. 6 it can be seen that the predictions usually tend to be an over or underestimate on a per unit basis. This suggests that further improvement could be made by attempting to learn the residuals of each unit.

REFERENCES

[1] "Challenge Problem"; http://www.phmconf.org/OCS/index.php/phm/2008/challenge.

[2] I. Nabney, *NETLAB: Algorithms for Pattern Recognition*, Springer, 2002.

[3] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, 1982, pp. 59-69.

[4] J.W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on Computers*, vol. 18, 1969, pp. 401-409.

[5] D. Lowe and M. Tipping, "Feed-forward neural networks and topographic mappings for exploratory data analysis," *Neural Computing & Applications*, vol. 4, 1996, pp. 83-95.

[6] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, USA, 1995.

[7] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, 1989, pp. 359-366.

[8] A. Krogh and P. Sollich, "Statistical mechanics of ensemble learning," *Physical Review E*, vol. 55, 1997, pp. 811-825.

[9] R.E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, 1960, pp. 35–45.