Denoising autoencoder anomaly detection for correlated data

Peter Goldthorpe[1], and Antoine Desmet[2]

[1] *School of Mathematical and Physical Sciences, University of Newcastle, NSW, 2308, Australia*
Peter.Goldthorpe@newcastle.edu.au

[2] *Komatsu Mining Corporation, Rutherford, NSW, 2320, Australia*
Antoine.Desmet@mining.komatsu

ABSTRACT

In the mining industry, primary digging units such as wheel loaders are critical components due to their position at the start of the process chain. Consequently, the cost of unexpected downtime is high: this motivates efforts to provide an early warning of faults using remote diagnostics.

Machines are equipped with sensors that measure machine health. Some sensors are highly correlated and a model based on machine learning techniques can leverage such relationships across sensors to detect within-group abnormalities.

Autoencoders are auto-associative artificial neural networks which are trained to compress and rebuild the original input with minimal loss. The information is stored in the lower dimensional hidden layers as an internal coding. This is susceptible to a phenomenon called spillover, where the error in a single input can propagate through the network, corrupting the coding and biasing the entire reconstructed data. A denoising autoencoder is a more robust variation on the traditional autoencoder, trained to remove noise and build an error-free reconstruction.

We created a denoising autoencoder to utilize the noise removal on corrupted inputs, and rebuild from working inputs. While this technique is novel to this problem it remained susceptible to spillover. We show our findings and discuss future work for anomaly detection techniques in correlated data.

1. INTRODUCTION

Collection of telemetry data is becoming commonplace on mobile mining plants and other key elements in the mining process. Starting from basic payload and availability data, improvements in on-site network bandwidth have supported an increase in the scale of data collection (Siegel & Lee, 2012). Mining shovels and loaders record readings from hundreds of sensors, with sampling periods ranging from 100ms to 1s. This provides a clear picture of the state of the machine, its components, and the surrounding environment. Sensor data can be analysed to uncover symptoms of developing faults by monitoring the machine health over time. This enables the machine operator to make data-driven maintenance decisions, and reduce the cost in unplanned downtime and process interruptions. From the point of view of the manufacturer, this supports the development of better products and services.

While each sensor measures a unique environmental factor, some sensors jointly collect information on the same underlying phenomenon. For instance, there may be both temperature and current sensors in each phase winding of the same motor, which collectively measure the overall amount of mechanical work produced. Likewise, sensors across mechanically coupled motors are expected to produce readings that adhere to strong relationships. These sensors are said to exhibit dependence, where the levels of dependence can vary from being entirely independent to highly correlated.

Dependent sensors form correlated groups which have an added benefit of verifying the integrity of each channel in the group. Indeed, as the readings from sensors can be expressed as functions of one another, we can infer a sensor's reading from other group values. This inference can then be compared to the *actual* sample, to ensure the usual relationships between the dependent sensor values still hold. This approach is used to detect anomalies: if the relationships between dependent sensors are violated, it could indicate a faulty sensor, or a fault in the equipment being sensed. The interrelated sensor measurements also create a redundancy in the data, which can therefore be compressed in a lower dimension. The low-dimension data can then be decompressed back to original values by applying the known relationship to other sensors in the group.

Modeling highly correlated data with a single corrupt channel gives rise to a problem called spillover; a dimensionality reduction induced error when faults in a single input propagate

through the group reconstructions. Spillover is a problem as it "dilutes" any errors from a channel into the remaining healthy channels. The faulty channel corrupts the basis dimension, and from this incorrect coding, the error is propagated to every channel upon decoding.

To demonstrate spillover, suppose we have four inputs $X_{1\cdots4}$ with an identical value x. Because the values are all highly correlated the dimensionality reduction perfectly encodes the inputs to a single value ($\overline{x}$). The coding is directly mapped to the four outputs in the decoding stage. Now let us suppose one of the channels, X_1 instead has an error term a, which gives us a corrupt value of $X_1 = x + a$. The mean of these values becomes $x + a/4$. Upon reconstruction, the faulty channel exhibits a reconstruction error: $\widehat{X_1} = x + a/4$ instead of $X_1 = x + a$. While this is desirable, we notice that the three other healthy channels also have an induced error of $a/4$ compared to their input value. The error "spills over" to the non-faulty channels.

The spillover effect means the reconstruction error on individual outputs are not proportional to the corresponding input's deviation from normality. In our previous example, an ideal output would have $\widehat{X_{1\cdots4}} = x$, or what the values *should* be if the sample were error-free. The only reconstruction error would be on X_1 and would be equal to a while the other normal outputs ($\widehat{X_{2\cdots4}}$) would have zero reconstruction error.

The spillover effect is inversely proportional to the number of channels in a group as a single error contributes $1/N$ of the spillover effect to other channels. There are two methods to minimize or remove spillover. The first is by adding a larger number of correlated channels to a model. This is done by increasing the number of sensors recording the underlying phenomena and validating the current channels. This is not a practical solution as it overburdens a machine with sensors and increases the number of redundant data points. A better option is to build a model which is robust to corruptions in a minority of inputs, and thus avoids spillover.

This paper focuses on building an improved model for anomaly detection using the method previously outlined to avoid the spillover problem altogether. We will focus our work on highly dependent sensor data that share the same lower dimensional basis.

2. OBJECTIVES

This approach presented in the previous section has the advantage of being easily automated. An entire dataset can be mined to uncover dimensionality-reduction relationships using statistical methods, or machine learning-based modeling. We can feed one of these methods a sample of time-series data produced by a machine in healthy condition, and extract relationships that are typical of a machine in good condition. This gives a baseline to monitor how well incoming data adheres to these relationships. The system we present in this paper searches for broken relationship across a group of dependent sensors that indicates a possible departure from normality. The cause may be due to various reasons; a faulty sensor, a change in the underlying relationship, or a difference in environment, all of which represent a fault or failure. The type of event we are most interested in predicting is one that highlights a change in the environment, that can be solved with a maintenance intervention.

3. RELATED WORK

In this section, we first provide an overview anomaly detection techniques, and then focus on the techniques that exploit strong relationships between dependent sensors.

3.1. Expert Models

The most time consuming model is one manually coded by an expert (Witten, Frank, Hall, & Pal, 2016). A human writes rules that describe their perception of normal operation into code and this code is run against some dataset to detect anomalies. As the working environment of each machine is different, a new model may be required for each machine. This is time consuming and subject to human error and requires experience and good knowledge of the data to place in manual checks. Preferably we would search for a model that can be easily generalized and automatically trained on data from the same machine (Langley & Simon, 1995).

3.2. Autoregressive Models

The autoeregressive model uses an anomaly detection method based on a forecasting approach. Forecasts are issued based on sensor readings, using some mathematical relationship between past, current and future data. Forecast values are compared to actual sensor readings, and an anomaly is detected when residuals exceed a certain threshold.

A univariate autoregressive model uses the sum of weighted coefficients from previous time measurements to predict future values (Fitzmaurice, Davidian, Verbeke, & Molenberghs, 2008; Akaike, 1969). This model is most suitable for modeling periodic trends, where an effect is most apparent from previous time measurements. As many phenomena are dependent on the overall state and operation of the machine, the data in our study will appear irregular when observed over time. Example of irregularities occur when starting and stopping the machine at arbitrary times, or switching between a digging task and relocation of a mobile digging unit. A univariate autoregressive model is unsuitable for our application as machines are not deterministic or time independent.

3.3. Autoencoders

An autoassociative model aims to perform an identity mapping given some restraint. That is, it will attempt to reproduce its input on the output side, so long as the sample's values validates some relationship. If the input sample does not fit these constraints, the model will output a sample which is as close as possible to the input, but that also meets constraints.

If the relationships in the autoassociative model are built over healthy data, then further healthy data will be recreated. In contrast, loss of fidelity on the output side is a sign that there is an anomaly in the input data. Therefore the anomaly detection step amounts to searching for discrepancies between the input and output of the model, which has been built using healthy data.

Autoencoders (AE) are a type of autoassociative model based on dimensionality reduction. An input layer is mapped to a lower dimension (coding layer) then rebuilt by reversing the mapping (Goodfellow, Bengio, Courville, & Bengio, 2016). The process of mapping to a lower dimension is referred to as "encoding" and rebuilding the original sample from the codings is "decoding". An advantage of AEs is that the AE only needs to be trained once and, this is an offline process. Once training is complete, the AE can quickly process inputs through a set of matrix multiplications. This is in contrast with clustering techniques where the distances or proximity metrics need to be evaluated at each run for both normal samples, and those yet to be classified as anomalous (Desmet & Delore, 2017).

3.3.1. Principal Component Analysis Autoencoders

Principal Component Analysis (PCA) is a form of dimensionality reduction which performs a linear mapping to a set of lower dimensional basis vectors, which are listed in descending order of maximized variance. The least explained dimensions are dropped, and the model is reconstructed from the lower dimension. This ensures maximum information about the dataset is retained (Géron, 2017; Friedman, Hastie, & Tibshirani, 2001). This method is useful to model redundant data, as minimal information is lost when mapping highly correlated inputs to the N best principal components (Christopher, 2016).

Decoding is performed by an inverse mapping of the encoding. Normal samples will have most of their variance on the principal components, and will be rebuilt with minor loss. On the other hand, abnormal samples which are not well modeled by the PCA process will exhibit large variance on the "minor" components – which the AE omits. In the case of abnormal samples, the omission or minor component values amounts to a loss of information. This prevents the faithful reconstruction of anomalous samples as it is expected from an AE. PCA has the advantage of being a deterministic algorithm which produces optimal and reliable outputs. It is extremely fast to build the model and to compute projections. However, its major limitation is the inability to map non-linear relationships.

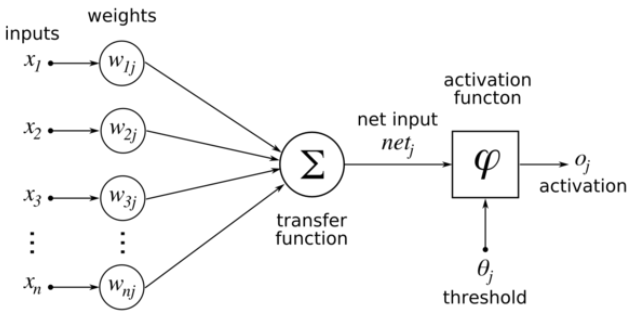

Figure 1. Artificial Neural Network.

3.3.2. Artificial Neural Network Autoencoders

Many models can be described in terms of an artificial neural network (ANN). At a high level, an input layer is passed through a number of hidden layers before returning a desired transformed output. An ANN is shown in Figure 1 where an input layer is passed through transformations as they propagate through the hidden layers with different levels of abstraction (LeCun, Bengio, & Hinton, 2015).

Artificial neural networks can be trained to map inputs to outputs, which makes it an AE. For the remainder of this paper, neural network architectures are implied when we refer to AEs. The dimensionality reduction is achieved by restricting the number of neurons in the central hidden layer. Without this constraint, the neural network would simply learn the identity function. Unlike its PCA counterpart which can only use linear manifolds, the ANN AE supports advanced nonlinear mappings. However, it is important to note that an ANN with linear activation, mean squared error (MSE) loss function and a single hidden layer is equivalent to PCA. The number of neurons in the central hidden layer are equivalent to the number of top principal components retained.

3.3.3. Denoising Autoencoders

In this paper, we introduce a particular training method to produce a "denoising autoencoder". A denoising AE is trained by purposefully corrupted samples as inputs and given the "true" samples as target output. The result is an AE which is trained to deal with noisy inputs in a way that they do not affect the coding (Sakurada & Yairi, 2014). The result is a "denoised" output where not only anomalies can be identified, but the channel(s) responsible for the error are also clearly identified.

3.3.4. Hierarchical Extreme Learning Machines

An Extreme Learning Machine (ELM) is a single layered ANN with randomly assigned weights and biases in the first and hidden layer. These parameters are not tuned after randomization but instead the weights and biases from the output layer allow for training (Huang, 2014; Cambria et al., 2013)

This had led to promising research in Hierarchical Extreme Learning Machines (HELMs) for the detection and isolation of faulty channels in correlated inputs (Michau, Palme, & Fink, 2017; Hope, Resheff, & Lieder, 2017). After training, the hidden layer (or coding) is extracted and passed through a single point classifier to quantify a deviation from normality. This method avoids the spillover problem as machine health is summarized as one statistic: the deviation from healthy data. If the statistic returns a faulty reading, The original hidden layer is used to determine where the fault lies. Both methods are worth exploring further, but for this paper we choose to investigate a denoising autoencoder.

4. CASE STUDY

The purpose of this paper is to create a denoising autoencoder to objectively scan and validate individual inputs in highly correlated data. The autoencoder will selectively remove noise from inputs to reconstruct an anomaly-free sample. By harnessing the denoising effect to map faulty inputs to their expected value, the denoising autoencoder should independently map inputs, avoiding the dependent spill-over effect induced in other models.

4.1. Preparing data

Data were queried from Komatsu telemetry databases on four highly correlated motor temperatures for a single articulated wheel loader machine. 28 days of data were collected from the 10th November 2017 to 8th December 2017 with a sampling period of 100ms, giving a total of 3545799 time points.

4.1.1. Cleaning and resampling data

The data were rebinned to 30 second intervals, as we observed the motor temperatures were unlikely to greatly change in under 30 seconds. Re-sampling was performed by calculating the median of 3000 non-missing data values. The median was chosen over mean to reduce the significance of outliers from faulty data. Re-sampling reduced the number of points in the 28 day period down to 56484. Missing data were filled by linear interpolation as the temperature was expected to slowly change between any of the last two observed values. Samples of data are shown in Figures 2 and Figure 3.

The data from November exhibited healthy behaviour with no faults and was chosen as the training data, consisting of 40503 points (72% of the dataset). In December an intermittent fault became apparent in one of the channels. The temperature reached unrealistically high levels in a few seconds – diagnosed as a sensor error. As December data had a good distribution of working and faulty data, it was chosen as the test data, consisting of 15982 points (28% of the dataset).

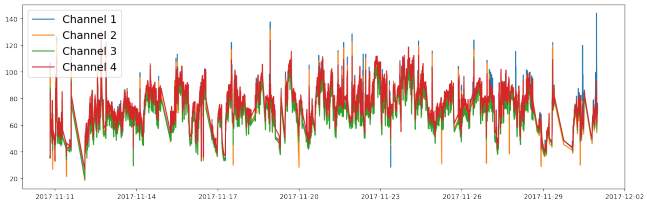

Figure 2. Loader temperature data November 2017

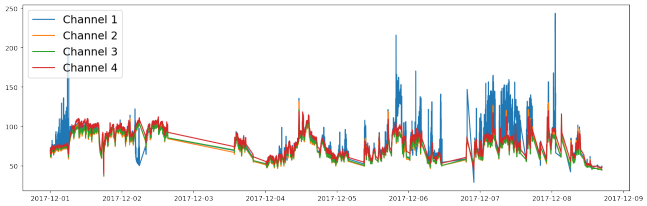

Figure 3. Loader temperature data December 2017 (channel 1 displays intermittent anomaly)

4.2. Setup and experiment

We developed a denoising autoencoder in Python 3.6 using the Google TensorFlow library (Zaccone, Karim, & Menshawy, 2017). The model is trained by introducing errors into individual channels with a Gaussian N(0,50) deviation, chosen to teach the model how to become more robust to a range of errors. For each input, 20% of the data is injected with an error so that a total of 80% of the data is noisy. The remaining 20% of data is left clean. The dataset is then shuffled to avoid learning time dependence patterns then fed as inputs to the training routine, where the target is the original clean sample. Thus, we utilize the denoising effect to teach the AE to produce the original sample, given its corrupted version.

4.2.1. Hyperparameters

Beyond the training process, ANNs have a set of hyperparameters that must be tuned to achieve optimal performance. In our case, tunable parameters are:

- network architecture
- activation function
- weights and biases initialization
- loss function
- number of epochs
- learning rate

The network architecture and activation function are adjustable hyperparameters to produce the optimal model. The remaining parameters adjust the learning rate. They have no direct consequence on the final trained model but instead influence

training time and escaping local minima to achieve the optimal model.

Network architecture is the most important feature for this problem, as it determines the dimensionality reduction, and how well the network can detect noise and implement ways to remove it. The task of choosing a network architecture is made difficult by the infinite number of architectures to choose from. Because the channels are highly correlated and redundant, we start with an architecture featuring a single hidden layer with one hidden node. We then increase the number of nodes in the hidden layer up to 100, and trial 2 additional layers with up to 10 nodes per layer.

Activation function sets the ANN AE apart from PCA by choosing the activation functions in hidden layers. We trial sigmoid, tanh and leaky rectified linear unit (leaky RELU) with leaks in steps of 0.05. We also trial a leak of zero, corresponding to a linear activation function.

Weights and Biases are initialized using Xavier initialization. Weights are sampled from a truncated normal distribution with standard deviation of 0.1, and biases are set to a constant value of 0.1 (Glorot & Bengio, 2010).

Learning rate was tested in 10-fold magnitude steps between 10^{-6} and 10^{-1}.

Optimizer used ADAM optimization based on the successful use in previous literature (Kingma & Ba, 2014; Géron, 2017; Goodfellow et al., 2016).

Epochs We find that learning rate coupled with ADAM optimizer and Xavier initialization converged consistently after 5000 epochs.

5. RESULTS

For each set of hyperparameters, we train three models to ensure consistency in results. No difference was observed between the iterations. We made three main discoveries from our experiment with detecting anomalies in highly correlated inputs using a denoising AE.

1. **Larger network architectures provided no benefit**
The denoising autoencoder was trained on different network architectures, but we found that increased complexity did not provide any performance improvement. The simplest architecture (single hidden layer with one node) performed identically to networks with a larger depth and width. This is due to the highly correlated nature of the data. All inputs can be represented by a single dimension so additional nodes in any layer become redundant and shared the coding value including errors. Additional layers provided no benefit as the error propagates from the smallest hidden layer.

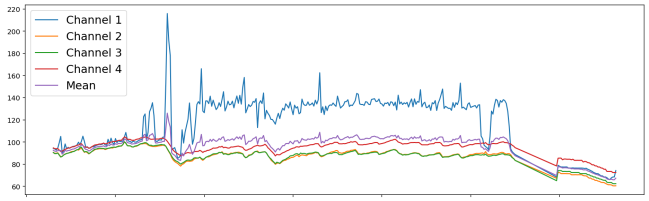

Figure 4. Raw Data.

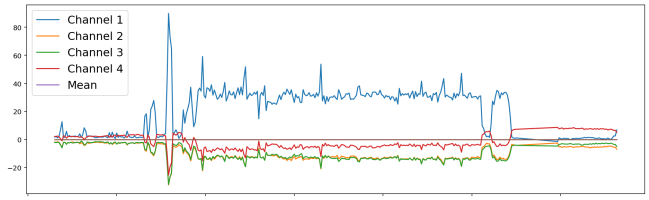

Figure 5. Reconstruction Error.

2. **A non-linear activation function performed best**
The model performed best with a linear activation function. The leaky RELU activation function performed as well as linear activation functions. We assume this is due to the fact that the input data were linearly correlated, therefore linear activation functions perform best.

3. **The denoising AE did not remove spillover**
The third finding is a direct consequence of the first two. The minimal topology with a linear activation function performed almost equivalent to PCA. This is because a near-optimal model, one with a single node in the hidden layer can be achieved. Since there is only one hidden node, the other layers do not develop the ability to remove noise so the outputs remained susceptible to spillover. The denoising effect slightly adjusted (or blurred) the weighting of each input to the hidden layer and this "blurring" effect was the only difference between the denoising autoencoder and PCA.

A denoising AE with a single hidden layer and linear activation function is shown to demonstrate the spillover effect.

Figure 4 shows a sample of the spillover effect induced from the denoising autoencoder. Channel 1 has an error resulting from an intermittent sensor wire connection. The "mean" curve shows the AE coding. Because of an error in channel 1 the coding is artificially high.

Figure 5 shows the reconstruction error (comparing the original data to the model). The channel 1 curve which is responsible for the error exhibits the highest error which spills over to the three other channels. Hence channels 2,3,4 are overestimated during the reconstruction process, since the coding is artificially raised by the error in channel 1.

6. CONCLUSION

In this paper we have discussed the application of denoising AEs. Our aim was to train a model to remove errors from inputs to train the model, given error-free inputs. This is helpful not only to detect anomalous samples, but also to pinpoint in which channel are affected. The largest finding from our experiment is that our denoising autoencoder did not independently remove errors from the channels in highly correlated data. Instead the model continued to be susceptible to spillover. This is in spite our attempts to increase the network architecture and activation functions. We assume the training became stuck in a local minima, which consists of taking the average of all inputs as the coding. This solution indeed measurably reduces the reconstruction error, yet the global minima (i.e. a true denoising encoder) might have been too distant (in terms of training steps) from this local minima to be attained.

Neural networks are universal approximators, i.e. able to model any function given a sufficiently large architecture to deal with the complexity of the problem. Therefore, the theory indicates that an ANN AE could produce our desired denoising behaviour. Future research will focus on understanding why the model became stuck in a local, sub-optimal minima. This could be due to training, or because even the largest of architecture we used was still too small.

REFERENCES

Akaike, H. (1969). Fitting autoregressive models for prediction. *Annals of the institute of Statistical Mathematics*, *21*(1), 243–247.

Cambria, E., Huang, G.-B., Kasun, L. L. C., Zhou, H., Vong, C. M., Lin, J., ... others (2013). Extreme learning machines [trends & controversies]. *IEEE Intelligent Systems*, *28*(6), 30–59.

Christopher, M. B. (2016). *Pattern recognition and machine learning*. Springer-Verlag New York.

Desmet, A., & Delore, M. (2017). Leak detection in compressed air systems using unsupervised anomaly detection techniques..

Fitzmaurice, G., Davidian, M., Verbeke, G., & Molenberghs, G. (2008). *Longitudinal data analysis*. CRC Press.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer series in statistics New York.

Géron, A. (2017). *Hands-on machine learning with scikit-learn and tensorflow: concepts, tools, and techniques to build intelligent systems*. " O'Reilly Media, Inc.".

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256).

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.

Hope, T., Resheff, Y. S., & Lieder, I. (2017). *Learning tensorflow: A guide to building deep learning systems*. O'Reilly Media, Inc.

Huang, G.-B. (2014). An insight into extreme learning machines: random neurons, random features and kernels. *Cognitive Computation*, *6*(3), 376–390.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Langley, P., & Simon, H. A. (1995). Applications of machine learning and rule induction. *Communications of the ACM*, *38*(11), 54–64.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

Michau, G., Palme, T., & Fink, O. (2017). Deep feature learning network for fault detection and isolation..

Sakurada, M., & Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the mlsda 2014 2nd workshop on machine learning for sensory data analysis* (p. 4).

Siegel, D., & Lee, J. (2012). Advanced diesel engine health monitoring algorithms for ground vehicles..

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Zaccone, G., Karim, M. R., & Menshawy, A. (2017). *Deep learning with tensorflow*. Packt Publishing Ltd.

BIOGRAPHIES

Peter Goldthorpe completed his B. Mathematics/B. Science and M. Medical Statistics from the University of Newcastle. He performed a research internship at Komatsu Mining Corp in the machine health and prognostics department.

Antoine F.D. Desmet received his undergraduate degree in electronics engineering from ESEO, Angers, France; Master by research from the University of Wollongong, Australia; and PhD from Imperial College London, UK. He is currently working for Komatsu Mining Corp., applying Machine Learning to anomaly detection on mining machinery.