On partitioning of an SHM problem and parallels with transfer learning

G. Tsialiamanis, D.J. Wagg, P.A. Gardner, N. Dervilis & K. Worden
Dynamics Research Group, Department of Mechanical Engineering, University of Sheffield
Mappin Street, Sheffield S1 3JD, UK

Abstract

In the current work, a problem-splitting approach and a scheme motivated by transfer learning is applied to a structural health monitoring problem. The specific problem in this case is that of localising damage on an aircraft wing. The original experiment is described, together with the initial approach, in which a neural network was trained to localise damage. The results were not perfect, partly because of a scarcity of training data, and partly because of the difficulty in resolving two of the damage cases. In the current paper, the problem is split into two sub-problems and an increase in classification accuracy is obtained. The sub-problems are obtained by separating out the most difficult-to-classify damage cases. A second approach to the problem is considered by adopting ideas from transfer learning (usually applied in much deeper) networks to see if a network training on the simpler damage cases can help with feature extraction in the more difficult cases. The transfer of a fixed trained later between the networks is found to improve classification by making the classes more separable in the feature space and to speed up convergence.

Key words: Structural health monitoring (SHM), machine learning, classification, problem splitting, transfer learning.

1 Introduction

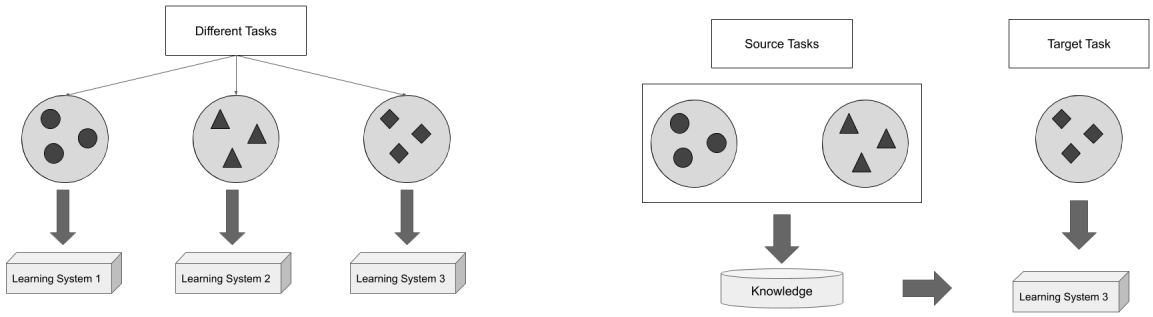
Structural health monitoring (SHM) refers to the process of implementing a damage detection strategy for aerospace, civil or mechanical engineering infrastructure [1]. Here, damage is defined as changes introduced into a system/structure, either intentionally or unintentionally, that affect current or future performance of the system. Detecting damage is becoming more and more important in modern societies, where everyday activities depend increasingly on engineering systems and structures. One the one hand, safety has to be assured, both for users and for equipment or machinery existing within these structures. On the other hand, infrastructure is often designed for a predefined lifetime and damage occurrence may reduce the expected lifetime and have a huge economic impact as a result of necessary repairs or even rebuilding or decommison. Damage can be visible on or in structures, but more often it is not, and has to be inferred from signals measured by sensors placed on them.

An increasingly useful tool in SHM is *machine learning* (ML) [1]. In many current applications large sets of data are gathered by sensors or generated by models and these can be exploited to gain insight into structural dynamics and materials engineering. Machine learning is employed because of its efficiency in classification, function interpolation and prediction using data. Data-driven models are built and used to serve SHM purposes. These models can also be used to further understand how structures react to different conditions and explain their physics. However, one of the main drawbacks of such methods is the need for large datasets. ML models may have many parameters which are established during *training* on data which have to span all the health conditions of interest for the given structure or system. Larger datasets assist in better tuning of the models as far as accuracy and generalisation are concerned. In the current paper, increased accuracy of a data-driven SHM classifier will be discussed in terms of two strategies: splitting the problem into two sub-problems and attempting transfer of information between the two sub-problems in a manner motivated by transfer learning [2].

Transfer learning is the procedure of taking knowledge from a source domain and task and applying it to a different domain and task to help improve performance on the second task [2]. An accurate representation of the difference

between traditional and transfer learning schemes can be seen in Figure 1. The SHM problem herein will be addressed using neural networks [3], for which transfer learning has been proven quite efficient (although usually in deeper learning architectures). Due to the layered structure of the networks, after having created a model for a task, transferring a part of it (e.g. some subset of the layers) is easy. The method is used in many disciplines, such as computer vision [4, 5]. The most commonly-used learners are deep Convolutional Neural Networks (CNNs), which can be very slow to train and can need a lot of data, which in many cases can be hard to find (e.g. labelled images). These problems can be dealt with by using the fixed initial layers of pre-trained models to extract features of images, and then train only the last layers to classify in the new context. In this way, both the number of trainable parameters and the need for huge datasets and computation time are reduced. Another topic that transfer learning has been used in, is natural language processing (NLP) [6], where the same issues of lack of labelled data and large amounts of training time are dealt with by transferring of pre-trained models into new tasks. Further examples of the benefits of transfer learning can be found in web document classification [7, 8]; in these cases, in newly-created web sites, lack of labelled data occurs. To address this problem, even though the new web sites belong to a different domain than the training domain of the existing sites, the same models can be used to help classify documents in the new websites.

In the context of the current work, transfer learning is considered in transferring knowledge from one sub-problem to the other by introducing pre-trained layers into new classifiers. The classification problem that will be presented is related to damage class/location. A model trained to predict a subset of the damage classes (source task) with data corresponding of that subset (source domain), will be used to boost performance of a second classifier trained to identify a different subset of damage states.



(a) (b)

Figure 1: Traditional (a) and transfer (b) learning schemes (following [2]).

2 Problem description

SHM problems are often broken down according to the hierarchical structure proposed by Rytter [9]:

1. Is there damage in the system (*existence*)?
2. Where is the damage in the system (*location*)?
3. What kind of damage is present (*type/classification*)?
4. How severe is the damage (*extent/severity*)?
5. How much useful (safe) life remains (*prognosis*)?

A common approach to the first level is to observe the structure in its normal condition and try to find changes in features extracted from measured signals that are sensitive to damage. This approach is called *novelty detection* [10, 11], and it has some advantages and disadvantages. The main advantage is that it is usually an *unsupervised* method, that is only trained on input data that are considered to be from the undamaged condition of the structure, without a specific target class label. These methods are thus trained to detect any *changes* in the behaviour of the elements under consideration, which can be a disadvantage, since structures can change their behaviour for benign reasons, like changes in their environmental or operational conditions; such benign changes or *confounding influences* can raise false alarms.

In this work a problem of damage localisation is considered (at Level 2 in Rytter's hierarchy [9]); the structure of interest being a wing of a Gnat trainer aircraft. The problem is one of supervised-learning, as the data for all damage cases were collected and a classification model was trained accordingly. Subsequently, the classifier was used to predict the damage class of newly-presented data. The features used as inputs to the classifier were novelty indices calculated between frequency intervals of the transmissibilities of the normal condition of the structure (undamaged state) and the testing states. The transmissibility between two points of a structure is given by equation (1), and this represents the ratio of two response spectra. This feature is useful because it describes the response of the structure in the frequency domain, without requiring any knowledge of the frequency content of the excitation. The transmissibility is defined as,

$$T_{ij} = \frac{FRF_i}{FRF_j} = \frac{\frac{\mathcal{F}_i}{\mathcal{F}_{excitation}}}{\frac{\mathcal{F}_j}{\mathcal{F}_{excitation}}} = \frac{\mathcal{F}_i}{\mathcal{F}_j} \tag{1}$$

where, $\mathcal{F}_i$ is the Fourier Transform of the signal given by the i^{th} sensor and FRF_i is the *Frequency Response Function* (FRF) at the ith point.

The experiment was set up as described in [12]. The wing of the aircraft was excited with a Gaussian white noise using an electrodynamic shaker attached on the bottom surface of the wing. The configuration of the sensors placed on the wing can be seen in Figure 2. Responses were measured with accelerometers on the upper surface of the wing, and the transmissibilities between each sensor and the corresponding reference sensor were calculated. The transmissibilities were recorded in the 1-2 kHz range, as this interval was found to be sensitive to the damage that was going to be introduced to the structure. Each transmissibility contained 2048 spectral lines.

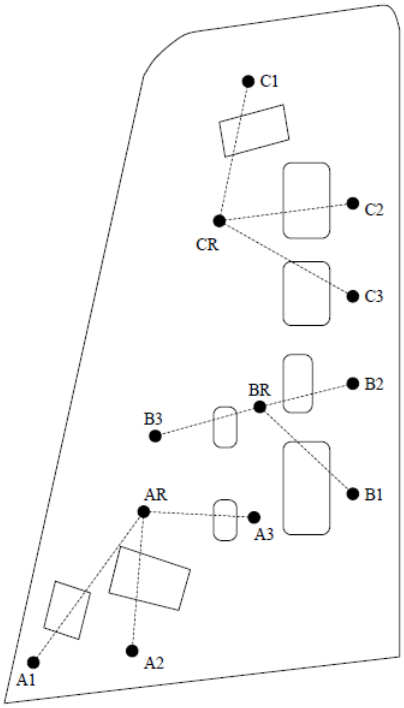


Figure 2: Configuration of sensors on the Gnat aircraft wing [13].

Initially, the structure was excited in its normal condition, i.e. with no introduced damage. The transmissibilities of this state were recorded and subsequently, to simulate damage, several panels were removed from the wing, one at a time. In each panel removal, the wing was excited again with white Gaussian noise and the transmissibilities were recorded. The panels that were removed are shown in Figure 3. Each panel has a different size, varying from 0.008 to 0.08m^2 and so the localisation of smaller panels becomes more difficult, since their removal affects the transmissibilities less than the bigger panels. The measurements were repeated 200 times for each damage case, ultimately to 1800 data points belonging to nine different damage cases/classes. The data were separated into training, validation and testing sub sets, each having 66 points per damage case.

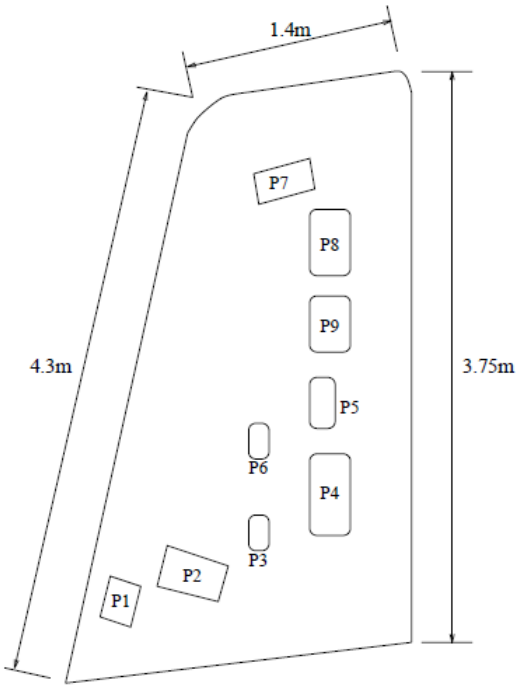


Figure 3: Schematic showing wing panels removed to simulate the nine damage cases [13].

For the purposes of damage localisation, features had to be selected which would be sensitive to the panel removals; this was initially done manually [13], selecting by visual 'engineering judgement' the intervals of the transmissibilities that appeared to be more sensitive to damage and calculating the novelty indices of each state by comparison with the transmissibilities of the undamaged state. The novelty indices were computed using the Mahalanobis squared-distance (MSD) D_ζ^2 of the feature vectors $\mathbf{x}_\zeta$, which in this case contained the magnitudes of transmissibility spectral lines. The MSD is defined by,

$$D_\zeta = (\mathbf{x}_\zeta - \overline{\mathbf{x}})^T S^{-1} (\mathbf{x}_\zeta - \overline{\mathbf{x}}) \tag{2}$$

were $\overline{\mathbf{x}}$ is the sample mean on the normal condition feature data, and S is the sample covariance matrix.

After selecting 'by eye' the most important features for damage detection [13], a genetic algorithm was used [14] to choose the most sensitive ones further, in order to localise/classify the damage. Finally, nine features were chosen as the most sensitive and an MLP neural network [3] with nine nodes in the input layer, ten nodes in the hidden layer and nine nodes in the decision layer was trained. The confusion matrix of the resulting classifier is shown in the Table 1. It can be seen that the misclassification rate is very low and that the damage cases that are most confused are the ones where the missing panel is Panel 3 or Panel 6, which were the smallest ones.

Predicted panel	1	2	3	4	5	6	7	8	9
Missing panel 1	65	0	0	0	0	0	0	0	1
Missing panel 2	0	65	0	1	0	0	0	0	0
Missing panel 3	1	0	62	0	0	1	0	1	1
Missing panel 4	0	0	0	66	0	0	0	0	0
Missing panel 5	0	0	0	0	66	0	0	0	0
Missing panel 6	0	3	0	0	0	62	0	1	0
Missing panel 7	0	0	0	0	0	0	66	0	0
Missing panel 8	1	0	0	0	0	0	0	65	0
Missing panel 9	0	0	0	0	0	0	0	0	66

Table 1: Confusion Matrix of neural network classifier, test set, total accuracy: 98.14% [14]

3 Problem splitting

As mentioned in [15], the rule-of-thumb for a network that generalises well is that it should be trained with at least ten samples per weight of the network. The aforementioned network had 180 trainable weights (and another 19 bias terms) so the 596 training samples do not strictly suffice. As a solution, a splitting of the original problem into two sub-problems is considered here to try and reduce the misclassification rate on the testing data even further. The dataset is split into two parts, one containing all the damage cases except Panels 3 and 6 and the second containing the rest of the data. Subsequently, two neural network classifiers were trained separately on the new datasets. This was thought to be a good practice, since the panels are the smallest, and their removal affects the novelty indices less than the rest of the panel removals. The impact is that the points appear closer to each other in the feature space, and are swamped by points belonging to other classes, so the initial classifier cannot separate them efficiently. By assigning the tasks to different classifiers, an increase in performance is expected, especially in the case of separating the two smallest panel classes.

To illustrate the data feature space, a visualisation is attempted here. Since the data belong to a nine-dimensional feature space, principal component analysis (PCA) was performed on the data and three of the principal components, explaining 71% of total variance, are plotted in scatter plots shown in Figure 4a. Points referring to data corresponding to the missing panels 3 and 6 (grey and magenta points respectively) are entangled with other class points causing most of the misclassification rate shown above.

The two neural networks were initialised several times and trained for different sizes of the hidden layer to find the ones with optimal structure for the newly-defined problems. After randomly initialising and training multiple neural networks for both cases and keeping the ones with the minimum loss function value the best architectures were found to be networks with nine nodes in the hidden layer for both cases and seven output nodes for the first dataset and two for the second. The loss function used in training was the categorical cross-entropy function given by,

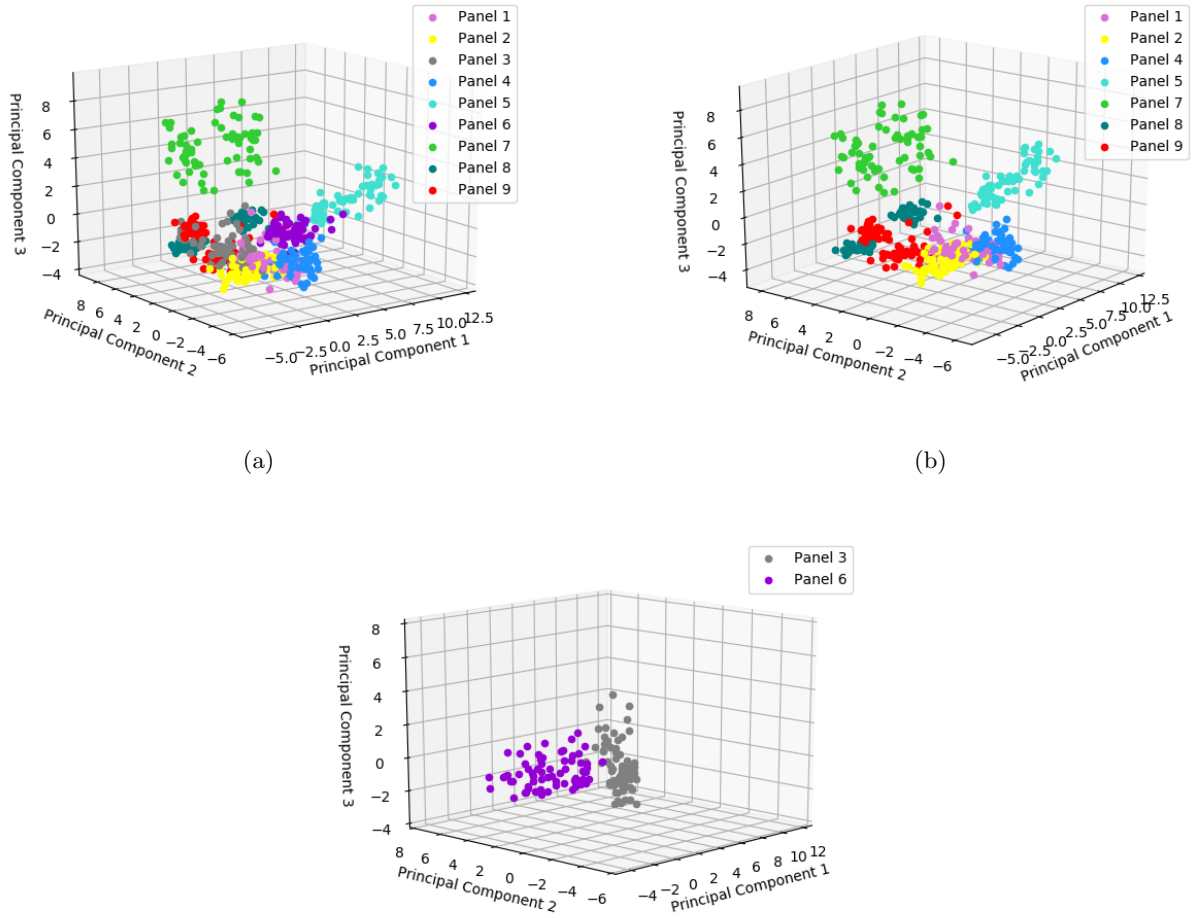
$$L(y,\hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n_{cl}}[y_{i,j}log\hat{y_{i,j}} + (1-y_{i,j})log(1-\hat{y_{i,j}})] \tag{3}$$

In Equation (3), N is the number of samples during training, n_{cl} is the number of possible classes, $\hat{y_{i,j}}$ the estimated probability that the ith point belongs to the jth class and $y_{i,j}$ is 1 if the ith sample belongs to the jth class, otherwise it is 0.

Confusion matrices on the test sets for the classifiers are shown in Tables 2 and 3. By splitting the dataset into two subsets the total accuracy is slightly increased from 98.14% to 98.82%. This is best considered in terms of classification error, which has been reduced from 1.86% to 1.18%, and this is an important reduction in SHM terms. Reduction of the number of trainable parameters has certainly contributed to this improvement, since the amount of training data is small. Performance on the task of separating only the two smallest panel classes was also increased because it is an easier task for the classifier than trying to discriminate them among the panel removals with greater impact on the novelty indices. This fact is also clear in Figure 4c, where the principal components of samples belonging to the classes of missing Panels 3 and 6 are clearly separable.

Predicted panel	1	2	4	5	7	8	9
Missing panel 1	65	1	0	0	0	0	0
Missing panel 2	0	63	1	0	0	0	2
Missing panel 4	1	0	65	0	0	0	0
Missing panel 5	0	0	0	66	0	0	0
Missing panel 7	0	0	0	0	66	0	0
Missing panel 8	1	0	0	0	0	65	0
Missing panel 9	1	0	0	0	0	0	65

Table 2: Confusion Matrix of neural network classifier trained on the first dataset, test set, total accuracy: 98.48%



(a)

(b)

(c)

Figure 4: Principal components of all samples (a), samples excepting panels 3 and 6 (b) and samples of panels 3 and 6(c).

Predicted panel	3	6
Missing panel 3	66	0
Missing panel 6	0	66

Table 3: Confusion Matrix of neural network classifier trained on the first dataset, test set, total accuracy: 100%

4 Knowledge transfer between the two problems

Having split the problem into two sub-problems, a scheme motivated by transfer learning in deeper learners was examined. The idea being to establish if the features extracted at the hidden layer in one problem, could be used for the other. In transfer learning termology, the seven-class problem specifies the source domain and task, while the two-class problem gives the target domain and task. The transfer is carried out by using the fixed input and hidden layers from the classifier in the source task, as the input and hidden layers of the target task; this means that only the weights between the hidden and output layers remain to be trained for the target task. This strategy reduces the number of parameters considerably. The functional form of the network for the source task is given by,

$$\mathbf{y} = f_0 W_2 (f_1(W_1 \mathbf{x} + b_1) + b_2 \tag{4}$$

where f_0 and f_1 are the non-linear activation functions of the output layer and the hidden layer respectively, $W_{1,2}$ are the weight matrices of the transformations between the layers, $b_{1,2}$ are the bias vectors of the layers, $\mathbf{x}$ is the input vector and $\mathbf{y}$ the output vector. The *softmax* function is chosen to be the activation function of the decision layer, as this is appropriate to a classification problem. The prediction of the network, concerning which damage class the sample belongs to, is the index that maximises the output vector $\mathbf{y}$; the outputs are interpreted as the *a posteriori* probabilities of class membership, so this implements a Bayesian decision rule. Loosely speaking, one can think of the transformation between the hidden and ouput layers as the actual classifier, and the transformation between the input layer into the hidden layer as a map to latent states in which the classes are more easily separable. In the context of deep networks, the hope is that the earlier layers carry out an automated feature extraction which facilitates an eventual classifier. In the deep context, transfer between problems is carried out by simply copying the 'feature extraction' layers directly into the new network, and only training the later classificatiion layers. The simple idea explored here, is whether that strategy helps in the much more shallow learner considered here. The transfer is accomplished by copying the weights W_1 and biases b_1 from sub-problem one directly into the network for sub-problem two, and only training the weights W_2 and biases b_2.

As before, multiple neural networks were trained on the first dataset. In a transfer learning scheme, it is even more important that models should not be overtrained, since that will make the model too case-specific and it would be unlikely for it to carry knowledge to other problems. To achieve this here, an early stopping strategy was followed. Models were trained until a point were the value of the loss function decreases less than a percentage of the current value. An example of this can be seen in Figure 5 where instead of training the neural network for 1000 epochs, training stops at the point indicated with the red arrow.

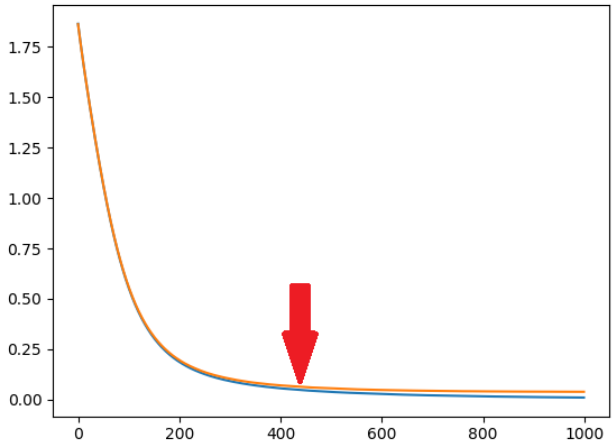


Figure 5: Training and validation loss histories and the point of early stopping (red arrow).

After multiple networks were trained following the early stopping scheme above, the network with the lowest value on validation loss was determined and the transfer learning scheme was applied to the second problem. The nonlinear transformation given by the transition from the input layer to the hidden layer was applied on the data of the second dataset. Consequently, another neural network was trained on the transformed data, having only one input layer and one output/decision layer. To comment on the effect of the transformation, another two-layer network was trained on the original second dataset and the results were compared.

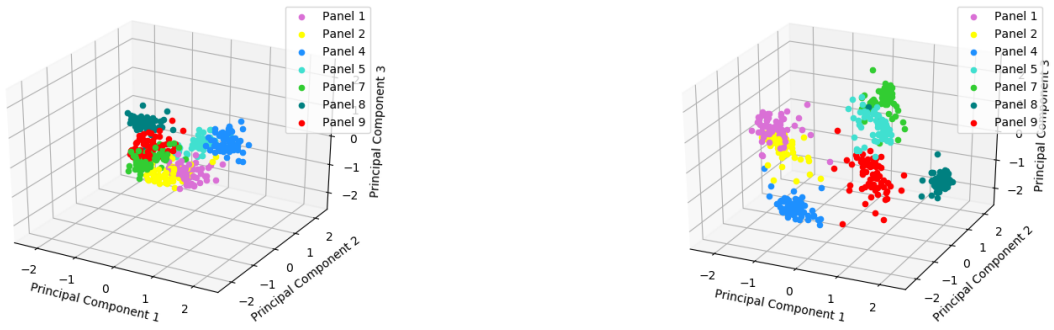
Predicted panel	3	6
Missing panel 3	65	1
Missing panel 6	2	64

Table 4: Confusion Matrix of neural network classifier trained on the original data of the second dataset, test set, total accuracy: 97.72%

The confusion matrices of the two neural networks on the testing data are given in Tables 4 and 5; the misclassification

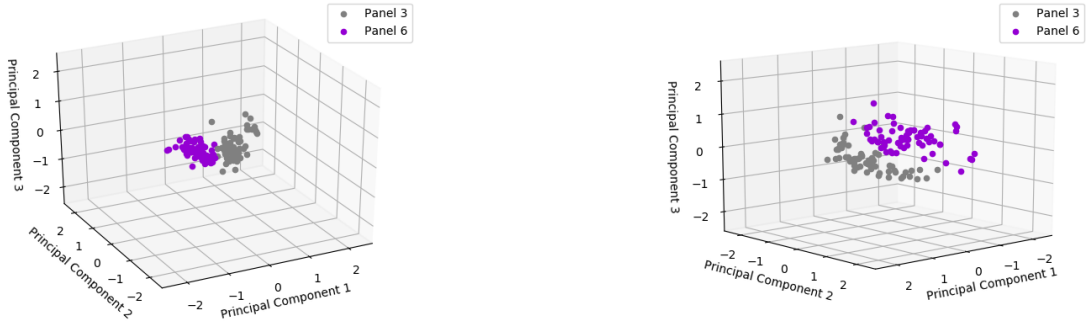
Predicted panel	3	6
Missing panel 3	65	1
Missing panel 6	3	63

Table 5: Confusion Matrix of neural network classifier trained on the transformed data of the second dataset, test set, total accuracy: 96.96%



(a) (b)

Figure 6: Principal components of original features of the first dataset (a) and transformed features (b).



(a) (b)

Figure 7: Principal components of original features of the second dataset (a) and transformed features (b).

rates are very similar. However, it is interesting to also look at the effect of the transfer on the convergence rate of the network trained on the transferred data and also to illustrate the feature transformation on the first and the second datasets.

The training histories of the two models can be seen in Figure 8. It is clear that the loss history of the model with transformed data (blue and cyan lines) converges faster, especially in the initial part of the training, and it also reaches a lower minimum value for the loss function in the same number of training epochs. This can be explained by looking at the effect of the learnt transformation on the data. In Figures 7 and 6 this effect is illustrated. (Note that the points are different from those in Figure 4, because principal component analysis was performed this time on the normalised data in the interval [-1, 1] for the neural network training.) The transformation spreads out the points of the original problem (first dataset) in order to make their separation by the decision layer easier; however, it is clear that it also accomplishes the same result on the second dataset. The points in Figure 7b are spread out compared to the initial points and thus, their separation by the single layer neural network is easier. Furthermore, the points lay further away from the required decision boundary and this explains both the faster training convergence

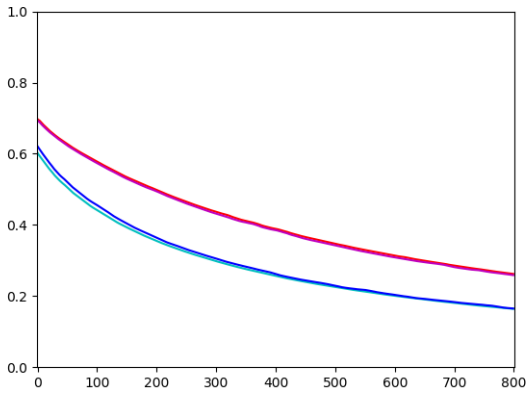


Figure 8: Loss histories of transferred model: train(blue), validation(cyan) and model trained on initial data: train(red), validation(magenta).

and the lower minimum achieved. In contrast to the transformation of the first dataset, in the second dataset, the transformation does not concentrate points of the same class to the some centre. In Figure 6b the points are both spread and concentrated closer according to the class they belong. This probably means that only a part of the physics of the problem is transferred in the second problem through this specific transformation.

5 Discussion and Conclusions

For the SHM classification (location) problem considered here, splitting the dataset into two subsets contributed to increasing the classification accuracy by a small percentage. This result was explained by the lesser effect that the small panel removals had on the novelty index features. This issue arose because the points representing these classes were close to each other and also points from other classes – those corresponding to large panel removal/damage. By considering the two damage cases as different problem, perfect accuracy was achieved in the task of classifying damage to the small panels, and there was also a small increase in the performance of the classifier tasked to identify the more severe damage states.

An attempt at a crude form of transfer learning was also investigated. Having trained the neural network classifier on the first dataset of the seven damage cases, transfer of knowledge to the second sub-problem was considered. This was accomplished by copying the first two layers of the first classifier – the 'feature extraction' layers – directly into the second classifier and only training the connections from the hidden layer to the output. The result is not particularly profound; the transfer does allow a good classifier, even with the smaller set of trainable parameters, but is not as good as training the network from scratch. The result is interesting, because it is clear that the source network is carrying out a feature clustering and cluster separation on the source data, that is still useful when the target data are presented. This suggests that the main issue with the small-panel damage classification is that the data are masked by the close presence of the large-panel data. Separating out the small-panel is the obvious answer. The results are interesting because they illustrate in a 'toy' context, how the early layers in deeper networks are manipulating features autimatically in order to improve the ultimate classification step. The other benefit of the separation into sub-problems, was the faster convergence of the network training.

6 Acknowledgement

The authors would like to acknowledge the support of the Engineering and Physical Science Research Council (EPSRC) and the European Commission (EC). This work was started while N.D., D.J.W and K. W. were supported by EPSRC grant EP/K003836/2, and finished under the EC Marie Skłodowska-Curie ETN grant "DyVirt" (764547) which directly supports G.T.

References

[1] C.R. Farrar and K. Worden. *Structural Health Monitoring.: A Machine Learning Perspective*. John Wiley and Sons, 2012.

[2] S.J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

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

[4] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[5] H.-C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R.M. Summers. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, 2016.

[6] J. Bingel and A. Søgaard. Identifying beneficial task relations for multi-task learning in deep neural networks. *CoRR*, abs/1702.08303, 2017.

[7] G.P.C. Fung, J.X. Yu, H. Lu, and P.S. Yu. Text classification without negative examples revisit. *IEEE Transactions on Knowledge and Data Engineering*, 2006.

[8] H. Al-Mubaid and S.A. Umair. A new text categorization technique using distributional clustering and learning logic. *IEEE Transactions on Knowledge and Data Engineering*, 2006.

[9] A. Rytter. *Vibrational Based Inspection of Civil Engineering Structures*. PhD thesis, Aalborg University, 1993.

[10] K. Worden. Structural fault detection using a novelty measure. *Journal of Sound and Vibration*, 201:85–101, 1997.

[11] K. Worden, G. Manson, and N.R.J. Feilelr. Damage detection using outlier analysis. *Journal of Sound and Vibration*, 229:647–667, 2000.

[12] K. Worden and G. Manson. The application of machine learning to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365:515–537, 2007.

[13] G. Manson, K. Worden, and D.J. Allman. Experimental validation of a structural health monitoring methodology: Part iii. damage location on an aircraft wing. *Journal of Sound and Vibration*, 259:365–385, 2003.

[14] K. Worden, G. Manson, G. Hilson, and S.G. Pierce. Genetic optimisation of a neural damage locator. *Journal of Sound and Vibration*, 309(3-5):529–544, 2008.

[15] L. Tarassenko. *Guide to Neural Computing Applications*. Elsevier, 1998.