Neural Networks applied to Structure Health Monitoring using autoregressive models as feature extractors

Felipe da Costa Pereira - (felipecostapereira@gmail.com)

PUC-RJ - ELE2389 - Redes Neurais - Trabalho Final

Professora: Marley Vellasco

July 6, 2022

Abstract

In Structure Health Monitoring, an important task is to detect damage patterns in a given physical system. Machine Learning techniques have been successfully applied in this area for this purpose. A critical step in a machine learning project is the feature extraction, as measured data acquired in these systems usually consist of important amount of timeseries information. This paper proposes a complete data processing pipeline, from feature extraction based on autoregressive models to state condition damage detection classification, using neural networks and comparing its preformance with other popular classification algorithms and based on a laboratory data acquisition assembly. This work also analyses a neural network based Kohonen Map as an unsupervised method to understand the feature space for the problem of damage detection in SHM.

1 Introduction

Structure Health Monitoring (SHM) consists of observing measured data from a given physical system in order to monitor properties and characteristics over time. Damage detection is one of the important objectives of this discipline, as it tries to detect, in advance, a failure event that can occur when the system is operating ou of its normal condition.

The measured data in this field consists of sensors response over time. The main approach to detect damage is try to select features from these data that are sensitive to damage but insensitive to operational and environmental variations ([Figueiredo et al., 2011]). In [Figueiredo et al., 2009], a complete discussion on several techniques for feature extraction can be found and the paradigms of feature extraction for strucutre health monitoring are widely discussed in [Worden et al., 2007].

Autoregressive models are a class of models which can be succesfully applied for feature extraction. The work [Figueiredo et al., 2011] analyses in detail the autoregressive (AR) timeseries models as feature extractor tools in order to detect damage patterns on a laboratory experiment. Such study also shows the importance of the model order as a key parameter to represent the patterns. According to the autors, the AR model coefficients are, themselves, sensitive features to the damage, since the appropriate model order is respected.

The approach of this case study is to use neural networks to identify damage in a controlled experiment, using the autoregressive (AR) models as feature exractors as discussed in [Figueiredo et al., 2011], and its coefficients as features. Then, we compare the results with other popular algorithms for classification.

2 Experimental Data

Several studies in SHM are conducted using standard datasets. In this study we will use a dataset consisting of measures from a three-floor structure assembled in the Los Alamos National Laboratory (LANL). The structure has different columns and plates connected by joints. Different damage scenarios are simulated by adjusting the gap between column and bumper in the middle of the top floor (figure 2). Environmental conditions are simulated varying the stiffness of some columns (figure 1) and adding mass to some plates (figure 2).

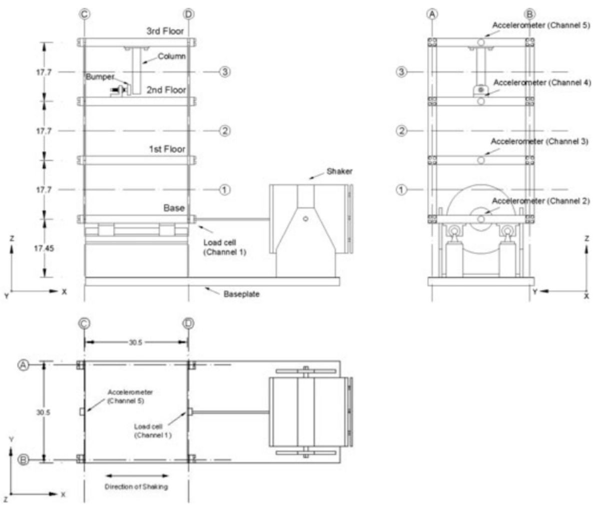


Figure 1: General overview of the three stage model

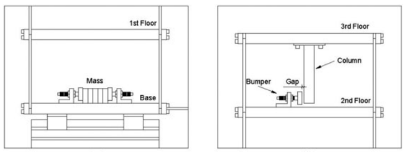


Figure 2: Sources of environmental and non-linear effects

The model is composed of four sensors (accelerometers) that register the vibration measures from the base and the three floors as seen in figure 1. For a given case, each channel accelerometer captures a set of 8192 datapoins correspondig to a time window of about 26 seconds. The details of the given structure and the data used in our case-study are described in [Figueiredo et al., 2009].

A total of 50 measures were made for each one of the 17 state conditions described in table 1, what gives a total of 850 observations in the dataset. The state conditions can be grouped as shown in table 2 according to the kind of stimulation imposed to the structure.

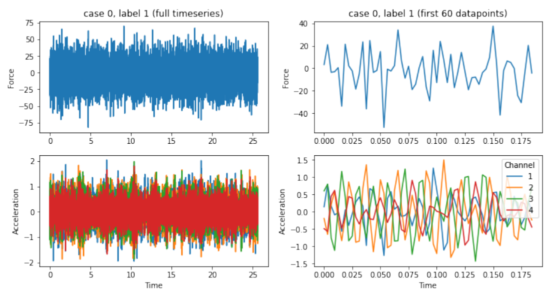
Table 1: Labels for the 17 state conditions

State	Condition	Description
#1	Undamaged	Baseline condition
#2	Undamaged	Mass=1.2kg(base)
#3	Undamaged	Mass=1.2kg(1st floor)
#4	Undamaged	87.5% stiffness drop (1BD)
#5	Undamaged	87.5% stiffness drop (1AD-1BD)
#6	Undamaged	87.5% stiffness drop (2BD)
#7	Undamaged	87.5% stiffness drop (2AD-2BD)
#8	Undamaged	87.5% stiffness drop (3BD)
#9	Undamaged	87.5% stiffness drop (3AD-3BD)
#10	Damaged	Gap=0.20mm
#11	Damaged	Gap=0.15mm
#12	Damaged	Gap=0.13mm
#13	Damaged	Gap=0.10mm
#14	Damaged	Gap=0.05mm
#15	Damaged	Gap=0.20mm, mass=1.2kg (base)
#16	Damaged	Gap=0.20mm, m=1.2kg (1st floor)
#17	Damaged	Gap=0.10mm, m=1.2kg (1st floor)

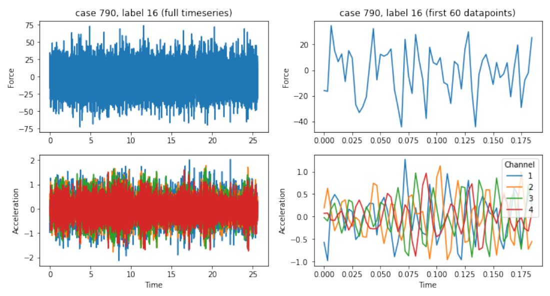
Table 2: Grouped classes

State	Group	Description
1	1	Baseline Condition
2,3,4,5,6,7,8,9	2	Environmental Change only
10,11,12,13,14	3	Damage only
15,16,17	4	Damaged/Environmental Change

Each case (observation) consists of the measures of 4 accelerometers composed by 8192 datapoints each. By concatenating the information from the four channels, the dataset is composed by 850 rows (cases) and (4 x 8192 = 32768) datapoints, which are considered as features. The timeseries for two of these observations (cases) are show in figure 3, where force and accelerations form the four measure channels are plotted. The plots in the right shows a detail of the first 60 datapoints either for the force and for the 4 accelerometers.



(a) Case (observation) #0, State Condition (class) #1



(b) Case (observation) #790, State Condition (class) #16

Figure 3: Measured timeseries for two of the 850 cases

3 Methodology

The Data Procesing Pipeline used in this study is shown in figure 4. The first step is to apply the autoregressive model to the measured data. As a high number of features is kept after applying the AR model, we use then a PCA to reduce the dataset dimensionallity. As a third step we make an exploratory analysis in order to get insights about the dataset and the classes distributions, followed by a data preparation step. The two final steps, model training and evaluation will be detailed in the next sessions, as we define the model validation protocol.

4 Feature Extraction - Auto Regressive Models

The AR(p) model, where p defines the order or number of parameters in the model, is developed from response time-series data $x_1, x_2, \cdots, x_n$ and it can be written as

$$x_i = \sum_{j=1}^{p} \phi_j x_{i-j} + \epsilon_i$$

where xi is the measured signal at the time instant ti. The parameters ϕ_j are fitted so that the residual in predicting the value of the timeseries x_i , using the previous values x_{i-j}, is minimized. This estimation of the unknown AR parameters, ϕ_j, can be done by using least-squares approach and the ϕ_j parameters can be used directly as damage-sensitive features. The appropriate AR model order is initially unknown. A higher order model may better fit the data, but may not generalize well to other data sets. On the other hand, a low-order model will not necessarily capture the underlying physical system dynamics [Figueiredo et al., 2011].

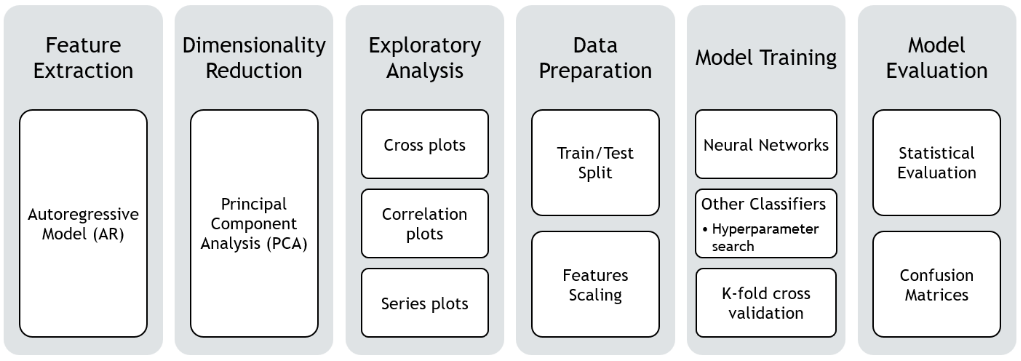


Figure 4: Data Procesing Pipeline

In this study we will use $p = 30$ as proposed by [Figueiredo et al., 2011] as being a good model order to work as a feature extractor for the database used. By fitting the AR(p=30) model for each one of the four channels we have 30 coefficients per channel. We apply the AR(30) model to each of the four channels timeseries of all the 850 cases in the dataset. Each channel 8192 datapoints are then transformed into the domain of the 30 AR coefficients. The mean value of the AR model parameters, computed in the 850 dataset observations is shown in figure 5, for the four accelerometers measures.

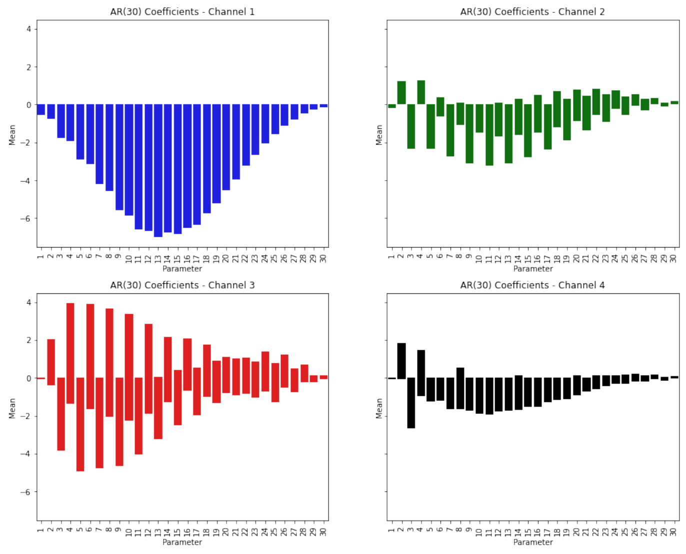


Figure 5: Mean value of the AR(30) coeffients, for each one of the four channels, computed over the 850 cases

To represent the complete set of measured data as being the complete set the 4 channels information, we then concatenate the 30 parameters for the four channels, wich gives a total of 120 features for each of the 850 cases in the dataset. Figure 6 shows a plot of the 120 composed feature set of the 850 measured cases.

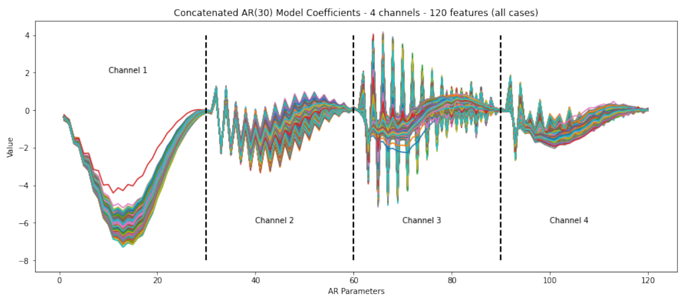


Figure 6: 850 cases x 120 feautures (AR Model parameters), 4 channels concatenated

At this point, the total number of features is still high to be used directly as a feature input space for any classification model, as it can lead to overfitting issues. In figure 7 we can see a correlation matrix between these 120 features. As high correlations are shown between some features,specially when it comes to channel 1 and 2 (first 60 parameters), we need to reduce the dimensionallity of this input space. To address this issue, we will apply a PCA analysis.

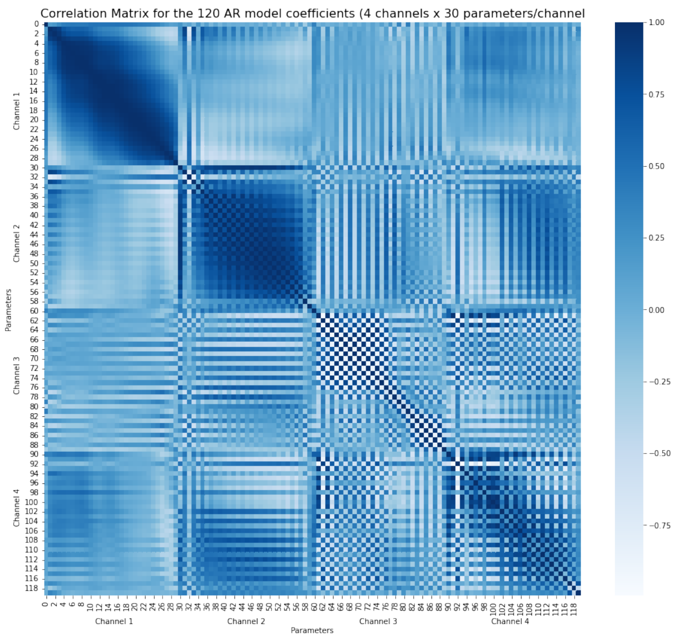


Figure 7: Correlation matrix between the features given by the AR Model

5 Dimensionality Reduction - PCA

Large or massive data sets are increasingly common and often include measurements on many variables. It is frequently possible to reduce the number of variables considerably while still retaining much of the information in the original data set. Principal component analysis (PCA) is probably the best known and most widely used dimension-reducing technique for doing this. PCA finds linear combinations of the original variables, called principal components, that successively have maximum variance for the data [Jolliffe, 2011].

In this study we apply a PCA transformation on the features provided by the AR Model. Figure 8 shows a relation between the number of PCA components and the explained variance of the data. For the next steps we choose to use the 6 first PCA components, which are responsable to represent 96% of the variance.

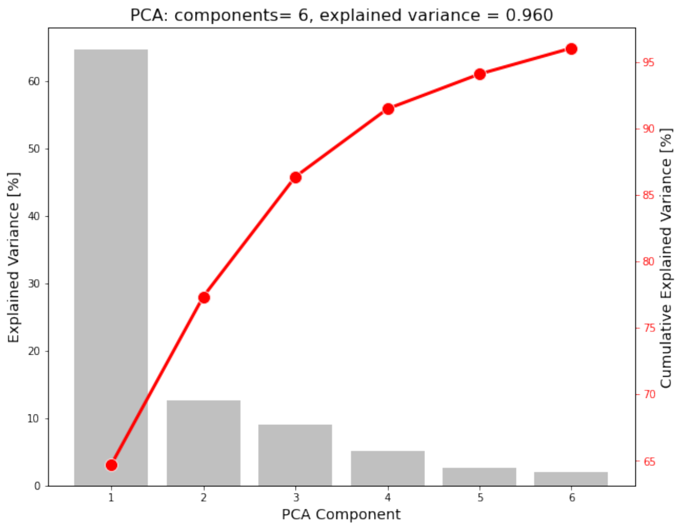


Figure 8: PCA explained variance sensibility to the number of components chosen

6 Exploratory Analysis

After transforming the data with the Principal Component technique, the feature space drops to six representative features. As shown in 9, the first PCA component carries information to distinguish some of the 17 state conditions. PCA components 2 and 3 also carry information to detect some labels but have more similar values between the classes. For the fourth PCA component, the 17 classes have similar values.

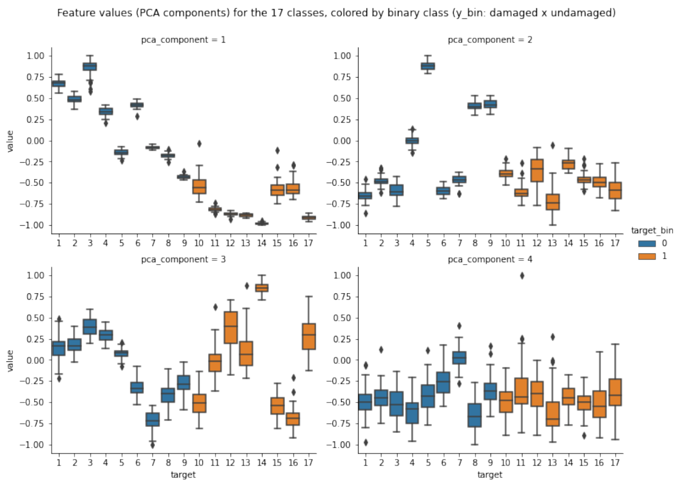
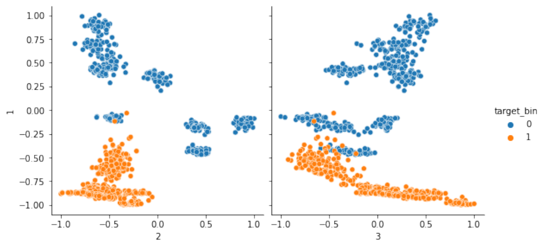
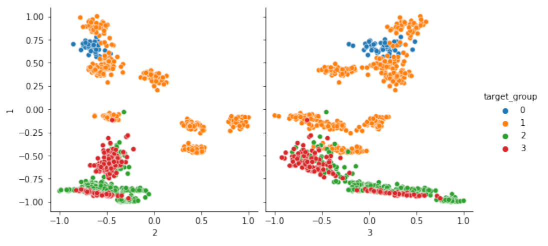


Figure 9: Distribution of the first four PCA component values for each one of the 17 state conditions

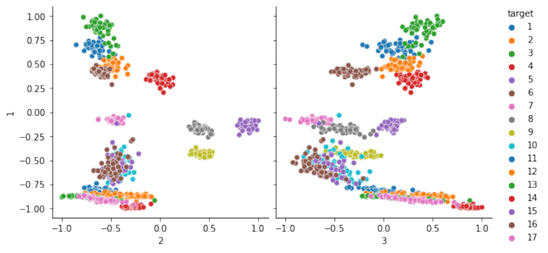
The cross plots in figure 10 show a relationship between components 1 and the other to most imporatant (2 and three). In the binary class problem (figure 10a), the classes seem to be linearly separable. On the other hand, for the 4 group and 17 state conditions classification, the classes are mixed in the space of features given by PCA components.



(a) binary classification



(b) four groups classification



(c) 17 state conditions

Figure 10: Crossplots between PCA components 1,2 and three for the three levels of classification

7 Classification

7.1 Neural Networks

A neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest: the network is usually implemented by using electronic components or is simulated in software on a digital computer. To achieve good performance, neural networks employ a massive interconnection of simple computing cells referred to as ”neurons” or ”processing units”. A neural network is a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use. The procedure used to perform the learning process is called a learning algorithm, the function of which is to modify the synaptic weights to attain a desired design objective. The modification of synaptic weights provides the traditional method for the design of neural networks [Haykin, 1999]

In this study, we'll train a Multi Layer Perceptron (MLP) neural network to predict the damage classes in the three store dataset using the backpropagtion method [Haykin, 1999]. In order to prevent the model from overfitting, we avoid to choose a very large number of neurons acording to the metric proposed in [Hetch-Nielsen, 1992]:

$$N_{hidden} <= 2N_{input} + 1$$

Other parameters were tested: number of hidden layers (one or two), learning rate, number of epochs, type of acivation function for the hidden processors. The neural network clasifier architecture (one relu activation hidden layer with 13 processors) was chosen over a 5-fold cross-validation as the model having the best score over the five folds.

As a second step, we used the selected model to measure its behaviour on different holdout sets by training that model once again for each of the ten train/validation/test splits as described in figure 11. The objective of this step is to provide a statistical assessment of the results by eliminating possible biases if a single holdout was used.

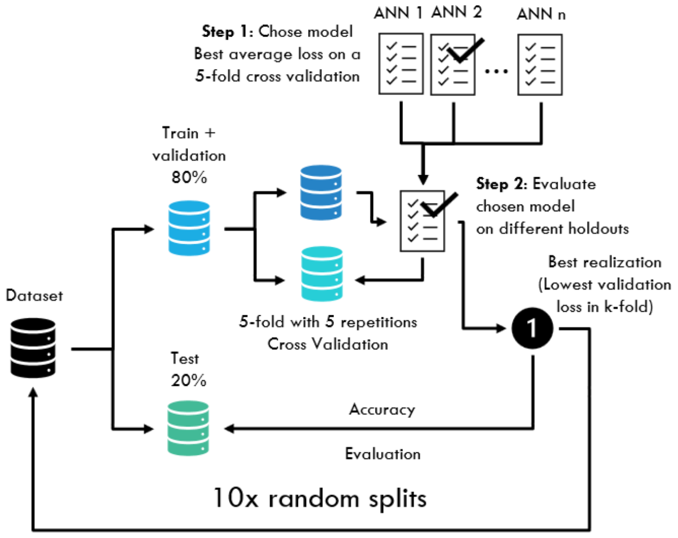


Figure 11: Neural network model selection and performance measure scheme

The selected architecture was: 6 input (PCA components), 13 hidden neurons in 1 hidden layer, softmax activation function for the output and relu for the hidden layer. Concerning the output, this architecture was tested in two situations: predicting the 17 damage states and predicting the 4 group (2). The procedure described in 11 was executed then, twice, with 17 and 4 outputs, respectivelly.

The confusion matrices (normalized by the true label), for these two problems, is shown in 12, for the best realization model over the 10 different splits. Although the model had a higher accuracy on predicting the four grouped stage conditions compared to the 17 classes, the results were quite similar in terms of accuracy.

7.2 Other Classifiers

As the neural network trained in the previous section could work as a good predictor either for the 17 state conditions or the 4 grouped states, another objecitve of this study is to compare its performance with other known classifiers. The most popular algortihms in machine learning were included for that purpose.

Since the neural network parameters (weights) learn through time with the backpropagation algorithm of trainning, we also provide a hyperparameter search for the soft classifiers, in order to tune them to get a more reliable comparison. A list of classifiers and the respecitve parameters used in hyperparameter search is found in table 3.

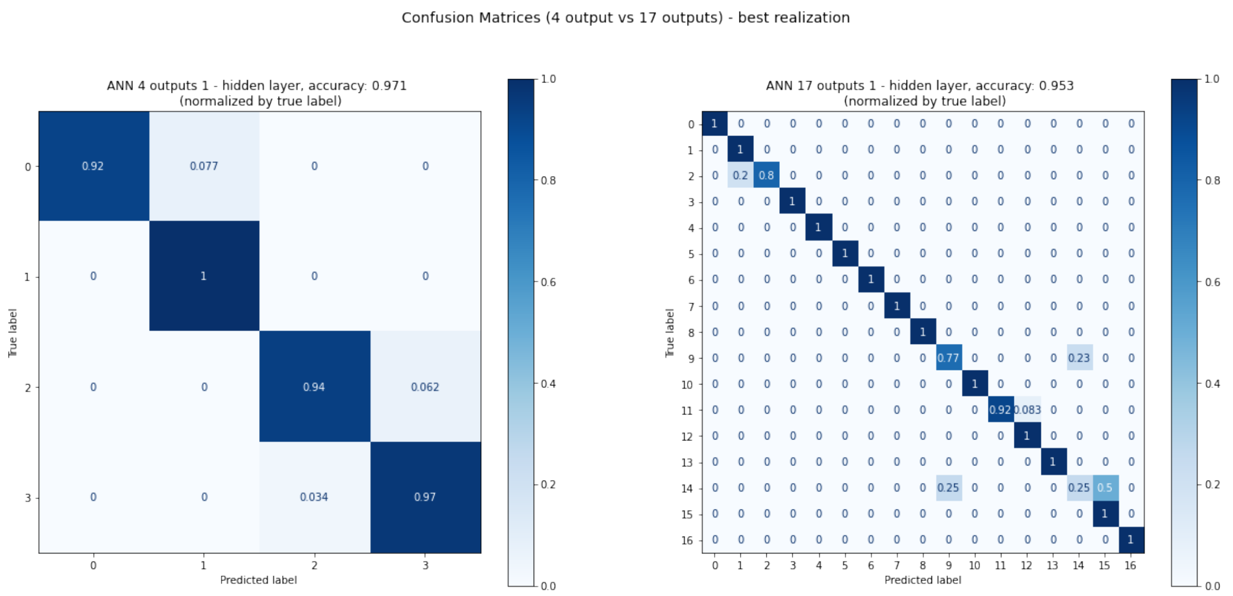


Figure 12: Neural network performance on predicting 4 groups (accuracy=0.971) or 17 classes (accuracy=0.953)

Table 3: Soft classifiers parameters used in hyperparameter search

Classifier	Parameter	Value Range
Gaussian Naive-Bayes	-	-
Logistic Regression	penalty	’l1’, ’l2’, ’elasticnet’, ’none’
	C	1-25
	solver	’newton-cg’, ’sag’, ’saga’, ’lbfgs’
SVM	C	1-25
	kernel	’linear’, ’poly’, ’rbf’, ’sigmoid’
	degree	1-10
K-nearest neighbors	n_neighbors	3-11
	algorithm	’auto’, ’ball_tree’, ’kd_tree’, ’brute’
	leaf_size	1-50
Decision Tree	criterion	’gini’, ’entropy’
	max_features	0-1
	max_depth	3-20
Random Forest	n_estimators	2-100
	criterion	’gini’, ’entropy’
	max_features	0-1
	max_depth	2-30
Adaboost	n_estimators	2-100
	learning_rate	0.1-20
Bagging Classifier	n_estimators	5-100
	max_samples	0.1-20
	max_features	0.1-8
Stacking Classifier	estimator	LogisticRegression, SVC, KNeighborsClassifier RandomForestClassifier

The model validation protocol, for the soft classifiers is described in figure 13. Ten random splits are made in the dataset and for each of them, a model is selected over a 100 realization parameter search for each one of the classifiers using a 5-fold with five repetitions cross validation. In order to keep the comparison with the neural network as reliable as possible, the same random seeds were used in both cases.

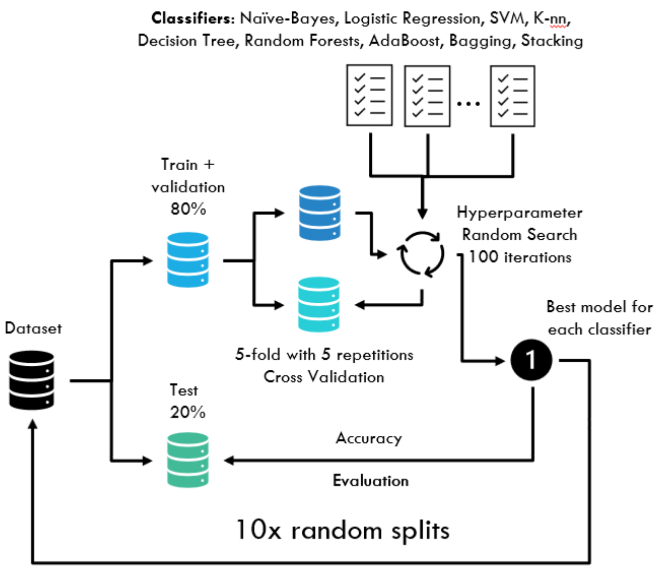

Classifiers: Naïve-Bayes, Logistic Regression, SVM, K-nn, Decision Tree, Random Forests, AdaBoost, Bagging, Stacking
Train + validation 80%
Hyperparameter Random Search 100 iterations
Best model for each classifier
Dataset
5-fold with 5 repetitions Cross Validation
Test 20%
Accuracy
Evaluation
10x random splits

Figure 13: Soft Classifiers model validation protocol with hyperparameters search

For each split, a model is selected as being the best parameter realization (of 100) in the 5x 5-fold cross validation and the best model (minimal validation loss) is then evaluated over the holdout set. By this procedure is possible to have the performance of the soft classifiers in ten different train/validation/test randomly generated splits.

A more complete reference on the soft classifiers algorithms tested in this work can be found in [Geron, 2017]

7.3 Classifiers Results

After training the neural network and performing a hyperparameter search for the soft models, the results over the 10 randomly generated tran/test splits is shown in 14. The average score is represented by the black dot.

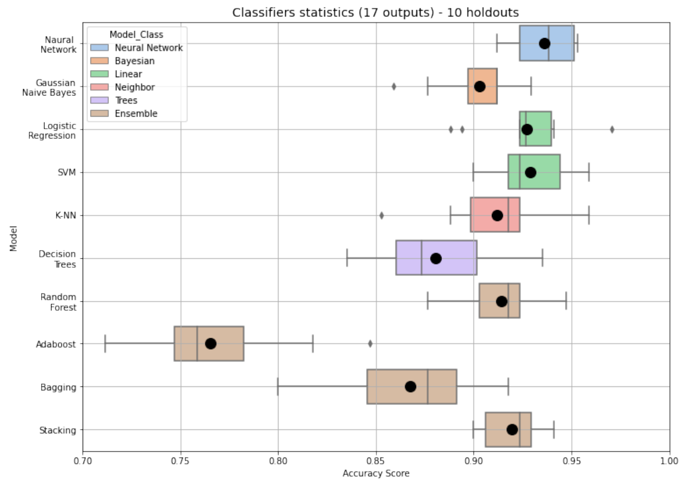


Figure 14: Artificial Neural Network and soft classifiers results on predicting the 17 state conditions

The neural network classifier had the highest score over the classification task when compared to the other models, although the methodology implemented was not exactly the same. The linear models (Support Vector Machines and Multiclass Logistic Regression - Softmax) also had a quite high accuracy when compared to the others algorithms.

Decision Trees and the Ensemble methods showed a higher variance on their predictions and the best model of these was the Stack classifier using LogisticRegression, SVC, k-Neighbors classifier and RandomForest Classifier as estimators, but each of them with their default parameters.

8 Kohonen Map

Kohonen or Self-Organizing Maps are a kind of neural network used to cluster the data into groups. Usually the neurons are organized in a 2-D so that the proximity in the input space is preserved in the 2D space after training. It's an unsupervised method that uses the technique of competitive learning and was proposed by [Kohonen, 2001].

The final step of this study is to apply a self organizing map to the input data. The self organizing map trained was a 12x15 neurons. The darker values in the map shown in figure 15a show neurons quite distant from their neighbors, what is interpreted as a separation between clusters. The map pattern shows two well separated regions, what is confirmed when we plot the labeled data in a binary level (damged x undamaged - figure 15b). The map is quite accurate in separating these two classes and detecting the presence of two groups of similar characteristics.

When we plot the labeled data with a more detail in classificatiom (4 groups or 17 state conditions), we see elements from the same class located in opposite positions in the map, what suggests that the map trained was not capable of separating groups in that level of detail (figures 15c and 15d).

Although we can see some coherence in figures 15c and 15d for some classes, the average quality of the map is poor.

9 Concluding Remarks

As the main conclusions of this study, we can mention:

1. AR models, when using an appropriate order, are indeed useful as a feature extractor for the problem presented, and the coefficients of that model are themselves features in the reduced space.

2. For the proposed problem, the PCA techinque was responsible for achieving a high comression in the feature space dimension, transforming a 120 dimension input space into a 6-component with only a few loss of information.

3. The neural network model as well as the other classifiers (specially the linear models) performed well in detecting damage in the studied laboratory structure, considering the pre-processing procedures implemented (feature extraction and dimensionallity reduction).

4. It's reasonable to think that the same neural network or any of the soft classifiers, if applied directly to the initial feature space would lead to a poor performance as classifiers, due to the huge amount of fetures, what shows the importance of the preliminary steps in a data processing pipeline.

5. Self-Organizing Map trained in this study was capable of capturing the groups at a high level (binary classification: damaged vs undamaged), but showed a weak performance when getting in more detailed classification degrees.

References

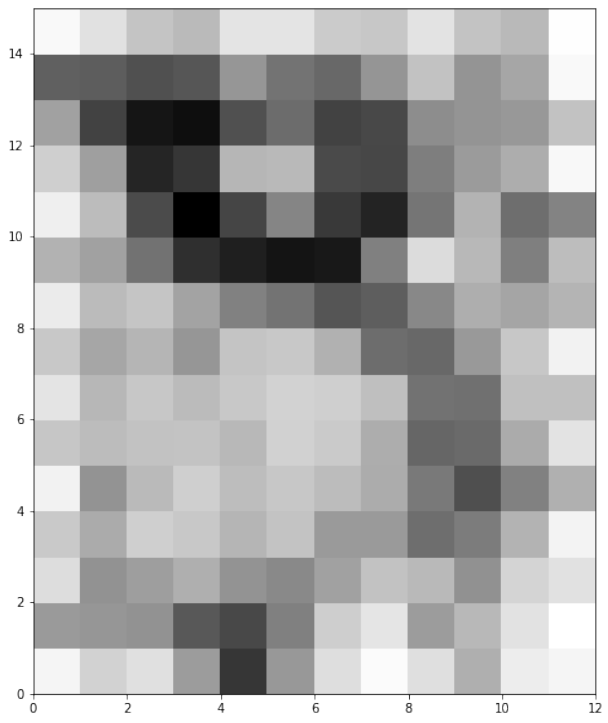
[Figueiredo et al., 2011] Figueiredo, E., Figueiras, J., Park, G., Farrar, C. R., and Worden, K. (2011). Influence of the autoregressive model order on damage detection. *Computer-Aided Civil and Infrastructure Engineering*, 26(3):225–238.

[Figueiredo et al., 2009] Figueiredo, E., Park, G., Figueiras, J., Farrar, C., and Worden, K. (2009). Structural health monitoring algorithm comparisons using standard data sets.

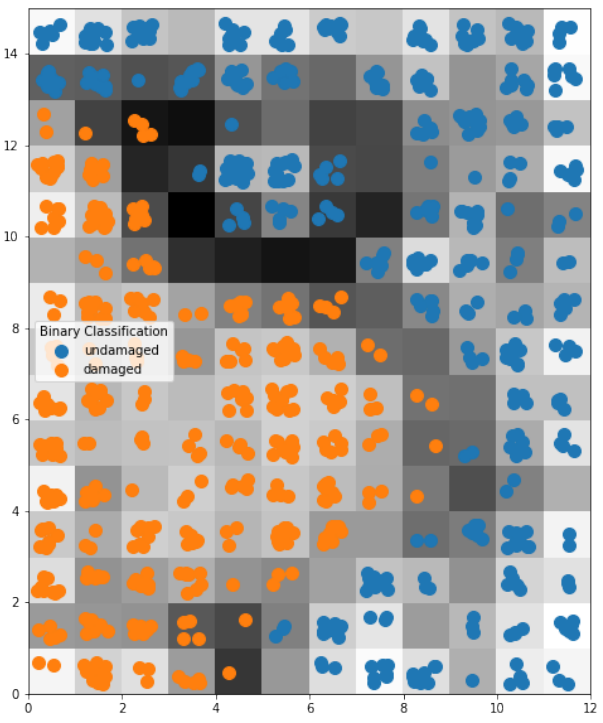
[Geron, 2017] Geron, A. (2017). *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, Sebastopol, CA.

[Haykin, 1999] Haykin, S. (1999). *Neural Networks and Learning Machines.* Prentice Hall.

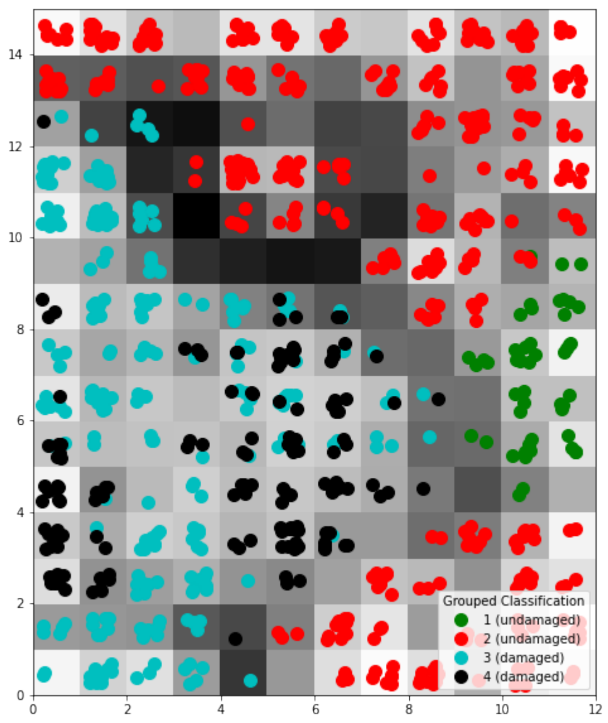
[Hetch-Nielsen, 1992] Hetch-Nielsen, R. (1992). *III.3 - Theory of the Backpropagation Neural Network**Based on "nonindent" by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.* Academic Press.



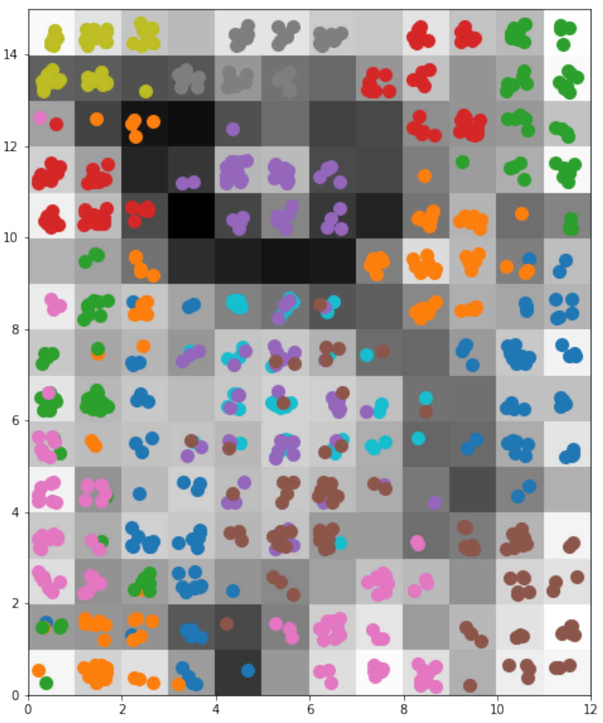
(a) no label



(b) Binary classification: damage x undamaged



(c) Grouped States



(d) 17 state conditions

Figure 15: Self organizing map on background (a) and target labels plotted (b,c,d)

[Jolliffe, 2011] Jolliffe, I. (2011). *Principal Component Analysis*, pages 1094–1096. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Kohonen, 2001] Kohonen, T. (2001). *Self-organizing maps*. Springer series in information sciences, 30. Springer, Berlin, 3rd edition.

[Worden et al., 2007] Worden, K., Farrar, C., and Manson, G. (2007). The fundamental axioms of structural health monitoring. *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences*, 463:1639–1664.