International Journal of Artificial Intelligence and Machine Learning
Volume 11 • Issue 2

An Integrated Process for Verifying Deep Learning Classifiers Using Dataset Dissimilarity Measures

Darryl Hond, Thales, UK

Hamid Asgari, Thales, UK

https://orcid.org/0000-0002-9317-7045

Daniel Jeffery, Thales, UK

Mike Newman, Thales, UK

ABSTRACT

The specification and verification of algorithms is vital for safety-critical autonomous systems which incorporate deep learning elements. The authors propose an integrated process for verifying artificial neural network (ANN) classifiers. This process consists of an off-line verification and an on-line performance prediction phase. The process is intended to verify ANN classifier generalisation performance and to this end makes use of dataset dissimilarity measures. The authors introduce a novel measure for quantifying the dissimilarity between the dataset used to train a classification algorithm and the test dataset used to evaluate and verify classifier performance. A system-level requirement could specify the permitted form of the functional relationship between classifier performance and a dissimilarity measure; such a requirement could be verified by dynamic testing. Experimental results, obtained using publicly available datasets, suggest that the measures have relevance to real-world practice for both quantifying dataset dissimilarity and specifying and verifying classifier performance.

KEYWORDS

Artificial Neural Network, Classifier Resilience Function, Dataset, Dissimilarity Measures, Machine Learning Classifiers, Performance Evaluation, Verification and Validation

1. INTRODUCTION

Autonomous systems make use of a suite of algorithms in order to understand the environment in which they are deployed and make independent decisions. These algorithms typically solve one or more classic problems, such as classification and prediction. Artificial neural networks (ANNs) are one such class of algorithms, which have shown great promise in view of their ability to learn complicated patterns underlying high-dimensional data. The decision boundary approximated by such networks is highly non-linear and difficult to interpret, which is particularly problematic in cases where these decisions can compromise the safety of either the system itself, or people. Furthermore,

DOI: 10.4018/IJAIML.289536

the choice of data used to prepare and test the network can have a dramatic impact on performance and, in consequence, safety.

Verification and validation (V&V) are vital parts of the development and deployment of any engineering system. V&V processes are well established in more mature sectors of engineering such as aerospace and automotive. However, they are not as well developed in areas such as autonomy and machine learning (ML), and the broader field of artificial intelligence (AI). Since ML technologies are being more widely adopted, it is ever more important that they behave as expected, and interact safely with people. Our focus is on the verification of ANNs when used for image classification in safety-critical systems.

Systems are verified with respect to the specified requirements. One such requirement for a classifier might state a necessary level of classification performance, and this requirement can be verified by dynamic testing. However, it might be the case that such a requirement does not specify any properties of the test dataset. If a test dataset provides only a modest classification challenge to a network, then a high-level of classification performance does not mean that the network will operate well during operation. An additional condition needs to be specified i.e. the properties of the test dataset used to evaluate the classification performance. For example, the test dataset might be characterized in terms of its relation to the dataset used to train the classifier, or in terms of its noise content, or in terms of the intrinsic separability of its component classes. System requirements addressing discriminative capability could then state the permitted form of a function mapping test dataset properties to classifier performance. If these requirements are specified and verified, we can have a degree of confidence that the classifier will perform at a certain level in an operational mode when applied to input instances of a certain type.

This paper introduces a measure and its variants that can be used to quantify the dissimilarity between a test dataset and a training dataset. This dissimilarity will henceforth be termed ‘dataset dissimilarity’. Classifier performance for a particular test dataset might itself be measured in terms of accuracy for example. If so, classifier accuracy can then be given as a function of this dataset dissimilarity measure i.e. each test dataset is assigned a dataset dissimilarity value, and this quantity will map to an accuracy value. This in turn allows system-level requirements to be formulated in terms of the required relationship between performance and the test dataset dissimilarity measure. If such a requirement is verified, evidence has been gathered that a classifier will perform at a certain level when applied to test datasets; there will be a greater level of confidence that a classifier will generalise as required to data which is dissimilar to the training dataset.

The contribution made by the study reported in this paper is, firstly, the introduction of a novel measure which gauges the dissimilarity between a test dataset and a training dataset. This measure adopts and extends some of the concepts reported in DeepGauge on testing criteria (Ma et al., 2018). Secondly, we demonstrate that the measure can be used to determine the relationship between test dataset dissimilarity and classifier performance. Thirdly, we investigate the suitability of the MMD, an established measure, for gauging test dataset dissimilarity and thereby predicting classifier performance. Finally, we propose an integrated process for the verification of ANN classifier generalisation performance. Dissimilarity measures play a key role within this verification process. The outputs of the verification process presented in this paper have “cross-domain usage” across many industries including maritime, transportation, and aviation.

The remainder of this paper is structured as follows. We briefly introduce some important V&V concepts derived from software engineering, and then review the relevant literature. We then discuss our contribution to the field, before presenting and analysing our experimental work. We next introduce a novel integrated verification process for ANN classifiers that describes how the dissimilarity measures are applied. Thereafter the paper examines how the approach described could be extended and finally draws conclusions.

2. KEY VERIFICATION AND VALIDATION CONCEPTS

Verification can be defined as: 'methods by which confidence can be gained in the correctness of a system with respect to its specification' (Hond, White, Asgari, 2020). In traditional software engineering, corner cases are a key aspect of verifying the correctness of a program's behaviour through dynamic testing. A corner case is a state in which several factors reach the edge of their operating or behavioural range (each being an edge case) simultaneously (e.g. when several program inputs achieve their maximum or minimum values) (Chico, 2021). The idea of a corner case naturally transfers to ML input data. Here, corner cases can be considered inputs that are outliers with respect to the training data. As ML algorithms enter deployment, we need to be able to state and verify the expected, or mandated, prediction or classification performance for corner cases.

There are a number of reasons why an input would be an outlier, and therefore a corner case, with respect to the training data. ML training and test dataset instances correspond either to points in an input space or, after feature extraction, to points in some feature space. One of the key assumptions of ML systems is that the training and test data is independently and identically distributed within the input space. This is referred to as the independent and identically distributed assumption (Chung, et al., 2018), (Vapnik, 1999). In practice, this means that the two datasets (test and train) must be randomly sampled from the same source, usually a single set of data. However, this assumption will often not hold when ML models are deployed to the real world because of domain shift (Kouw, 2018), (Tsymbal, 2004).

3. RELATED WORK

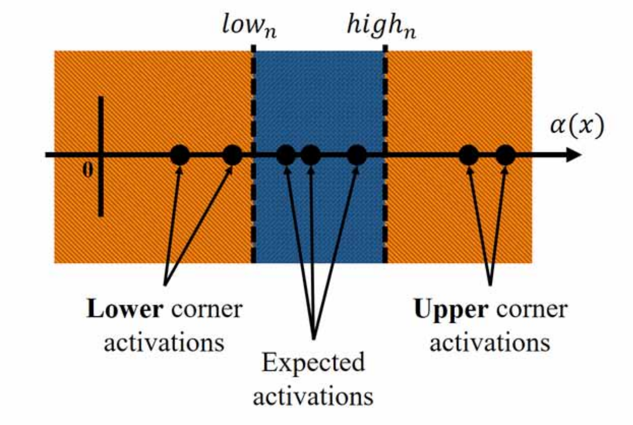
In (Asgari, et al., 2019), a review of selected aspects of Robotics and Autonomous Systems (RAS) from different sectors is covered. This review includes the defined level of autonomy, the technological and regulatory aspects, and current verification, validation, certification and assurance (VVCA) aspects. The current VVCA are mainly based on the standards and regulations for the safety and security of systems that are composed of deterministic functions. The incorporation of ML in RAS will require new techniques which can deliver reliable and resilient adaptation and learning, and new formal and dynamic verification methods which can be applied to non-deterministic autonomy algorithms.

Many studies addressing the formal verification of ANN classifiers have examined the extent to which test instances can be perturbed without yielding a change in the assigned class (Weng, 2018). This seam of research was a response to the observation that images which have been correctly assigned to a particular class by an ANN classifier, are sometimes assigned to an alternative, incorrect class when subject to minor modifications. While it is often the case that small perturbations happen by chance, a specially generated noise pattern can be generated which causes a specific misclassification. Such generated noise is termed adversarial (Szegedy, et al., 2013) and might be produced for malicious reasons.

Metamorphic Testing (MT) has been used for the dynamic testing of ANN classifiers (Spieker,et a., 2020). MT is a software testing method which makes use of metamorphic relations (MR). An MR is a property of the function under test which can be used to generate metamorphic test cases. In general, an MR will state the expected relationship between the outputs produced by a system for a set of inputs, where those inputs are related by systematic transforms; MT establishes whether the MR holds. For example, there are a number of transformations which can be applied to an image, such as rotation, which do not change its assigned ground truth class (if the extent of the transform is sufficiently limited).

The field of ANN classifier dynamic testing has also been concerned with identifying adversarial images. In (Ma et al., 2018), (Tian, et al., 2018), (Pie, et al. 2017), the authors propose test coverage metrics which assess the extent to which neural networks are exercised by test datasets.

Figure 1. The definition of upper and lower corner activation values for a given neuron, based on the maximum ($high_n$) and minimum (low_n) activation values recorded for that neuron over the set of training images



Adding adversarial images to a test image dataset tends to increase neuron coverage, as empirically demonstrated by the authors of DeepGauge (Ma et al., 2018).

The metrics proposed by the DeepGauge paper are all based on the range of activation values of the neurons within the deep neural network (DNN) under test. The paper defines a number of neuron coverage (NC) measures. At the core of the formulation of these measures are two complementary concepts: the major function region (MFR) and the corner-case region of a neuron. Suppose that a set of images is used to train a network to completion. If this same training set is then submitted to the now fully trained network, each of the member images will generate an activation value for each neuron. The MFR of a neuron is the interval between the highest and lowest activation values generated for that neuron by the set of training images. The corner case region of a neuron is then defined as the complement of the MFR: it is the set of values outside of that interval, and within $-\infty$ and $+\infty$. When a test image is submitted to a network, each neuron will produce an activation value which will fall within either its MFR or its corner case region. This is illustrated in Fig. 1. This conception is in keeping with the earlier discussion, in that a corner-case is considered an outlier. In Fig. 1, the blue region spans the major function region, i.e. the range of neuron output values observed when processing the set of training images. By definition, only test images can produce neuron output values outside of the major function region. The axis has a zero marked to the left because it is being assumed that the neuron is a rectified linear unit (ReLU).

The practical usefulness of NC criteria has been questioned (Kim, 2019), because of the aggregation of information over an entire test dataset, with the loss of detail about specific test inputs. A more practical measure should provide information about inputs on an individual basis. The authors of (Kim, 2019) propose Surprise Adequacy for Deep Learning (SADL) that measures how surprising a test input is with respect to the data the network was trained on. They introduce two variations: the Likelihood-based Surprise Adequacy and the Distance-based Surprise Adequacy. The likelihood-based method uses Kernel Density Estimation to estimate the probability distributions of the neuron activations during training. This estimated probability distribution is then used to determine the level of surprise of new inputs, for example, during testing. The second method measures the Euclidean distance in neuron space between the test input and two training inputs each from a different class.

Another algorithm which compares the neuron activations generated by training and testing data is Deep k-Nearest Neighbors (DkNN) (Papernot, 2018). Whilst not strictly intended for Deep Learning (DL) system testing, DkNN calculates the nonconformity of a test activation with respect to training activations. The nonconformity value is used as part of a conformal prediction calculation to assign both a confidence and credibility value to a given prediction.

A range of statistical distance measures for estimating the safety of machine learning classifiers are discussed in (Aslansefat, 2020). The measures in (Aslansefat, 2020) are not specific to a certain type of machine learning classifier, such as ANNs. The authors of (Ali, et al., 2018) employ similarity measures to enhance the classification of heterogeneous data. Data instances are assigned to groups on the basis of the similarity of their feature values, and customized classifiers are applied to each group.

The Maximum Mean Discrepancy (MMD) (Gretton, et al., 2012) is an established non-parametric measure for comparing distributions. It has been applied to sets of neural network activation values to measure the difference between real and synthetic imagery (Seo, et al. 2018), and between domain-shifted datasets (Tzeng, 2014). The MMD was applied to raw data in (Liang, 2017) in order to determine the distances between in- and out-of-distribution datasets.

4. A NOVEL MEASURE OF TEST DATASET DISSIMILARITY

An ML classifier requirement might state the required form of the relationship between classification performance and some quantitative property of a test dataset. This paper defines such a quantitative and informative dataset property: a value returned for a test dataset which measures its dissimilarity to a training dataset.

We have formulated a novel distance which returns a value for a given test instance with respect to a particular training dataset. We have also defined a normalized form of this distance. Our novel dataset dissimilarity measure is the median of this normalized distance over a test dataset. In this paper, these measures have only been applied to ANN classifiers used for image classification. However, they can be used for other forms of input data, and further ANN architectures and applications.

As per DeepGauge (Ma et al., 2018) let N = {n1, n2, …} be the set of neurons which make up a trained DNN and ϕ(x, *n*) be a function which returns the output of neuron *n* given network input x, e.g. an image. For a neuron *n*, let high_n and low_n be the upper and lower bounds of the set of values returned by ϕ(x, *n*) for all inputs in the training dataset. [low_n, high_n] is the MFR of n.

4.1. Neuron Region Distance

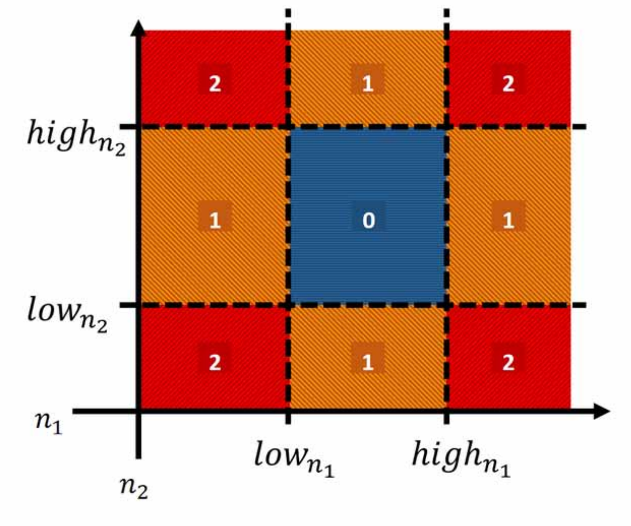
Our core, novel distance is termed the Neuron Region Distance (NRD), and is based on the MFR of the ANN neurons (Hond, Asgari, Jeffery, 2020). The NRD of a given input is defined as the number of neurons which do not activate inside their MFR. From (Ma et al., 2018) a network is entirely in its MFR for input x iff $\forall$ n $\in$ N: ϕ(x; n) $\in$ [low_n; high_n]. The NRD of an input x is therefore defined as:

$$NRD\left(x\right) = \left|\left\{n \in N : \phi\left(x,n\right) \notin \left[low_n, high_n\right]\right\}\right|$$

Note that the NRD is returned for each input x, rather than for a whole dataset. A simplified graphical representation of the NRD is shown in Fig. 2. With only two neurons, n_1 and n_2, the maximum NRD is two, which is assigned when both neurons are activated outside their MFR.

If the MFR for each neuron is mapped to 0 and the corner case region is mapped to 1, the neuron outputs produced by an input x can be represented by a binary vector, with as many components as there are neurons in the network. A training input would be represented by a vector of zeros. For this representation, the NRD can be treated as a Hamming distance between the binary vectors generated for x and any training input (as portrayed in Fig. 2). In Fig. 2, training inputs, by definition, produce neuron output values within the MFRs of both neurons (indicated by the central zone) and have an NRD of 0. An input which activates one neuron within its MFR and the other neuron outside of its

Figure 2. A representation of the NRD associated with activation values generated within a network with just two neurons, n_1 and n_2



MFR has an NRD of 1. If the input produces activation values external to the MFRs of both neurons it is assigned an NRD of 2.

4.2. Fractional Neuron Region Distance

The NRD can be normalized to produce a Fractional Neuron Region Distance (fNRD). The fNRD is calculated as the proportion of neurons in a network which are activated outside the MFR by a given input x:

$$fNRD\left(x\right) = \frac{\left|\left\{n \in N : \phi\left(x,n\right) \notin \left[low_{n}, high_{n}\right]\right\}\right|}{\left|N\right|}$$

fNRD values can be interpreted as indicating the normalized extent to which test instances differ from a training dataset; the values for an entire dataset can be used to generate a measure of test dataset dissimilarity. Since the measure is normalised, the fNRD returned for an input can be compared across different ANNs, where these networks might vary in their total neuron count.

4.3. Median Fractional Neuron Region Distance

Our dataset dissimilarity measure is the median of the set of fNRD values returned for an entire test dataset: the median fNRD. We show that the median fNRD of a dataset relates monotonically to classification performance, and thus helps add context to a quoted performance figure (such as accuracy or recall).

- Proposed Median fNRD Usage: Test Coverage

Our novel distances and measure can be used as additional analytical tools for test coverage assessment, for example in addition to those offered by DeepGauge.

- Proposed Median fNRD Usage: Qualifying Performance evaluation

Suppose that a neural network is trained with an image dataset, and that multiple test datasets are prepared. By finding the fNRD of each image, the median fNRD can be found for each test dataset. We suggest two potential behaviours in the context of image classification as described below:

Behaviour Type 1: If the performance of the network for each test dataset significantly decreases as the test dataset's median fNRD increases, then the network is not generalising well to more distant data.
Behaviour Type 2: If performance is stable across a range of median dataset fNRD values, then classifier generalisation is relatively good.

If Behaviour Type 1 is observed, and the system specification requires better generalisation than observed, then this indicates that the network requires further retraining in order to meet the specification. If Behaviour Type 2 is observed, and the range of median dataset values with acceptable classification performance is greater than that found in the specification, then this provides evidence towards the correct operation of the network.

5. INVESTIGATION OF FRND APPLICABILITY

Having established the median fNRD measure, an experimental investigation is required to assess its effectiveness for placing performance evaluation in the context of test dataset dissimilarity. The Research Questions (RQ) to be addressed are as follows:

RQ1: Is there a positive relationship between the median fNRD of a test dataset and some alternative established measure of the dissimilarity of the test dataset to the training dataset?
RQ2: Can the median fNRD be employed to predict the ability of an ANN classifier to generalise?

5.1. Research Question 1: Analysis

An ANN classifier employed in a real-world system, perhaps an autonomous vehicle, is likely to receive input data during operational deployment which is not drawn from the same distribution as the dataset used to train the network. It is also likely that operational data which differs from the training data will be semantically different, in ways perceptible to a person. Further, it can be assumed that the greater the dissimilarity between the operational data and the training data, the greater the extent to which classification performance will deteriorate. The median fNRD should therefore reflect semantic differences between training and operational (or test) datasets if it is to be of utility for measuring real-world, practical dataset dissimilarity. In order to investigate whether the median fNRD shows this desired property, a series of datasets were generated from a source test dataset, where each dataset in the series was treated with additive noise of progressively increasing variance.

The use of additive noise in this manner has four advantages: a) additive noise variance is a controllable measure of dataset dissimilarity, b) additive noise of progressively greater variance produces a semantic, perceptible change in imagery, c) metamorphic testing can be applied to images with added noise, since it is known that, given the variance is not too great, the ground truth of an image will not be altered, d) it is known that noise distortion typically reduces performance for both machine vision algorithms and humans alike (Geirhos, et al., 2017).

The nature of the relationship between the median fNRD and this alternative measure of dataset dissimilarity, the variance of additive noise, was experimentally determined. An increasing monotonic relationship between the median fNRD and the variance of additive noise would provide evidence that the median fNRD is an effective indicator of significant test dataset dissimilarity, and that it might reflect real-world, noticeable changes between datasets. This activity was intended to establish the measure's potential as a practical tool. In addition, the relationship between the degree of additive noise and classifier performance was relevant to the study.

5.2. Research Question 2: Analysis

The second research question is to establish that the median fNRD of a dataset can be used to predict the ability of an ANN to generalise under real-world conditions. Our approach was to determine the relationship between classification performance and the median fNRD for multiple test datasets.

6. APPLYING THE MMD TO NEURON ACTIVATION VALUES TO MEASURE DATASET DISSIMILARITY

6.1. The MMD

In addition to an experimental analysis of the median fNRD (a novel measure), an investigation was undertaken to determine the suitability of the maximum mean discrepency (MMD) (an established measure) for gauging dataset dissimilarity. The MMD is computed on the basis of the neuron activation values produced by each dataset instance. The median fNRD is also a function of neuron output values.

An input x supplied to a network will produce an activation value for each neuron. The activation values generated by a single input over the network can therefore be represented by a vector with $\left|\mathrm{N}\right|$ components, with each component being assigned the corresponding neuron activation value. Let *m* be the number of training images and *n* be the number of test images processed by a network. When supplied to the network, the images in the training dataset will produce a series of vectors $a_i : i = 1, \cdots, m$. Similarly, the images in the test dataset will generate a series of vectors $b_i : i = 1, \cdots, n$.

Following (Gretton, et al., 2012), the MMD between the training and test dataset is found, for some kernel *k*, as:

$$MMD^2 = \frac{1}{m\left(m-1\right)}\sum_{i=1}^{m}\sum_{j\neq i}^{m}k\left(a_i, a_j\right) + \frac{1}{n\left(n-1\right)}\sum_{i=1}^{n}\sum_{j\neq i}^{n}k\left(b_i, b_j\right) - \frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}k\left(a_i, b_j\right)$$

Therefore, in this work, the quadratic time version of the MMD is being calculated.

6.2. Determining the Bandwidth of Gaussian Kernel

The MMD is computed for the experiments undertaken using a Gaussian kernel as below:

$$k\left(x, y\right) = \exp\left(-\frac{\left\|x-y\right\|^2}{2\sigma^2}\right)$$

The value of the MMD is linked to the bandwidth of the Gaussian function, σ. A method for calculating sigma is given in (Liang, 2017). The method is based on finding the distance between all inputs and then taking the median of the distances. This was the method we adopted for computing sigma.

6.3. Benchmarking the Median fNRD With the MMD

If the MMD is regarded as a standard means of comparing distributions, then it can be used to benchmark the ability of the median fNRD to measure dataset dissimilarity. Therefore, as well as being assessed on its own merits as a tool for classifier generalisation verification, the MMD served as a point of reference for the investigation.

7. EXPERIMENTAL METHOD

The experiments described in this section were designed to answer the research questions.

7.1. Datasets

Our investigations centred around two well-known datasets: 1) the original Modified National Institute of Standards and Technology dataset (MNIST), which features images of handwritten digits (LeCun, 1998), and 2) Fashion MNIST (Xiao, et al., 2017), which features images of items of clothing. We refer to the MNIST data as the “Base” dataset. Each dataset is divided into 10 classes.

7.2. Building and Training Classifiers

The LeNet5 convolutional neural network (CNN) architecture (LeCun, et al. 1998) was used to build classifiers for both the Base MNIST and Fashion MNIST datasets. This choice was made to allow comparison of our computed DeepGauge-based statistics with those reported in the original paper (Ma et al., 2018). This architecture is also relatively lightweight, with 60K weight and bias parameters, and therefore fast to train, even without access to intensive GPU processing.

In order to stabilise computed statistics, five class-balanced training, validation and test splits were drawn in the ratio 819:81:100 to generate five trained networks for each dataset. A performance metric and median fNRD score was calculated for each of the five trained instances of the networks when they were supplied with their allocated test data. Finally, the mean of the five performance metrics and the mean of the five median fNRD scores were determined.

Training was carried out using the ADAM (Kingma, 2014) optimization routine and all networks were implemented using Keras 2.3.1 with a Tensorflow 2.0 backend. Hyper-parameters and learning rate schedule were set as per the original LeNet5 paper (LeCun, et al. 1998). All experiments were run on a computer with an i7-7820X with 16GB RAM and a GTX 1080 Ti, running Windows 10.

To standardise training for the networks, early-stopping was implemented based on the accuracy computed on a validation dataset. The criterion for halting training was an inter-epoch difference in validation accuracy of $< 0.5\%$. The model with the highest observed validation set accuracy was then retained.

7.3. Test Datasets

Two source test datasets were drawn from the Base and Fashion MNIST datasets respectively and derivative datasets were generated by adding additive white Gaussian noise. Fourteen further test datasets were produced, seven for each of the two source datasets, by adding noise of progressively greater variance. The variance of the additive noise for the two source datasets can be considered to be zero.

7.4. The Effects of Additive Noise Variance on the Median fNRD of Test Datasets

This and the following section outlines the two experiments used to investigate RQ1. The first experiment for RQ1 was designed to determine the effects of increasing the variance of additive noise on the median fNRD value returned for a dataset. After fully training the network, the training data was re-input to the network, and the relevant MFR boundaries were recorded. Upon presentation of the augmented testing datasets to the network, the fNRD of each image was measured. The median fNRD value was then calculated for each test dataset.

7.5. The Effect of Additive Noise Variance on Classifier Performance

The second experiment recorded the effects of additive noise on classifier performance. The experimentation process was the same for both the Base and Fashion MNIST datasets, and was repeated five times in order to improve the statistical estimates. The classifier was applied to each of the noisy datasets. The measure of classification performance used was accuracy.

7.6. The Effect of the Median fNRD on Classification Performance

RQ2 was addressed by determining the relationship between classifier performance and the median fNRD. The global performance achieved for a dataset was measured in terms of accuracy.

7.7. Applying the MMD as a Dataset Dissimilarity Measure

The last two experiments were repeated using the MMD as a dissimilarity measure in place of the median fNRD.

8. RESULTS AND DISCUSSION

8.1. Effects of Additive Noise Variance on Classifier Performance

Fig. 3 shows the effects of the eight different additive noise variances on the accuracy of the classifier. The dark round points correspond to the Base MNIST and the light crosses to the Fashion MNIST datasets. Each graphed accuracy measurement is an average, calculated over five repeats, as discussed above. The plots in Fig. 3 show that as the additive noise increases in strength, accuracy degrades.

8.2. Effects of Additive Noise Variance on Median fNRD

As illustrated by Fig. 4, the median fNRD appears to increase with greater additive noise variance, and this is observed for both MNIST datasets. This makes intuitive sense i.e. adding noise shifts the

Figure 3. Accuracy achieved for Base MNIST (round) and Fashion MNIST (crosses) datasets as a function of additive Noise Variance

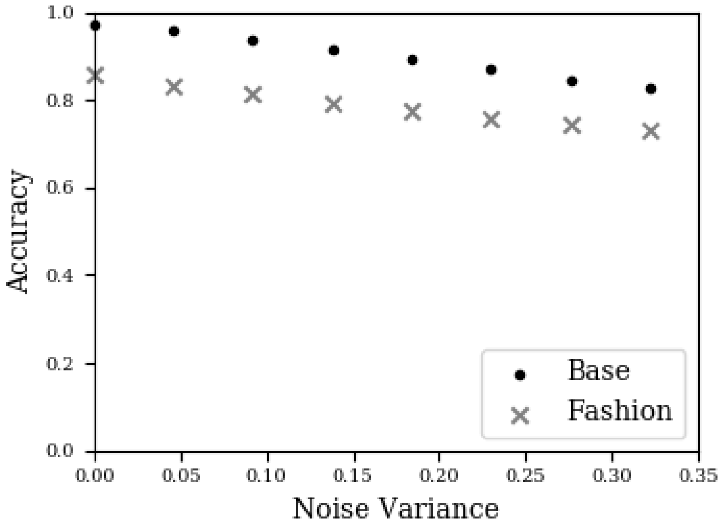
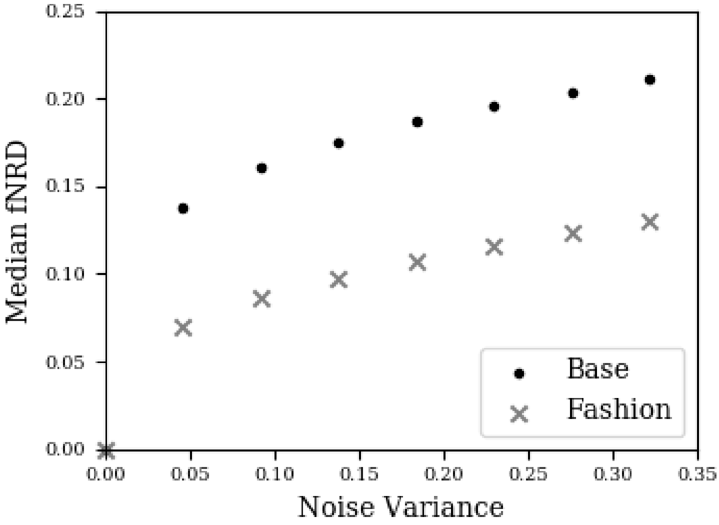


Figure 4. Variation in Median fNRD per dataset against the variance of additive noise applied to the datasets



distribution of the test images away from that of the images used to train the network. This establishes, in this particular case, that the median fNRD has an increasing monotonic relationship with additive noise variance.

8.3. Effect of Median fNRD on Classification Performance

Fig. 5 suggests that the relationship between classifier performance (accuracy) and the median fNRD score is monotonic and decreasing. This is the case for both MNIST datasets. These quantitative relationships capture the ability of the networks to generalise to dissimilar test datasets.

8.4. Effects of Additive Noise Variance on the MMD

Fig. 6 shows that the relationship between additive noise variance and the MMD is monotonic for both the MNIST and Fashion MNIST datasets. These trends mirror those observed for the median fNRD. For the test datasets with no added noise, the MMD is close to zero for both Base and Fashion MNIST datasets.

8.5. Effect of the MMD on Classification Performance

Again, the results displayed in Fig. 7 for the MMD parallel those seen for the median fNRD: performance steadily decreases as the MMD increases for both MNIST datasets.

8.6. Data Dissimilarity Due to Additive Noise Compared to Real-World Data Variation

The experiments which have been conducted have employed additive noise to render test datasets dissimilar to training datasets. However, there are many possible modes of variation between training data and real-world operational data. For example, real-world data received during operation could be intercepted and subject to systematically designed adversarial modifications. These experiments may therefore not be sufficient to establish the comprehensive behaviour of the median fNRD and MMD measures when applied to test or operational data whose dissimilarity is not wholly due to additive noise.

Figure 5. Accuracy scores for the Base MNIST (solid round) and Fashion MNIST (dashed crosses) datasets as a function of Median fNRD

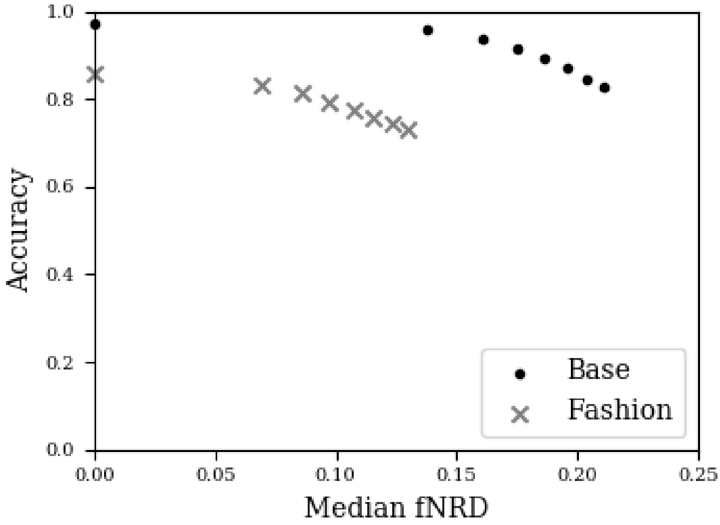
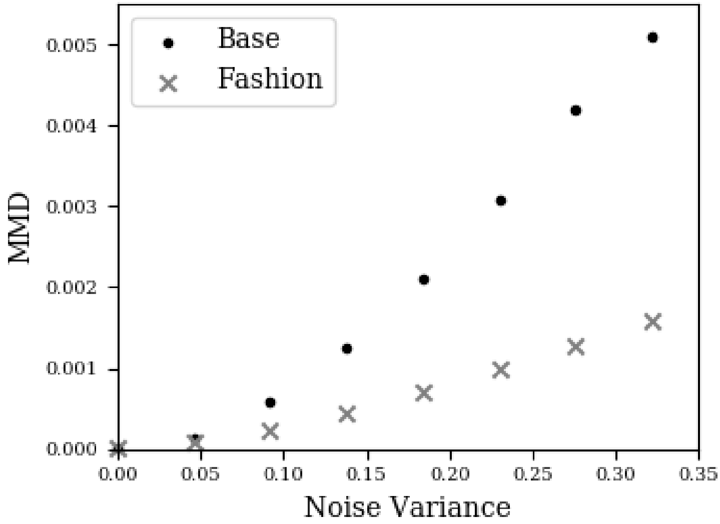


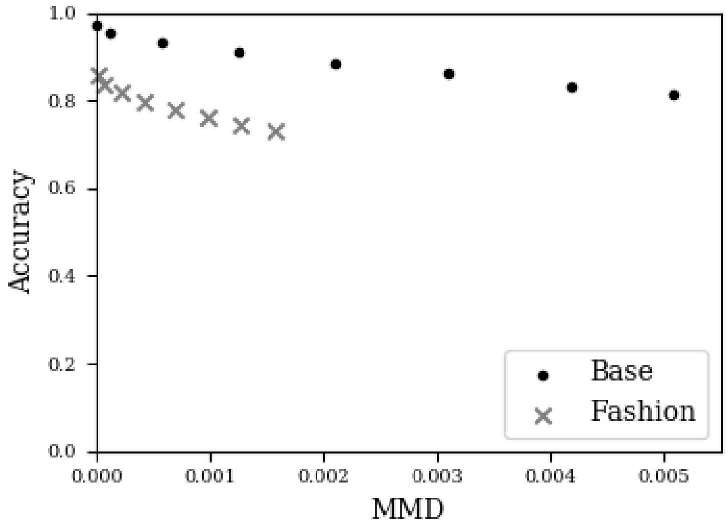
Figure 6. Relationship between MMD and Noise Variance



8.7. The MMD in Relation to the fNRD

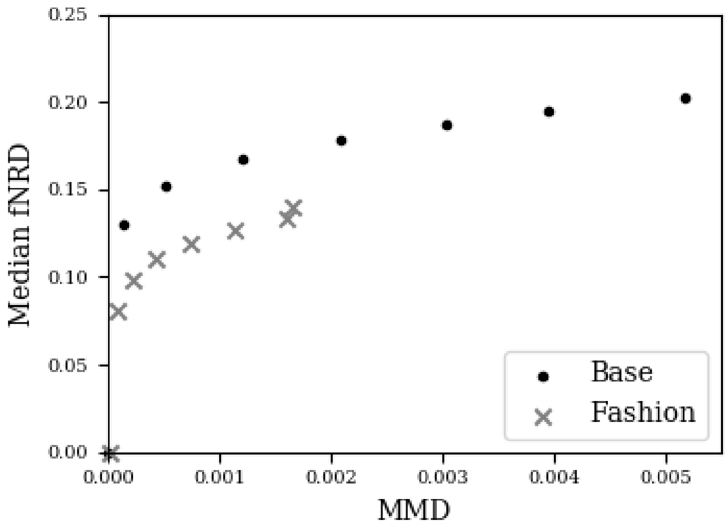
The results in Fig. 4 and Fig. 6 suggest that there is a monotonic relationship between the median fNRD and the MMD. However, the results presented in these two figures were obtained for different random splits of the data into training and test sets. Further evidence that a monotonic relationship holds is given in Fig. 8, which presents a plot of the median fNRD against the MMD for values generated for the same training and test datasets. Each Base MNIST point corresponds to one of a series of datasets, generated

Figure 7. Classifier Accuracy as a function of MMD



by adding noise of progressively greater variance to a source Base MNIST dataset. The same applies to the Fashion MNIST points. The noise variance values are equal to those found in Fig. 4 and Fig. 6. As with all results presented in this paper, the points in Fig. 8 are generated by averaging over five repeated experimental runs, as described in the experimental method. Fig. 8 shows that datasets to which noise of a greater variance has been added return correspondingly greater values for both the median fNRD and the MMD. Since the MMD has been allocated the role of a benchmark, this monotonicity indicates that the median fNRD can operate as an effective measure of dataset dissimilarity.

Figure 8. Relationship between MMD and fNRD



The MMD, especially in its linear time form (Gretton, et al., 2012), is conducive to datastream processing. Therefore, the MMD is a candidate for possible on-line deployment as a dataset dissimilarity measure. Given its low computational complexity, it is also highly feasible that the median fNRD could be put to on-line operational use. Moreover, in contrast to the MMD, which is generally found between two sets of multiple instances, the NRD and fNRD are defined for individual test instances (in relation to a set of multiple training instances). It follows that an fNRD value can be computed as each input is received by a classifier during operation, and this in turn should allow rapid and scalable dissimilarity calculations, enabling on-going self-verification and performance predication.

The NRD and fNRD also provide explicit information about the occurrence of neuron activation values outside of neuron MFRs, whereas the MMD does not record such a form of analytical detail, nor offers such insight. There might be advantages to applying multiple measures to gauge dataset dissimilarity, for example using both the median fNRD and the MMD. It could be the case that a measurement obtained by processing the outputs of a set of measures in some way, would prove superior to that returned by any individual measure.

8.8. The NRD and Test Coverage

The NRD is based on neural network properties used to define DeepGauge test coverage metrics. Statistics describing the NRD or fNRD could be used as further test coverage metrics in the DeepGauge suite, since they provide additional information about the extent to which neurons are exercised by test datasets. Similarly, there is a dual to the NRD which is informative: the number (or proportion) of test instances in a dataset which activate a neuron outside of its MFR.

9. PROPOSED INTEGRATED VERIFICATION PROCESS FOR ANN CLASSIFIER GENERALIZATION CAPABILITY

Here, we propose an integrated ANN classifier verification process. The scheme comprises an off-line verification component, and an on-line performance prediction component. Verification relies on dynamic testing rather than formal means. In particular, the process addresses ANN classifier generalisation capability.

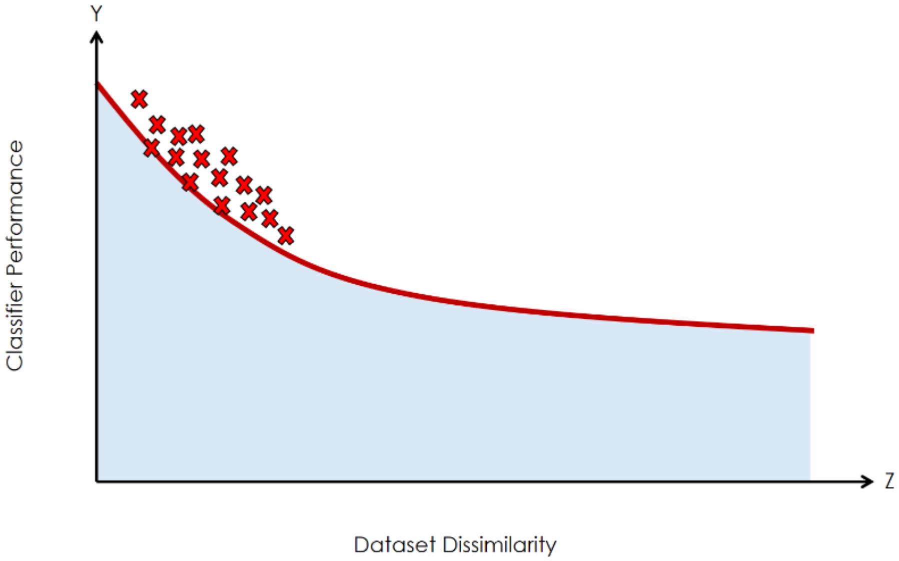
9.1. The Classifier Resilience Function (CRF)

The relationship between performance and dataset dissimilarity plays a key role within both of the verification process components. In general, this relationship might not take the form of a function, but instead be statistical in nature. In the latter case, a function can be derived from the statistical mapping. Fig. 9 illustrates how a function could be fitted to a statistical plot such that it returns near worst-case performance for a given dissimilarity. Each red cross in the figure represents a dataset. The red CRF curve serves to illustrate how a function could be fitted to the set of crosses so as to capture the lower limits of performance.

Any function that maps dissimilarity to performance will be termed a Classifier Resilience Function (CRF). Fig. 5 and Fig. 7 display examples of CRFs. A CRF can be used to specify and verify performance off-line, and to predict performance on-line. More generally, a CRF captures classifier performance as a function of some quantifiable property of a dataset. The gradient of a CRF can be referred to as the resilience of the classifier for the property quantified. A multivariate CRF would take values returned for multiple dissimilarity measures as arguments.

CRFs can be specified, when stating a requirement, or empirically measured, when verifying the requirement. The performance that a classifier is required to attain as a function of dataset dissimilarity is specified by means of a CRF, or a family of CRFs. The CRF produced during the verification process for a classifier, the empirically measured function, would have to belong to the specified family of CRFs if it is to be verified. Fig. 10 shows synthetic functions constructed for

Figure 9. A stylised example of fitting a CRF to a statistical relationship between performance and dissimilarity



illustration, with an example of a specifying and a measured CRF. In this instance, the specifying CRF takes the form of a horizontal line with the constant value $Y_{min,}$ and this gives a lower bound on the family of acceptable CRFs. The full specification is that a measured function must belong to the set of functions whose performance values are at least equal to the specified lower-bound CRF for dissimilarity values less than or equal to some threshold $Z_{th.}$ Such a requirement can be stated as: the classifier X must perform at a level >= Y_{min} for dataset dissimilarity <= $Z_{th.}$ The second (discrete) function displayed in Fig. 10 represents a function based on measurements made during a classifier verification procedure. Since the measured function lies above the minimal, specifying CRF for dissimilarity values <= Z_{th}, the classifier has been verified for this particular requirement.

9.2. The Full Verification Process

The full verification process (Fig. 11) commences with a stakeholder identifying requirements for classifier behaviour. These requirements are then restated as technical requirements, expressed in terms of one or more CRFs. Using the example given above, it might be the case that a classifier X must perform at a level >= Y_{min} for any test or operational dataset which has a dissimilarity <= Z_{th} relative to the training dataset. The level of performance could be measured in terms of accuracy for example. Under these conditions, a technical requirement might specify a minimally acceptable CRF (being a lower bound on the family of acceptable CRFs) which has the form shown in Fig. 10.

The next stage of the process is to train the classifier by means of a training dataset in the standard manner. After training, various properties of the trained network are recorded in preparation for the subsequent verification stage. When the NRD is being used as the dataset dissimilarity measure, the MFR would be established for all (or a subset of) relevant neurons.

The off-line verification stage consists of a comparison of specified and measured CRFs: the classifier is verified if the measured CRF has a form belonging to the specified family of CRFs. If the classifier fails to be verified, there are a number of options: the classifier can be retrained using a modified training dataset; the requirement can be changed - for the example above, Y_{min} or Z_{th}, or both, could be altered; and a modified or alternative classifier could be chosen. If the set of test datasets do not cover the dissimilarity range required for verification, in that the maximum

Figure 10. A stylised example of a specifying CRF (blue, continuous line) and a CRF which has been measured for verification (red crosses)

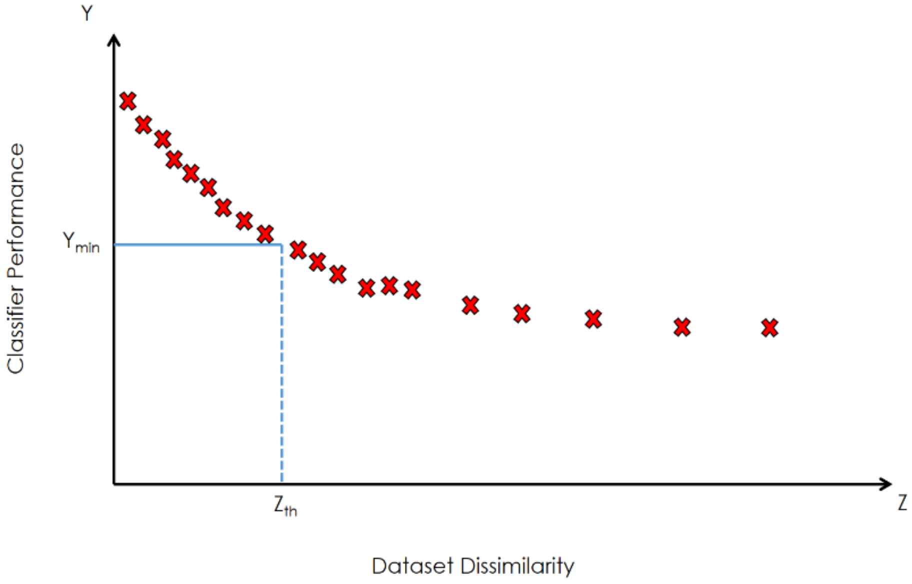
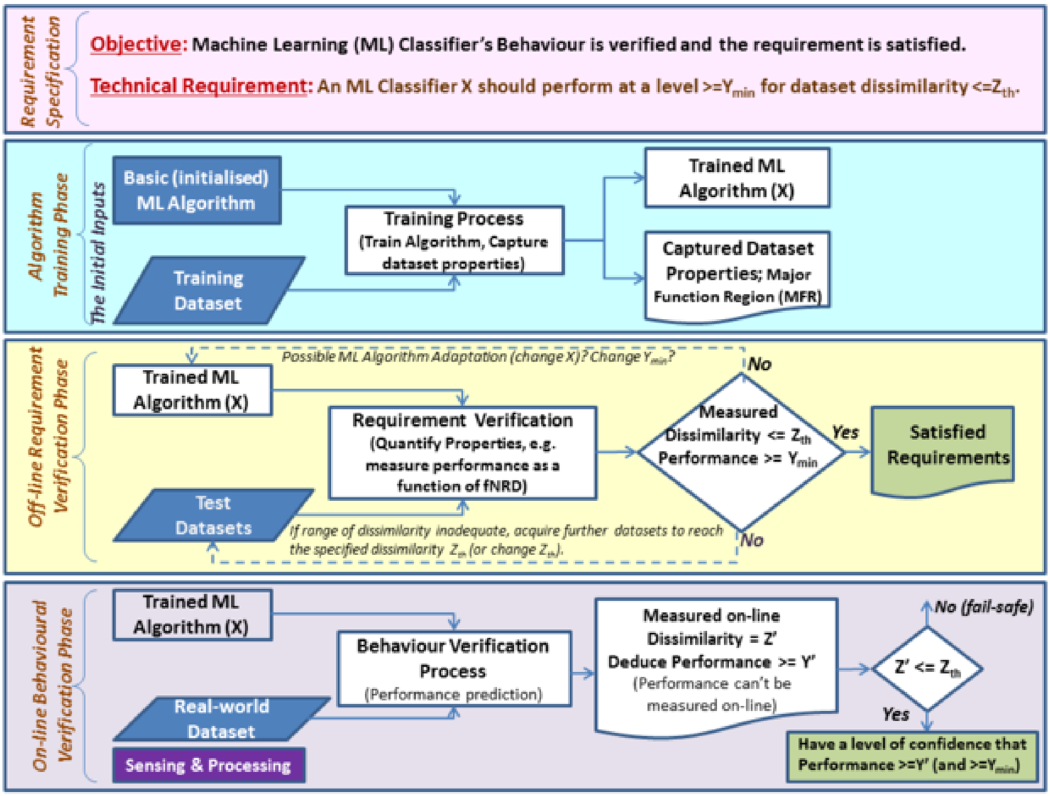


Figure 11. Classifier Verification Process Block Diagram


Requirement Specification
Objective: Machine Learning (ML) Classifier's Behaviour is verified and the requirement is satisfied.
Technical Requirement: An ML Classifier X should perform at a level >=Y min for dataset dissimilarity <=Z th.
Algorithm Training Phase
The Initial Inputs
Basic (initialised) ML Algorithm
Training Dataset
Training Process (Train Algorithm, Capture dataset properties)
Trained ML Algorithm (X)
Captured Dataset Properties; Major Function Region (MFR)
Off-line Requirement Verification Phase
Possible ML Algorithm Adaptation (change X)? Change Y min?
Trained ML Algorithm (X)
Requirement Verification (Quantify Properties, e.g. measure performance as a function of fNRD)
Test Datasets
Measured Dissimilarity <= Z th Performance >= Y min
No
Yes
No
Satisfied Requirements
If range of dissimilarity inadequate, acquire further datasets to reach the specified dissimilarity Z th (or change Z th).
On-line Behavioural Verification Phase
Trained ML Algorithm (X)
Behaviour Verification Process (Performance prediction)
Real-world Dataset
Sensing & Processing
Measured on-line Dissimilarity = Z' Deduce Performance >= Y' (Performance can't be measured on-line)
Z' <= Z th
No (fail-safe)
Yes
Have a level of confidence that Performance >=Y' (and >=Y min)

dissimilarity observed for a dataset is significantly less than Z_{th}, then datasets whose dissimilarities extend to Z_{th} must be sought.

The CRF measured and recorded during the off-line verification stage can also be used for the final stage of the verification process: on-line performance prediction. This would require measuring the dissimilarity of data received during operation relative to the training dataset. Furthermore, it might be essential that the dissimilarity could be continually recalculated in real-time. These computed dissimilarity values would then be mapped to performance values by means of the CRF as measured off-line. This predicted performance can then be made available to downstream processing modules as necessary. A system can be engineered to respond accordingly to the level of on-line performance predicted. Returning to our example, and the case of a classifier which has been verified off-line, if the measured on-line dissimilarity Z' is below $Z_{th,}$ then the performance Y' will be predicted to be greater than or equal to Y_{min}. In this case, the system response might be set to be proportionate to Y'. However, if the dissimilarity exceeds Z_{th} then another course of action such as a fail-safe response might be required.

It is feasible to measure dissimilarity on-line, and to update the measured value in real-time. The low computational complexity of the NRD makes it a candidate real-time dissimilarity measure.

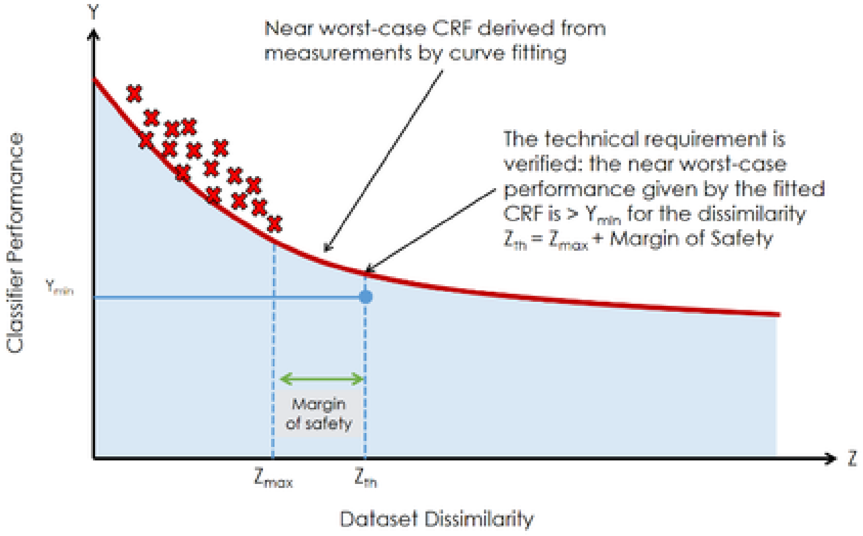
9.3. Mapping Stakeholder Requirements to Technical Requirements

The topic of deriving technical requirements from stakeholder requirements is complex. In order to give a perspective on how stakeholder requirements might flow down to technical requirements, an example will now be given. This will be with respect to the off-line component of the full verification process.

Suppose that an ANN classifier is being trained for operation in an autonomous road vehicle. The classifier must identify on-road and roadside objects in a city environment. Training and test data is collected for several cities, where the test data is presumed to have been drawn from the same distribution as the training data. The classifier achieves a performance of, say, 97% for the test data. A stakeholder-level requirement on classifier generalisation capability might have been stated in the form: 'The classifier must operate in all unseen cities with a performance of >=95%'. Note that stakeholders might well express classifier requirements in keeping with their own operational needs, viewpoints and linguistic terms; there is no reason to expect stakeholders to be familiar with the concept of dissimilarity measures, let alone typical values returned for such measures. Therefore, stakeholder-level requirements, such as the example quoted above, will need to be mapped to a technical requirement.

This mapping could be achieved in the following way. First, test data is collected from a number of unseen cities (locations from which no data has previously been gathered). This data would comprise one or more datasets associated with each city. For each dataset, the classification performance and corresponding dissimilarity measure value are found. The stakeholder requirement has specified a value for Y_{min}. The corresponding Zth value for the requirement is now derived from the scatter graph formed by plotting the set of measured (classification performance, dissimilarity measure value) points. In Fig. 12 each red cross corresponds to an unseen city dataset point. First, the maximum dissimilarity Z_{max} value is determined for the set of points. Next, a margin of safety is added to Z_{max} to give Z_{th}. The margin of safety, a dissimilarity value interval, is introduced because of the uncertainty that the unseen city data represents the range of all possible unseen cities. Methods of calculating the margin of safety will not be covered here, but it would be at least partially based on the range of dissimilarity values returned for the unseen city datasets for which data has been compiled. At this point, a technical requirement can be stated in terms of the values established for Y_{min} (95%) and Z_{th}: 'Classifier performance must be >= Y_{min} for dissimilarities <= Z_{th}'. The blue line in Fig. 12 is a CRF which is the lower bound for the family of CRFs which satisfy the above requirement. This technical requirement can now be subject to verification. A near worst-case curve is fitted to the scattered points to produce an empirically based CRF (shown in red). The technical requirement demands

Figure 12. Verification of a CRF against a technical requirement derived from a stakeholder requirement



that this near worst-case curve should return performance values >= Y_{min} for dissimilarities <= Z_{th}. The figure shows that this technical requirement has been verified: the near worst-case performance given by the fitted CRF is >= 95% for dissimilarities less than or equal to Z_{th}=Z_{max}+Margin of Safety.

When the training and test datasets are compiled for the seen cities, they will need to be representative of factors which influence the nature of the captured imagery, such as the type of buildings, roads and vehicles found in a city, or the climate. The intention would be to sample the variation exhibited by these identified factors, both individually and in combination, so that an effective classifier can be developed. However, it is difficult to avoid bias, and so the factors referred to might not be sampled adequately with respect to the set of unseen cities. It is for this reason that the classifier performance for the unseen cities might fall below 97%, as anticipated by and accepted by the stakeholder. However, there might be further factors which are not present or active when the training data is collected, but whose subsequent impact could lead to the generation of test or operational imagery dissimilar to the training data. The possible presence of novel factors which could influence unseen data is one reason for introducing a margin of safety.

10. FUTURE WORK

So far, we have examined datasets which are frequently used by the ML community, using Base MNIST and Fashion MNIST as the source of training and test data. Applying the dissimilarity measures we have examined to more extensive public or industrial datasets would be the next step, and would allow us to check whether our results hold for more applications. Although imagery and CNNs have featured in this study, the measures can be applied to any form of input data, and other ANN architectures. We also intend to apply other transformations to source datasets, in the same manner that additive noise was applied in the set of experiments described. These would include further image processing transformations such as rotation and blurring.

We anticipate that classifier performance will degrade as the dissimilarity between the training and test datasets increases. Computational issues aside, the most useful dissimilarity measure is the one that is most effective at predicting this degradation. Finding an optimal dissimilarity measure would support its application in the verification processes described.

11. CONCLUSION

The Assuring Autonomy International Programme (AAIP) is developing a Body of Knowledge which will serve as a reference for the safety assurance of autonomous systems (Hawkins, 2019). Our proposed verification approach addresses several assurance objectives within the AAIP document. More generally, novel approaches for verifying the correctness, performance, and behaviour of ANN classifiers will raise levels of confidence in their robustness and safe operation, and in their suitability for real-world deployment. Our contribution, a proposed integrated ANN classifier verification process, and an assessment of the potential of novel and established dissimilarity measures for requirement specification, is a step towards this end.

In this paper, the neuron region distance (NRD) and fractional neuron region distance (fNRD) have been introduced as dissimilarity measures. Experiments conducted on two image databases have shown that classification accuracy is a monotonically decreasing function of the median fNRD. A similar trend was also observed for the MMD.

An acceptable relationship between classification performance and the median fNRD, or the MMD, could be specified in a requirement. An integrated verification process has been proposed which makes use of such performance specifications for off-line verification. The process then exploits the verified relationship for on-line performance prediction.

ACKNOWLEDGMENT

This work is fully funded by Thales UK, Research, Technology and Innovation (RTI). The authors would like to thank their colleagues at RTI, especially Ben Pritchard.

REFERENCES

Ali, N., Neagu, D., & Trundle, P. (2018). Classification of heterogeneous data based on data type impact on similarity. In *UK Workshop on Computational Intelligence* (pp. 252-263). Springer.

Asgari, H., Farrell, J., & Pritchard, B. (2019). *Review of regulatory issues of robotic and autonomous systems: Learning for civil nuclear industry*. Engineering Safe Autonomy, 27th of Safety Critical Systems Symposium, Bristol, UK.

Aslansefat, K., Sorokos, I., Whiting, D., Kolagari, R. T., & Papadopoulos, Y. (2020). SafeML: Safety Monitoring of Machine Learning Classifiers Through Statistical Difference Measures. In *International Symposium on Model-Based Safety and Assessment* (pp. 197-211). Springer. doi:10.1007/978-3-030-58920-2_13

Chico State's Software Engineering. (2021). *Software testing - edge and corner cases*. https://github.com/ChicoState/SoftwareEngineering/wiki/Software-Testing-Vocabulary

Chung, Y., Haas, P. J., Upfal, E., & Kraska, T. (2018). *Unknown examples & machine learning model generalization*. arXiv preprint arXiv:1808.08294.

Geirhos, R., Janssen, D. H., Schütt, H. H., Rauber, J., Bethge, M., & Wichmann, F. A. (2017). *Comparing deep neural networks against humans: object recognition when the signal gets weaker*. arXiv preprint arXiv:1706.06969

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, *13*(1), 723–773.

Hawkins, R. (2019). *Body of knowledge - structure and scope*. https://www.york.ac.uk/assuring-autonomy/

Hond, D., Asgari, H., & Jeffery, D. (2020). Verifying Artificial Neural Network Classifier Performance Using Dataset Dissimilarity Measures. *19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 115-121. doi:10.1109/ICMLA51294.2020.00027

Hond, D., White, A., & Asgari, H. (2020). *Quantifying dataset properties for systematic artificial neural network classifier verification*. Assuring Safe Autonomy, 28th of Safety Critical Systems Symposium, York, UK.

Kim, J., Feldt, R., & Yoo, S. (2019). Guiding deep learning system testing using surprise adequacy. In *IEEE/ACM 41st International Conference on Software Engineering (ICSE)* (pp. 1039-1049). doi:10.1109/ICSE.2019.00108

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

Kouw, W. M., & Loog, M. (2018). *An introduction to domain adaptation and transfer learning*. arXiv preprint arXiv:1812.11806.

LeCun, Y. (1998). *The MNIST database of handwritten digits*. http://yann.lecun.com/exdb/mnist/

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. doi:10.1109/5.726791

Liang, S., Li, Y., & Srikant, R. (2017). *Enhancing the reliability of out-of-distribution image detection in neural networks.* arXiv preprint arXiv:1706.02690.

Ma, L., Juefei-Xu, F., Zhang, F., Sun, J., Xue, M., Li, B., Chen, C., Su, T., Li, L., Liu, Y., & Zhao, J. (2018). Deepgauge: Multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (pp. 120-131). doi:10.1145/3238147.3238202

Papernot, N., & McDaniel, P. (2018). *Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning.* arXiv preprint arXiv:1803.04765.

Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). Deepxplore: Automated whitebox testing of deep learning systems. In *Proceedings of the 26th Symposium on Operating Systems Principles* (pp. 1-18). doi:10.1145/3132747.3132785

Seo, J., Jeon, S., & Jeon, T. (2018). *Domain adaptive generation of aircraft on satellite imagery via simulated and unsupervised learning.* arXiv preprint arXiv:1806.03002.

Spieker, H., & Gotlieb, A. (2020). Adaptive metamorphic testing with contextual bandits. *Elsevier Journal of Systems and Software*, *165*, 110574. doi:10.1016/j.jss.2020.110574

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). *Intriguing properties of neural networks*. arXiv preprint arXiv:1312.6199.

Tian, Y., Pei, K., Jana, S., & Ray, B. (2018). Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering* (pp. 303-314). doi:10.1145/3180155.3180220

Tsymbal, A. (2004). The problem of concept drift: Definitions and related work. Computer Science Department. *Trinity College Dublin*, *106*(2), 58.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., & Darrell, T. (2014). *Deep domain confusion: Maximizing for domain invariance*. arXiv preprint arXiv:1412.3474.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, *10*(5), 988–999. doi:10.1109/72.788640 PMID:18252602

Weng, T. W., Zhang, H., Chen, P. Y., Yi, J., Su, D., Gao, Y., Hsieh, C. J., & Daniel, L. (2018). *Evaluating the robustness of neural networks: An extreme value theory approach*. arXiv preprint arXiv:1801.10578.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). *Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.* arXiv preprint arXiv:1708.07747.

Darryl Hond is a Thales Specialist in computer vision and image processing and has a PhD in face recognition. He is a member of Thales UK Research, Technology and Innovation (RTI), which is based in Reading. Darryl Hond has conducted research for over 20 years into object recognition, face analysis, motion detection and stereovision, with a view to incorporating the resulting algorithms into real-world applications. Recently, he has been developing neural network algorithms for object classification and scene understanding, as well as designing a verification methodology for artificial neural network classifiers.

Hamid Asgari is Thales Expert and a visiting professor at King's College London. He is currently leading both Verification &Validation and Network activities at Thales UK Research, Technology, and Innovation (RTI). He is a highly experienced in leading large collaborative R&D teams across Europe and in UK, a technical expert in Network & Cyber Security Architectures and concepts, Safety and Security Risk Management, Verification, Validation and Performance Evaluations of complex systems. Hamid has a proven track record and published more than 65 book chapters and papers in the most respected scientific journals and peer reviewed conferences. Hamid is IET fellow, Senior Member of IEEE and ACM. For full list of publication please visit: https://scholar.google.co.uk/citations?user=Mj_zIOEAAAAJ&hl=en.

Dan Jeffery graduated with an MEng in Electronic Engineering from the University of Portsmouth in 2019. During his time at university he completed four summer placements with Thales UK Research, Technology and Innovation. He has now returned to a full time role at Thales UK and is a member of Autonomous Systems Research Group, working in the fields of V&V and mission planning.

Mike Newman is a Thales UK Specialist in Sensor Signal Processing, having a worked on sensing systems from optical to acoustic to radar, including the fusion of multi-modal data. A recent theme in his radar work, which has stretched from underground to through-wall to airborne to space, is automatic classification of drones and helicopters by their radar Doppler signatures, leading to an interest in how Neural Networks can be verified for real-world operation.