IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 10 October 2023, accepted 12 October 2023, date of publication 23 October 2023, date of current version 3 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3326725

RESEARCH ARTICLE

A Dataset Fusion Algorithm for Generalised Anomaly Detection in Homogeneous Periodic Time Series Datasets

AYMAN ELHALWAGY, (Graduate Student Member, IEEE), AND TATIANA KALGANOVA, (Member, IEEE)
CEDPS, Department of Electronic and Electrical Engineering, Brunel University London, UB8 3PH Uxbridge, U.K.
Corresponding author: Tatiana Kalganova (tatiana.kalganova@brunel.ac.uk)

This work was supported in part by Voltvision.

ABSTRACT The generalisation of Neural Networks (NN) to multiple datasets is often overlooked in literature due to NNs typically being optimised for specific data sources. This becomes especially challenging in time-series tasks due to difficulties in fusing temporal data from multiple sources. However, in a commercial environment, generalisation can effectively utilise available data and computational power which is essential to Green AI, the sustainable development of AI models. This paper introduces ''Dataset Fusion,'' a novel dataset composition algorithm for fusing periodic signals from multiple homogeneous datasets whilst retaining unique features for generalised anomaly detection. The proposed approach, tested on a case study of three-phase current data from two different homogeneous Induction Motor (IM) fault datasets on anomaly detection, outperforms conventional training approaches with an Average F1 score of 0.879 and effectively generalises across all datasets. Furthermore, when tested with varying percentages of the training data, results show that using only 6.25% of the training data, translating to a 93.7% reduction in computational power, results in only a 4.04% decrease in performance, demonstrating the advantages of the proposed approach in terms of both performance and computational efficiency. Moreover, the algorithm's effectiveness under imperfect conditions highlights its potential for use in real-world applications.

INDEX TERMS Generalisation, dataset fusion, data reduction, anomaly detection, neural network training, green AI, time series, environmental AI.

I. INTRODUCTION

Generalisation is a measure of a Neural Network's (NN) performance on data that it has not seen before but that is in the same class as the data that it has been trained on. The idea behind generalisation with Deep Learning (DL) is to transfer domain knowledge from data the NN has been trained on to unseen data in the same class, where the unseen data may contain conditions that slightly vary from the training data. This allows for a NN to be able to maintain performance across the dataset, and potentially transfer across multiple datasets with a similar distribution to the initial trained data. Various studies have been undertaken to understand the factors that affect the generalisation performance of a NN [1], and how to mitigate these factors to achieve the optimal level of performance [2]. The underlying concept is as follows: When training a NN on a data sample, the NN learns to represent the function between the input data and the output data through the adjustment of the weights and biases. If the distribution of the data sample used for training is not fully representative of the true distribution population, then the input-output function that is represented by this sample will inevitably vary from the function of the population. Extensive training on this sample will then result in a phenomenon known as overfitting [3], which refers to when the NN has accurately modelled the function represented by the training data but is not able to generalise to data in the same class due to the discrepancy between the functions represented by the sample and population.

Many works have been published with the aim of addressing overfitting and thus maximising the potential

The associate editor coordinating the review of this manuscript and approving it for publication was Thomas Canhao Xu.

generalisation ability of a NN. Many works focus on architectural improvements to the NN to increase the robustness of the NN to overfitting through novel architectures such as Capsule Networks [4], whilst other research directions focus on the manipulation of the training procedure to limit overfitting with techniques such as Dropout [5], Early Stopping [6], Pruning [7] and adding noise to the weights and biases of the NN whilst training [8].

There is much less focus on research concerning the effect of the composition and specification of the dataset on generalisation, especially regarding time series data. A common approach currently used includes denoising the training sample to better align the distribution of the sample to the population and hence limit the level of overfitting [9], [10]. Additionally, there is a consensus in ML research that increasing the volume of data through various means such as augmentation [11], [12] improves NN generalisation performance; whilst this has been empirically confirmed, more recent research has discovered that this improvement largely comes specific samples within the supplementary data, and a significant volume of this data is essentially redundant and does not contribute to a performance improvement [13], [14].

Furthermore, there is a gap in literature concerning the fusion of multiple time series datasets in a single training set to balance the probability distribution of the training sample so that it better aligns with the true distribution of the problem domain. This is largely due to a lack of necessity in an experimental environment since most ML research tends to optimise the solution for a single dataset source. However, from a commercial standpoint, this can have many benefits with regard to time and computational power saved, as well as the added benefit of reducing the data requirements for training. In addition to this, the dynamic shifting of the distribution of data is often a bottleneck to the performance of the NN; this is a prevalent issue that is encountered when a NN is deployed in a non-stationary environment, which is common with time series data. Some recent works have detected this shift [15], and mitigate the effect this has on the generalisation performance of the NN with both time series data [16], [17] and image data [18]. However, the majority of empirical evaluations of NN approaches in literature are mostly conducted on an isolated sample of data, which, in many cases, is not representative of the dynamic shifting of the distribution temporally.

To address the identified gaps in the literature, we propose a novel algorithm, named Dataset Fusion. The proposed method merges multiple homogeneous, periodic time series datasets into a single unified dataset for training anomaly detection NN models. The fusion process is designed to accurately represent population distributions and increase robustness against potential data distribution shifts. Our primary objective in this study is to examine efficient generalisation approaches that can minimise training time and computational demands for neural networks when working with new homogeneous time series data sources. The contributions of this paper can be summarised as follows:

- A novel dataset composition algorithm is proposed, referred to as *Dataset Fusion*
- The proposed approach is applied to a case study focused on motor current data, with a qualitative analysis conducted to assess the preservation of features from each individual dataset.
- The generalisation performance of the proposed method is empirically evaluated in anomaly detection with the LSTMCaps neural network architecture from previous work [4], and compared to the performance when using conventional training approaches
- The potential practical limitations of the proposed method in a real-world environment are discussed and assessed through further experimentation

In the context of the sustainable development goals (SDGs), our work primarily focuses on Goal 9 (Industry, Innovation, and Infrastructure) through the exploration of innovative approaches to improve the performance and efficiency of neural network models; Goal 12 (Responsible Consumption and Production) through the proposal of a method that demands less computational power and training data, and Goal 13 (Climate Action) through the resulting reduction in the energy consumption for model training.

II. RELATED WORKS

This section investigates the current literature on different methods of addressing NN generalisation performance, as well as recent works using multiple datasets in different domains and tasks.

A. NN BASED GENERALISATION TECHNIQUES

1) DROPOUT

Dropout [5] is a regularisation technique that can be used to prevent the neural network from overfitting on the training data. This is done by ignoring several randomly selected neurons, with the number of neurons dropped dependent on the ''dropout rate'' parameter, so they do not affect the output of the neural network on the forward pass. The idea behind dropout is to effectively train many subnets in your network so that your network acts as a sum of many smaller networks that can learn the representation of the data without the presence of the dropped-out neurons. This was found to improve the generalisation performance of the network by reducing data overfitting.

2) PRUNING

Pruning is a process whereby a NN selectively removes trainable parameters based on an established criterion with the aim of maintaining the performance of the NN [19]. There are two main categories of pruning: Structured and Unstructured pruning. Unstructured pruning directly removes trainable parameters from the network, such as connections to neurons (weights). Structured pruning involves removing entire structures from the network such as neurons and filters. Structured pruning allows for a faster computational time in relation to unstructured pruning, as most frameworks existing for Machine Learning (ML) do not allow for the acceleration of sparse matrix calculations, therefore the NN will be able to reduce the number of calculations for the former but not the latter.

Various criterion has been established in literature for the pruning of NNs. One primitive but popular method is known as the weight magnitude criterion. The criterion dictates that the weights of the smallest magnitude are removed. The idea behind this is to remove all the weights that contribute the least to the function as they are less likely to impact the final prediction.

The most established framework for pruning is known as the train, prune and fine-tune method [19]. As the name suggests, the model is first trained, then iteratively pruned and fine-tuned based on the weight magnitude criterion. More recently however, an increasing number of works [7], [20], [21] have evolved this framework with novel methodology that has allowed for further reduction of NN parameters and hence more efficient training and computation whilst maintaining similar performance.

B. DATA-BASED GENERALISATION TECHNIQUES

Data-based generalisation techniques are largely overlooked in the field of deep learning in comparison to NN-based techniques. This is because the NN structure and learning optimisation algorithms are usually the reason for such weak performance, so improvements can largely be made by improving and optimising how the NN learns as opposed to what the NN is learning. However, it is still important to consider the training data as it can be a bottleneck for learning ability if not composed in the correct manner [22].

An important aspect to consider is the difference in the data distribution between the data used to train the NN and the data that the NN will eventually be applied to. Often the data picked for training is not fully representative of the true distribution of the dataset, which creates a bottleneck to generalisation as the NN is not prepared for the distribution found in the overall population since it has been tuned to the distribution of the training sample. Shuffling the data before taking the training sample often helps with this to increase the likelihood of the sample representing the true distribution. Furthermore, shuffling during training also helps the NN weights to escape local minima and converge towards the global minimum of the function. Many works in deep learning falsely assume that the data distribution is static, which is largely incorrect as in practice data distributions are generally dynamic and tend to shift away from the test data distribution with real-world data; this phenomenon is known as distribution shift. This shift has been detected and quantified in recent works [23], [24], and accounted for with online adaptation to the shift [25], which allows the NN weights to adjust themselves to account for this distribution shift.

Data with excessive noise can inhibit the NNs ability to learn the true input-output function required by a specific application. Mathematically, this can be explained by the bias-variance decomposition [26], specifically the variance component. The variance value refers to the change in prediction accuracy over different subsets of the data. A high variance value indicates that the NN has learnt the noise in the data, or in other words overfit on the training data. The most common method of reducing the variance is increasing the volume of the training data, which will naturally bring the distribution of the data closer to the required distribution that represents the overall dataset as opposed to just the subset of training data. As well as using real-world data, data augmentation has also proved to be an effective method of increasing data volume using data that is already accessible [11], [12]. Another effective method that is commonly used is the denoising of data [27]. This also contributes to the reduction of the variance: By removing the noise in the data, the training data subset will be cleaner and will more accurately represent the true distribution of the dataset. Whilst both methods are generally proven to work, there are still major issues with both methods that are still being addressed in literature, such as effectiveness in a real-world situation.

The financial and computational costs associated with increasing the volume of training data are not just substantial but also rapidly increasing [28]. This makes it increasingly difficult to train models without significant resources. Consequently, only well-funded entities with considerable resources can achieve a performance boost using this method, as their computational power and finances typically surpass those of other research entities and companies. This disparity creates a bottleneck for smaller entities, hindering their competitiveness in the field [29].

Denoising is still a very active field of research due to the many limitations of the currently proposed techniques. Since it is very difficult to determine which aspects of the data features are representative of the true distribution [27], it is very difficult to use denoising techniques such as filtering since features that are potentially important to NNs could be filtered out, limiting the potential performance of the NN. Furthermore, filtering techniques are largely contextual, so it is very difficult to develop filtering methods that work across multiple contexts to the same level of effectiveness.

Transfer learning [30] is also widely regarded in literature as a robust method of improving performance by utilising data from a different set but in the same domain as the target data. Whilst there have been many recent works regarding transfer learning [31], there is a gap in research concerning dataset fusion methods, an alternative to transfer learning that utilises multiple datasets in the same training phase, as opposed to multiple training phases to train a model more robust to the difference in data distributions between the training data and the data encountered during operation. This will be further explored in the present work.

C. MULTI-DATASET TRAINING

Recently, researchers have identified the potential benefits of training generalisable NNs using multiple datasets to enhance performance and expand the capabilities of the NN models. Many of these approaches primarily focus on image-based applications [32], [33]. For example, Yao et al. [34] introduced a novel framework for cross-dataset training, leveraging pre-existing labels from multiple datasets to create a single model capable of detecting the union of labelled features from all contributing datasets.

This approach aims to maximise the utility of available labels for distinct classes in each dataset, thereby circumventing the time-consuming and resource-intensive process of labelling a single dataset with new classes that are already present in another dataset. Empirical evidence provided by the authors demonstrates the effectiveness of this approach when applied across multiple datasets, achieving comparable performance levels without sacrificing accuracy. In a related study, Zhou et al. [35] introduce a technique for training a unified object detection model on several extensive datasets by using dataset-specific training methods and losses, but maintaining a shared detection architecture with outputs specific to each dataset. This approach circumvents the necessity for manual taxonomy alignment, as it automatically combines the outputs of different datasets into a unified semantic taxonomy. The authors show that this multi-dataset detector achieves comparable performance to dataset-specific models in their respective domains while effectively generalising to unseen datasets without the need for fine-tuning. By utilising multiple training datasets, these approaches can lead to reduced resource requirements in terms of training data and improved performance. However, it is worth noting that they do not specifically address the aspect of reducing computational power during the training process.

D. DOMAIN ADAPTATION

Domain adaptation techniques are designed to enable models to leverage common features between different but related datasets, while also accommodating the unique characteristics of each domain [36]. Techniques such as Domain-Adversarial Neural Networks (DANN) [37] and Domain Adaptive Faster R-CNN [38] have shown success in allowing models to generalise across domains with slight non-homogeneity. These models identify both domain-specific and domain-invariant features, facilitating training on both source and target domains. Furthermore, they show considerable strength in dealing with the domain shift from source to target domains, which is a common issue when integrating datasets from different contexts or applications. However, while these techniques offer significant advancements in handling domain shifts, they do not explicitly address the distribution characteristics of the initial source domain, an aspect which the present study aims to explore and address using the proposed method.

In summary, analysing previous works on multi-dataset utilisation reveals clear benefits, such as reduced dataset labelling requirements and increased generalisation performance. However, there is a lack of exploration in time-series-based multi-dataset models due to challenges in fusing sequential data from different sensors and collection specifications. The current study aims to address these points by proposing a novel approach for effectively integrating time-series data from multiple sources. Furthermore, it is essential to consider computational efficiency, as training on multiple datasets may require additional computational resources. By addressing these points, this study aims to contribute valuable insights to the field of multi-dataset modelling and help pave the way for more robust and efficient approaches in various applications, particularly those involving time-series data.

III. DATASET FUSION

A summary of the dataset fusion process is depicted in Figure 1.

The signal in each dataset is first down-sampled to the sampling frequency, F_s, of the dataset with the lowest sampling frequency. This step is essential to models with look-back such as Recurrent Neural Networks so that the same signal length is considered for every motor when training the model. Taking the example in Figure 1, for a set of signals $\{A[n], B[n], \ldots, N[n]\}$ with sampling frequencies $\{F_{s_A}, F_{s_B}, \ldots, F_{s_N}\}$, the target sampling frequency, $F_{s_{new}}$, is expressed in Equation 1.

$$F_{s_{new}} = min\{F_{s_A}, F_{s_B}, \ldots, F_{s_N}\} \tag{1}$$

The re-sampling is implemented using the Fourier method. This method was chosen over decimation due to the simplicity of implementation and with the assumption that the signals used are periodic in nature. To avoid aliasing and other artefacts, a low-pass windowed-sinc filter is first designed and applied to the signal, with a cutoff frequency based on the target Nyquist frequency. The Hann window, $hann(n)$, was employed in the filter design due to its desirable characteristics for resampling, such as reduced spectral leakage and smooth sidelobes. In the present work, 101 taps were used, giving a filter order of 100. Equation 2 expresses the impulse response, $h[n]$, of this filter.

$$h[n] = K \cdot \frac{\sin\left(2\pi f_c(n - M/2)\right)}{n - M/2} \cdot hann(n) \tag{2}$$

where K is the normalisation factor, f_c is the cutoff frequency in Hz, n is the discrete time index, and M is the filter length or number of taps.

To implement the Fourier Method, the time series signal is first transformed into the frequency domain with a Discrete Time Fourier Transform (DTFT). For a discrete periodic sequence $x[n]$ its DTFT, $X[\omega]$ is expressed in Equation 3.

$$X[\omega] = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \tag{3}$$

where $\omega = 2\pi f$ and $j = \sqrt{-1}$.

The spectrum $X[\omega]$ is then band-limited to the Nyquist frequency of the lowest F_s. The resulting spectra are then inverse-transformed back into the time domain. The down-sampling operation is expressed in Equation 4a and the new number of samples, N_{new}, is calculated using Equation 4b. Equation 5 shows the Inverse DTFT operation used to obtain the resampled time series signal.

$$x[\omega]_{resampled} = \{x[\omega] : \omega < \frac{F_{s_{new}}}{2}\} \tag{4a}$$

$$N_{new} = \frac{N}{F_s} * F_{s_{new}} \tag{4b}$$

$$x[n]_{resampled} = \frac{1}{N_{new}} \sum_{\omega=0}^{N_{new}-1} X[n]e^{\frac{j2\pi\omega n}{N_{new}}} \tag{5}$$

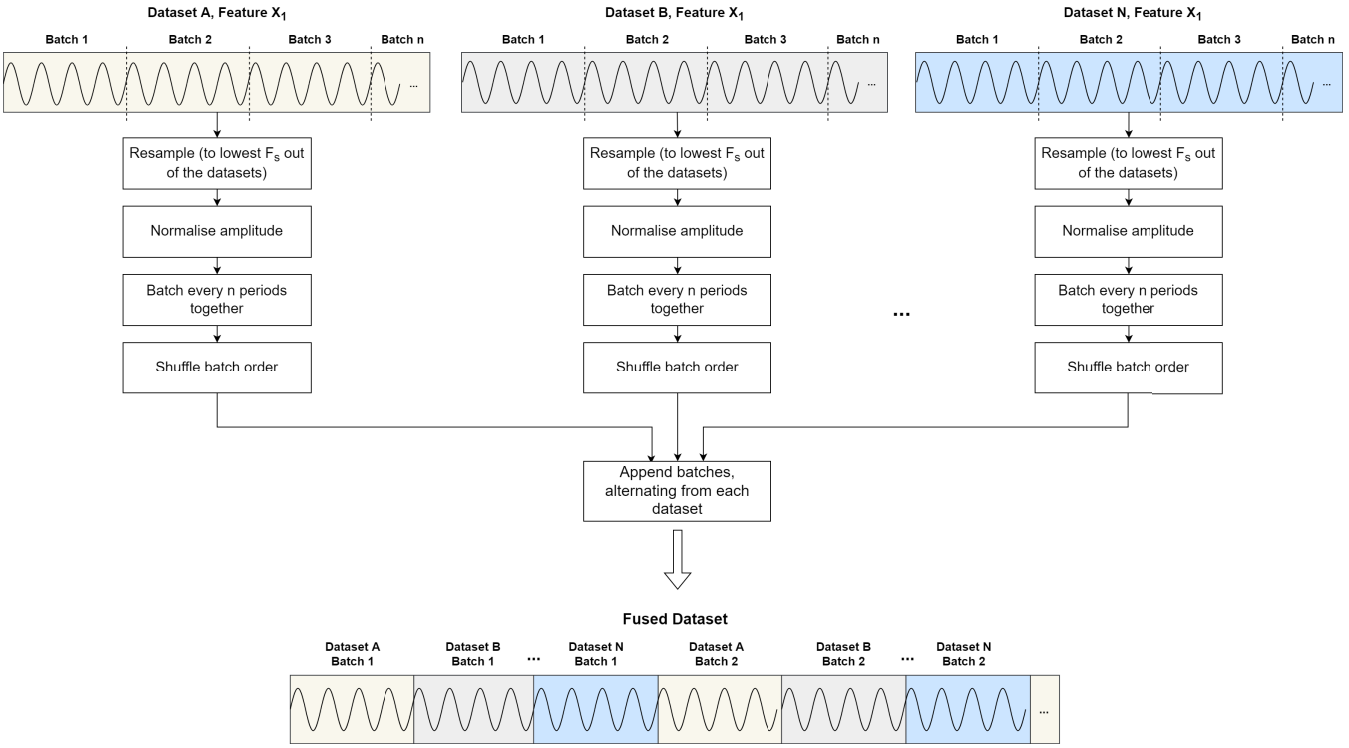


FIGURE 1. A summarised illustration of the Dataset Fusion algorithm.

Each dataset is then normalised using Z-score normalisation, to overcome varying motor currents. For the resampled sequence, $x[n]_{resampled}$, the normalised sequence, $x'[n]$, is calculated using Equation 6.

$$x'[n] = \frac{x[n]_{resampled} - \bar{x}_{resampled}}{\sigma_{x_{resampled}}} \tag{6}$$

where $\bar{x}_{resampled}$ is the mean of the resampled sequence, and $\sigma_{x_{resampled}}$ is the standard deviation of the re sampled sequence.

To batch the periods together, a zero-crossing algorithm, configured to detect crossings from positive to negative, is employed to first identify a single period, and then concatenate n periods based on the user-defined parameter. Given that z-score normalisation is utilised, any periodic time-series data will exhibit sign changes, making the zero-crossing algorithm applicable. In cases where multiple features are present in the data, the zero-crossing algorithm calculates only the first feature as a reference for period batching, in order to maintain the temporal integrity of the data and preserve the spatial relationship between features. The impact of varying batch sizes on training performance differs based on the nature of the data and problem domain; hence, it is recommended that this parameter be optimised alongside other training hyperparameters. For a discrete periodic signal $x[n]$ with assumed periodic sign changes, the set of indices for the positive-to-negative zero crossings, $c_{+\to-}$, can be expressed as shown in Equation 7.

$$c_{+\to-} = \{\, n \mid x[n-1] \; > 0 \geq x[n] \,\} \tag{7}$$

For each dataset, the batch order is then shuffled randomly. This is done in order to mitigate the effect of distribution shift and prevent noise in one area of the signal from being prevalent in other areas of the signal. In other words, this step helps to reduce the variance of the NN prediction. A new signal is then constructed using the shuffled batches from each dataset by appending a batch from each dataset in an alternating fashion.

The full process of the Dataset Fusion algorithm is expressed in Algorithm 1.

where % represents the modulo operator and $+\!\!+$ represents concatenation.

A. COMPUTATIONAL COMPLEXITY

The computational complexity of the Dataset Fusion algorithm, represented in Big O notation, can be determined by breaking down the steps of the algorithm when practically applied. The breakdown of the complexity of each stage is provided in Table 1.

TABLE 1. Algorithm complexity for dataset fusion algorithm, for *n* datasets with *m* length.

Algorithm step	Big O Complexity
Filtering and Resampling	$O(nm(1 + log(m)))$
Normalisation	$O(nm)$
Period Batching	$O(nm)$
Chaining and Stacking batches	$O(nm)$
Total	$O(nm(1 + log(m))) + 3O(nm)$

Algorithm 1 Pseudocode of Dataset Fusion Algorithm

Function: Dataset_Fusion(x, F_s, P)
Input:
A set of s finite discrete periodic sets x where $s > 1$
A set of sampling frequencies F_s corresponding to X
Number of periods batched P
Output: x_{fused}
Determine $F_{s_{new}}$ using Equation 1
for x_1 to x_s **do**
if $F_{s_{X_s}} \neq F_{s_{new}}$ **then**
Apply filter shown in Equation 2
Calculate $X[\omega]$ using Equation 3
Calculate $X_{[\omega]_{resampled}}$ using Equation 4a
Calculate N_{new} using Equation 4b
Calculate $x[n]_{resampled}$ using Equation 5
Calculate x' using Equation 6
Calculate $c_{+\rightarrow -}$ using Equation 7
Calculate $x'_{batched}$ through grouping P periods by slicing x' at the values where $c_{+\rightarrow -}[n]\%P = 0$
Shuffle $x'_{batched}$
$x_{fused} = \{x'_{1_{batched}}[0] +\!+ \ldots +\!+ x'_{s_{batched}}[0] +\!+ x'_{1_{batched}}[1] +\!+ \ldots +\!+ x'_{s_{batched}}[1] +\!+ \ldots\}$
return x_{fused}

As Table 1 shows, the filtering and resampling step has a complexity of $O(nm(1 + log(m)))$, since it involves applying a finite impulse response (FIR) filter with a complexity of O(m) and performing resampling using the Fast Fourier Transform (FFT) with a complexity of $O(m * log(m))$ for each of the n datasets. The Normalisation step scales each dataset using Z-score normalisation with a complexity of $O(m)$ for each dataset, resulting in a total complexity of $O(n * m)$. The Period Batching step identifies zero-crossings and creates period batches with a complexity of $O(m)$ for each dataset, resulting in a total complexity of $O(n * m)$. Finally, the Chaining and Stacking batches step involves filtering, chaining, and stacking the period batches with a total complexity of $O(n * m)$. The overall complexity of the Dataset Fusion algorithm is the sum of the complexities of these steps, which is $O(n * m * (1 + log(m))) + 3 * O(n * m)$, with the dominating term being $O(n * m * log(m))$ due to its faster growth as the input size (n and m) increases.

The logarithmic factor in the dominating term, $O(n * m * log(m))$, makes the Dataset Fusion algorithm scale well with increasing input size. This is because logarithmic growth is slow growth, ensuring that the algorithm remains efficient even as the number and size of the datasets (n and m) increase. Additionally, since the complexity is dependent on both the number of datasets (n) and the length of the datasets (m), the algorithm can efficiently handle varying dataset sizes and compositions. This scalability makes the Dataset Fusion algorithm a versatile algorithm and suitable for processing large and diverse datasets, which is essential in the context of real-world applications where data size and complexity are constantly evolving.

B. REQUIREMENTS FOR APPLICATION

Whilst the proposed algorithm is domain-independent, there are requirements regarding the data that must be met for the proposed method to be applicable. These requirements, as well as the reasoning, are detailed in the following sections.

1) HOMOGENEOUS DATASETS

Although the methodology can be used in varying problem domains, the dataset fusion algorithm can only fuse homogeneous data, since the aim of the algorithm is to capture the data distribution of a problem domain as a whole in order to mitigate overfitting on a specific dataset. Generalisation to multiple problem domains is not in the scope of this algorithm.

2) DATA PERIODICITY

As explained in section III, The algorithm relies on the fact that the data is periodic, due to the resampling method used, the zero crossing method, and to be able to create a coherent and usable sequential fused time series dataset.

3) TIME DOMAIN DATA

The proposed approach will only be applicable in the time domain representation of the datasets, as it relies on the sequential nature of the data to fuse it together in a meaningful way.

If the datasets being fused meet the requirements detailed above, then there is feasibility in applying the proposed method. Some examples of where the proposed method may be feasible are daily temperatures in a region, electrical power data and vibration data.

C. PROPOSED BENEFITS OF DATASET FUSION

The Dataset Fusion algorithm seeks to eliminate the necessity of multiple NNs for a single problem domain. This approach theoretically enables the development of an NN that can adapt to unseen data from the same domain, even if originating from different data sources. Moreover, the Dataset Fusion algorithm aims to reduce data requirements from individual sources, as achieving ideal data collection conditions from each source can often prove to be challenging. Potential issues with collected data, such as data corruption, sensor faults, or insufficient data volume, among other data collection complications, further emphasise this need. The present study will experimentally investigate the proposed benefits of the Dataset Fusion algorithm.

IV. CASE STUDY: DATASET FUSION FOR 3-PHASE MOTOR CURRENT DATA

This section will explore the feasibility of applying the proposed method with a case study on motor current signals. The aim of the case study is to empirically test and validate the effectiveness of our proposed method. The datasets used will first be introduced and the feasibility of the proposed method will be confirmed. The dataset fusion algorithm will then be applied, and the resulting signal will be compared and analysed to the original signals.

A. DATASET INTRODUCTION

For the present case study, two homogeneous open-source datasets [39], [40] will be used to confirm the feasibility of the Dataset Fusion methodology. Specifications of the datasets used are detailed in Table 2. Both datasets are composed of three-phase motor current signals, however, one dataset, which will be referred to as Dataset A, contains fault data for an inter-turn short circuit fault, and the other dataset, which will be referred to as Dataset B, contains current signals for a broken rotor bar fault.

1) DATASET A

Dataset A [40] contains files from a motor running at a variable operating frequency F_o, ranging from 30Hz to 60Hz with 5Hz increments. The motor has the following specifications: 4 poles, 1HP mechanical power, 220V supply and 3A rated current. The authors simulated both high-impedance and low-impedance short circuits, for different levels of fault severity. For the purpose of this case study, only the files captured at $F_o = 60Hz$ were used to meet the limitation of Dataset Fusion of only being applicable to homogeneous data. The new dataset structure is depicted in Table 3.

2) DATASET B

The motor used to capture Dataset B [39] is a squirrel cage AC motor, running at a constant $F_o = 60HZ$ and has similar specifications to the motor used to capture Dataset A. The breakdown of the dataset is shown in Table 4. At the beginning of each file, for roughly the first 4 seconds, a transient signal representing the motor startup was also recorded. For the purpose of the case study, and for compatibility with the proposed algorithm, the transient subset of the signal, the first 200,000 samples, was discarded from each file, so that only the steady state of the motor remained. This left 801,000 samples left in each file, representing an approximate 20% redundancy of data.

B. APPLICATION OF DATASET FUSION AND ANALYSIS

The Dataset Fusion algorithm was used to fuse the healthy files from Datasets A and B into a single, fused dataset. First, all healthy files were extracted from each dataset and concatenated into a large signal. The files in Dataset B were first sliced to remove the motor startup signature, then resampled to 10,000Hz, the same F_s as Dataset A. Each Dataset was split into batches of 4 periods and then concatenated alternating between each Dataset to create the final fused dataset. This was found to be the optimal value for this dataset through a grid-search-based optimisation of the parameters for DF to maximise NN performance. The batches were then concatenated, alternating between each dataset, to create the final fused dataset.

For each dataset, an initial analysis was conducted in order to understand the data and enable a more accurate interpretation of the fused dataset experimental results. The time series signal, a Probability Distribution Function (PDF) and FT representations from 0-500Hz were generated for a single phase from a healthy file from each dataset. The plots are illustrated in Figure 2.

A Principal Component Analysis (PCA) was also performed on the healthy data from each dataset as well as the fused data to gain comprehensive insights into the data, wherein the resulting axes are linear combinations of the original variables, defined by the eigenvectors and eigenvalues. This method allows for the identification of the most significant patterns to increase the interpretability of the proposed method. All datasets were uniformly downsampled to 10,000Hz, which corresponds to the minimum sampling frequency in Dataset A. Subsequently, the samples were partitioned into groups of 100,000, aligning with the smallest sample size per file in Dataset A. The data's three features, representing the three phases, were flattened into a single axis before being subjected to the PCA algorithm. The visualisation of the first two Principal Components in a 2D scatter plot can be found in Figure 3.

It is clear to see from Figure 2(d) and Figure 2(e), as well as Figure 2(g) and Figure 2(h) that Dataset B contains a considerable amount of noise in comparison to Dataset A. In addition to this, the frequency spectra of Dataset A show more pronounced harmonics in comparison to Dataset B. Although this may not be as evident from the time series signal plot, a NN will most likely pick up these differences in noise, and thus a NN trained on a single dataset, especially in the case of Dataset B, will struggle to distinguish files from Dataset A with fault signals as anomalous. This hypothesis will be further discussed in the results and discussion.

The time series signal of the fused data looks similar visually in comparison to the datasets, albeit in a different input space due to normalisation. From the PDF and FFT representations shown in Figure 2(f) and Figure 2(I) however, there are subtle indications of features present in both Dataset A and Dataset B. For instance, the overall shape of the frequency spectra follows Dataset A, however, there is noise clearly present in the spectra, a significant feature of Dataset B. Furthermore, the harmonic peaks in the fused frequency spectra contain the same characteristics of both datasets, which is interesting to note as this representation would still be considered a healthy signal. Future work will investigate the use of a fused Dataset in the frequency domain to train a classifier NN. However, the scope of this study is to validate the use of the time series representation to train a generalised time series anomaly detector with reduced data requirements.

Upon examining the PCA plot depicted in Figure 3, it is clear that the healthy data from both datasets exhibit comparable traits and patterns. Interestingly, the fused dataset forms a cluster around the origin, positioning itself at the center of the two datasets. This central location of the fused dataset within the circular arrangement of points from the two datasets signifies that it effectively captures the salient features of both datasets. By doing so, the fused dataset aids in bringing the training data closer to the population distribution of the problem domain, thereby enhancing the robustness and generalisability of the model derived from this data. Further evidence of this will be given in the experimental results.

TABLE 2. Specification for motor datasets used in the case study.

Dataset Name	Data volume per file (samples, features)	Faulty Data Files	Healthy Data Files	Sampling Frequency (Hz)	Duration per file(s)
Dataset A: Inter-turn short circuit fault dataset (Cunha, 2021)	(100,000, 3)	2264	353	10,000	10
Dataset B: Broken Rotor Bar Dataset (Maciejewski, 2020)	(1,001,000, 3)	320	80	55,611	18

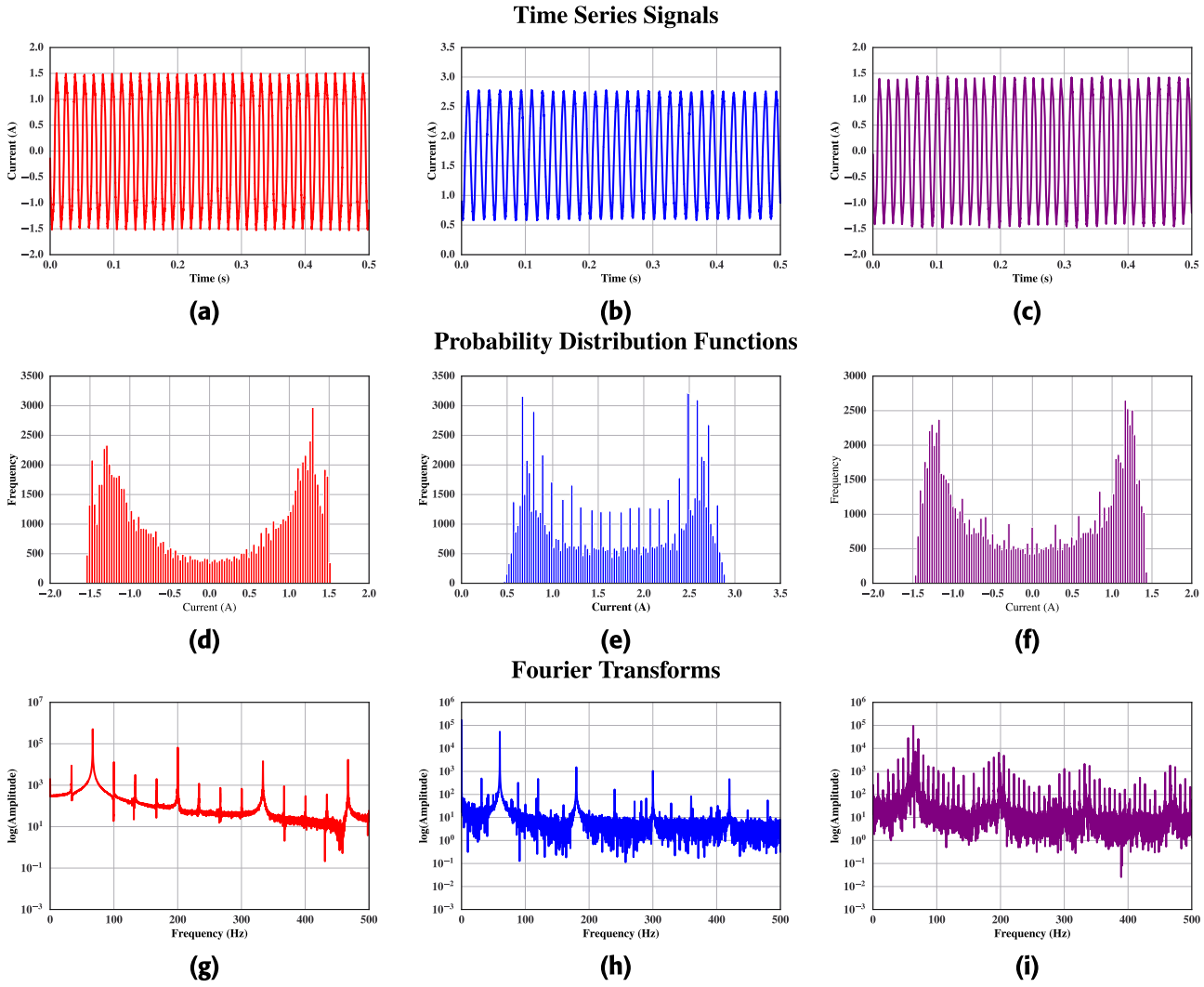


FIGURE 2. Time series signal from Dataset A (a), B (b) and fused (c), Probability Distribution Function from Dataset A (d), B (e) and fused (f), and Fourier Transform from Dataset A (g), B (h) and fused (i) healthy files.

It is important to note that simply concatenating two healthy files from each Dataset will produce a similar outcome to the representations shown in Figure 2. However, the purpose of this analysis is to show that the Dataset algorithm will still preserve the individual features of each representation in a new signal and still be usable for a data-driven approach. The PCA plot, as displayed in Figure 3, provides more compelling evidence of the impact of the Dataset Fusion technique on the combined dataset. Subsequent experimental results on anomaly detection, utilising the various datasets, will further explore the implications of employing a fused dataset for training an anomaly detection model.

C. EXPERIMENTAL DESIGN

The aim of the experimentation presented in this study is to observe the effectiveness of Dataset Fusion in training an anomaly detector Neural Network (NN) using some of the healthy files from Dataset A and Dataset B, and then evaluating the model on the remaining healthy and faulty files, in comparison to commonly used training methods. The training methods selected as baselines in this study are commonly utilised in the domain of anomaly detection and have been previously validated in literature [41], [42]. These methods serve as a standard against which the proposed Dataset Fusion method is compared. The following

TABLE 3. Breakdown of dataset A 60Hz files.

Motor State	Number of files	Samples per feature
Healthy	48	4,800,000
High Impedance 1 (1.41% of stator winding)	53	5,300,000
High Impedance 2 (4.81% of stator winding)	52	5,200,000
High Impedance 3 (9.26% of stator winding)	48	4,800,000
Low Impedance 1 (1.41% of stator winding)	55	5,500,000
Low Impedance 2 (4.81% of stator winding)	54	5,400,000
Low Impedance 3 (9.26% of stator winding)	60	6,000,000

TABLE 4. Breakdown of dataset B.

Motor state	Number of files	Samples per feature
Healthy	80	1,001,000
1 Broken Bar	80	1,001,000
2 Broken Bars	80	1,001,000
3 Broken Bars	80	1,001,000
4 Broken Bars	80	1,001,000

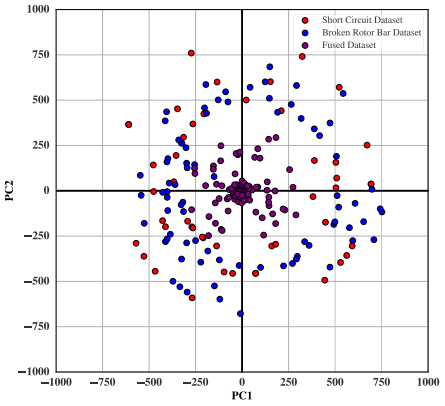


FIGURE 3. Principal component analysis of dataset A, B and the fused dataset.

training methods will be compared for all of the subsequent experiments:

- **Traditional Approach**: This approach involves training on a single dataset. It's a common baseline method used in many studies [41].
- **Transfer Learning**: A two-phase training method where the first training phase occurs on one dataset, followed by a second training phase on another dataset. This method leverages knowledge transfer between datasets and has shown promise in related works [42].
- **Mixed Dataset**: A single training phase is conducted using all healthy files from each dataset. This method aims to leverage the diversity of multiple datasets.
- **Dataset Fusion**: Our proposed method involves a single training phase on fused healthy data consisting of all datasets.
- **Dataset Fusion with Transfer Learning**: Combines the dataset fusion and transfer learning approaches, with the first training phase on fused healthy data consisting of all datasets, followed by a second training phase on a single dataset.

To provide clarity on the variations within each training method, Table 5 provides a full breakdown of the different variants, along with labels used in the experimental results tables.

TABLE 5. Experiment variants and corresponding key for results tables.

Experiment	Key
Traditional Approach - Dataset A	T - Dataset A
Traditional Approach - Dataset B	T - Dataset B
Transfer Learning - Dataset A to Dataset B	TL - Dataset A to B
Transfer Learning - Dataset B to Dataset A	TL - Dataset B to A
Mixed Dataset	MD
Dataset Fusion	DF
Transfer Learning - Fused Dataset to Dataset A	TL - DF to Dataset A
Transfer Learning - Fused Dataset to Dataset B	TL - DF to Dataset B

The same workstation was used to conduct all experimentation in order to maximise experimental rigour. The specifications of this workstation are given in Table 6, for the purpose of experiment reproducibility.

TABLE 6. Specifications for workstation used for experimentation.

Component	Specification
Operating System	Windows 10 Version 21H2
CPU	AMD Ryzen Threadripper 2990WX 32-Core 3.5GHz
RAM	64GB
GPU	NVIDIA RTX A6000 48GB VRAM

Each experiment iteration was repeated 10 times for experimental rigour. The outcome of each experiment is validated for statistical significance using an Analysis Of Variance (ANOVA), to ensure the reproducibility of the results presented. The ANOVA is generated using custom functions on Python 3.9, and the numpy [43] and pandas [44] libraries, with versions 1.22.0 and 1.3.5 respectively.

The Precision, Recall and F1 score metrics, popularised in [45], will be used to evaluate the performance of the anomaly detector model with each training method. Equation 8, Equation 9, and Equation 10 show how the Precision, Recall, and F-beta scores are calculated, respectively:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{8}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{9}$$

$$F_\beta\ Score = (1+\beta^2) \times \frac{Precision \times Recall}{(\beta^2 \times Precision) + Recall} \tag{10}$$

where 'True Positives' (TP) denote the anomalies correctly identified by the NN model, 'False Positives' (FP) indicate the normal events incorrectly classified as anomalies, 'False Negatives' (FN) are anomalies that the NN model failed to detect, and β is the degree of prioritisation of recall over precision. In this study, we set β to 1, which means that the F-beta score becomes the F1 score, treating precision and recall equally in its calculation.

1) NEURAL NETWORK MODEL

The multi-channel LSTMCaps autoencoder NN developed in previous work will be used as the anomaly detection models for the following experiments. Further details regarding the architecture are given in [4]. For the following experimentation, three input branches were used to accommodate the three features present in the dataset: the three-phase motor current signals. The model hyperparameters were optimised for anomaly detection using a grid-search procedure. Table 7 details the optimal hyperparameters for this task, which will be used across all training methods. For all experiments, this model will be trained on ''healthy'' motor data only, since it is an unsupervised autoencoder.

TABLE 7. Optimised hyperparameters for neural network.

Hyperparameter	Value
Optimiser	Adam
Learning Rate	0.001
Epochs	8
Time Steps	1
Training samples	4,000,000
Batch Size	256
Input Branch Layer Width	32
Output Layer Width	96

As Table 7 shows, the NN trains best with 4M samples. Since the healthy data from each dataset contains over 4M samples, each file from the training set was randomly sliced to reduce the number of samples from each file, totalling the 4M samples required for each dataset when concatenated. This approach ensures that the model will train on a wide variety of data, and subsequently increase the reliability of the experimental results by avoiding optimising for a specific set of files.

There are various methods of determining the error thresholds of an Autoencoder NN. The most common method is through the use of an unseen validation set. After training the NN, the model is run on a file in the same class as the training set, but which has not been used for training. In this case, a single file containing data for a healthy condition motor is used. The Mean Absolute Error (MAE) for each prediction is then calculated, which provides a baseline for the accuracy of the NN with reconstructing healthy data. When multiple features are present in the dataset, as is the case with the data used in this case study, a threshold is calculated for each feature since the reconstruction performance of the NN model may vary across each feature. The overall threshold can be calculated through different methods, and the method used is determined based on the training performance of the NN as well as the data consistency and behaviour. In the present work, two methods are used: The largest MAE for each feature, or two standard deviations away from the mean MAE of each feature. The reasoning behind using two standard deviations from the mean as a threshold is, assuming a Gaussian distribution of error residual values, two standard deviations from the mean covers 95% of the data. In the case of an inconsistent or noisy dataset, it is better to use this method since there are more likely to be anomalous MAE values in the validation set, and if the largest MAE value is used, the threshold may be too high to consistently detect abnormalities in the test data. In the case of a consistent dataset with contains minimal noise, the largest MAE value is generally a good threshold to use since an MAE value to exceeds this threshold is more likely to indicate an anomaly as opposed to noisy but healthy behaviour.

2) ADDRESSING LIMITATIONS THOUGH EXPERIMENTAL DESIGN

A potential limitation of the proposed experimentation is the experimentation on static datasets. By experimenting on static datasets, results achieved in experimental conditions may not generalise effectively to real-world conditions or even other data in the same problem domain. One experiment will address this limitation by recreating the dataset for each of the ten test iterations, shuffling the dataset files for each test iteration randomly in each dataset prior to data formulation.

Another potential limitation is the availability of an equal amount of health data from each Dataset tested. For the purpose of this experimentation, an equal amount of data from each dataset will be used, however, it cannot be ensured that there is an equal amount of healthy data available outside of experimental settings. Therefore, one experiment will address this issue by modifying the ratio of data used from each dataset to obverse the difference that is made to the anomaly detection results.

D. EXPERIMENT 1 - TRAINING METHODS COMPARISON

Experiment 1 aims to provide an initial comparison of the various training approaches detailed in Section IV-C. To ensure experimental validity, the experiment will be repeated 10 times. However, each iteration will not utilise the same training data; instead, it will employ a randomly shuffled subset of data from each file to create a dataset comprising 4M samples. This approach further bolsters the validity of the proposed method and mitigates the risk of misrepresenting its performance by only validating it on a single subset of data.

Table 8 presents the Precision, Recall, and F1 score results for each experiment, as well as the average F1 score across the datasets. Table 9 displays the ANOVA for the Average F1 score results in 6. Additionally, Figure 4 features a box and whisker plot that visually compares the Average F1 scores from all 10 runs from each training approach, offering a representation of result spread and consistency.

E. EXPERIMENT 2 - VARYING DATA VOLUME

Experiment 2 aims to observe the effect of reducing the volume of training data on the performance of the anomaly detection model using the different training approaches. Similar to Experiment 1, the experiment will be repeated 10 times, and the dataset will be shuffled to ensure experimental validity. The experiment details for each test are shown in Table 10. The estimated FLOPs used for each number of samples are calculated using Equation 11 [46]:

$$FLOPs = 2 \cdot P \cdot 3 \cdot S \cdot E \tag{11}$$

TABLE 8. Experiment 1 results - A comparison of the performance of the anomaly detection model using all training approaches. The experiment key is shown in Table 5.

		Dataset A			Dataset B			
Training Approach	**Score**	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**	**Average F1**
T- Dataset A	Best (out of 10)	0.841	1.000	0.914	0.429	0.750	0.545	0.730
	Average	0.815	0.943	0.867	0.356	0.550	0.423	0.645
	Std Dvn	0.018	0.170	0.105	0.155	0.218	0.157	0.100
T- Dataset B	Best (out of 10)	0.821	0.711	0.762	1.000	0.500	0.667	0.714
	Average	0.806	0.600	0.687	0.477	0.625	0.515	0.601
	Std Dvn	0.025	0.061	0.048	0.212	0.256	0.164	0.088
TL- Dataset A to B	Best (out of 10)	0.818	1.000	0.900	0.429	0.750	0.545	0.723
	Average	0.819	0.823	0.813	0.379	0.625	0.448	0.630
	Std Dvn	0.016	0.168	0.089	0.056	0.256	0.096	0.046
TL- Dataset B to A	Best (out of 10)	0.813	0.967	0.883	0.500	0.750	0.600	0.742
	Average	0.800	0.671	0.695	0.407	0.625	0.440	0.568
	Std Dvn	0.022	0.298	0.196	0.242	0.340	0.185	0.170
MD	Best (out of 10)	0.818	1.000	0.900	0.429	0.750	0.545	0.723
	Average	0.818	0.999	0.899	0.350	0.675	0.454	0.677
	Std Dvn	0.001	0.003	0.002	0.086	0.195	0.108	0.054
DF (Proposed)	**Best (out of 10)**	**0.818**	**1.000**	**0.900**	**1.000**	**0.750**	**0.857**	**0.879**
	Average	**0.818**	**1.000**	**0.900**	**1.000**	**0.600**	**0.743**	**0.821**
	Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.122**	**0.093**	**0.047**
TL – DF to Dataset A	Best (out of 10)	0.818	1.000	0.900	0.400	1.000	0.571	0.736
	Average	0.809	0.890	0.834	0.357	0.900	0.507	0.670
	Std Dvn	0.020	0.221	0.134	0.024	0.122	0.021	0.070
TL – DF to Dataset B	Best (out of 10)	0.818	1.000	0.900	0.500	1.000	0.667	0.783
	Average	0.819	0.977	0.890	0.566	0.750	0.562	0.726
	Std Dvn	0.004	0.070	0.031	0.240	0.250	0.084	0.038

TABLE 9. One-way ANOVA test results for Experiment 1 - Training methods comparison.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F-stat	p-value
Between Groups	7	0.435	0.062	7.439	1.024×10^{-6}
Within Groups	72	0.601	0.008		
Total	79	1.036			

where P is the number of trainable parameters in the NN, N is number of training samples, and E is the number of epochs. Whilst the complexity of the model can undoubtedly affect the complexity of training, since the same model is used across all experiments, it will not be relevant for this calculation.

The tabulated results for experiment 2 can be found in Table 11. A summary of the results in the form of a box and whisker plot is illustrated in Figure 5. The One-way ANOVA table for the Average F1 scores is shown in Table 12.

F. EXPERIMENT 3 - VARYING DATASET RATIO

Experiment 3 aims to assess the performance of each training method with an imbalanced dataset containing a different number of samples from each dataset. The purpose of this experiment is to simulate a real-world environment, where the volume of data available from different sources will not be equal in many cases. Similar to Experiments 1 and 2, the experiment will be repeated 10 times, and the dataset will be shuffled to ensure experimental validity.

TABLE 10. Experiment settings for Experiment 2 - Varying data volume, including an estimation of the FLOPs used for training.

% Training data used	Number of Samples	Number of Epochs	NN Trainable Parameters	FLOPs used for training
100%	4,000,000	8	25,635	4.92×10^{12}
50%	2,000,000	8	25,635	2.46×10^{12}
25%	1,000,000	8	25,635	1.23×10^{11}
12.5%	500,000	8	25,635	6.15×10^{11}
6.25%	250,000	8	25,635	3.08×10^{11}

TABLE 11. Full results of experiment 2 - Varying data volume.

Percentage of data used	Experiment	Score	Dataset A Precision	Dataset A Recall	Dataset A F1	Dataset B Precision	Dataset B Recall	Dataset B F1	Average F1	% change of Average F1 from Baseline
100% (**Baseline**)	T- Dataset A	Average	0.815	0.943	0.867	0.356	0.550	0.423	0.645	N/A
		Std Dev	0.018	0.170	0.105	0.155	0.218	0.157	0.100	
	T- Dataset B	Average	0.806	0.600	0.687	0.477	0.625	0.515	0.601	N/A
		Std Dev	0.025	0.061	0.048	0.212	0.256	0.164	0.088	
	TL- Dataset A to B	Average	0.819	0.823	0.813	0.379	0.625	0.448	0.630	N/A
		Std Dev	0.016	0.168	0.089	0.056	0.256	0.096	0.046	
	TL- Dataset B to A	Average	0.800	0.671	0.695	0.407	0.625	0.440	0.568	N/A
		Std Dev	0.022	0.298	0.196	0.242	0.340	0.185	0.170	
	MD	Average	0.818	0.999	0.899	0.350	0.675	0.454	0.677	N/A
		Std Dev	0.001	0.003	0.002	0.086	0.195	0.108	0.054	
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.600**	**0.743**	**0.821**	N/A
		Std Dev	**0.000**	**0.000**	**0.000**	**0.000**	**0.122**	**0.093**	**0.047**	
	TL – DF to Dataset A	Average	0.809	0.890	0.834	0.357	0.900	0.507	0.670	N/A
		Std Dev	0.020	0.221	0.134	0.024	0.122	0.021	0.070	
	TL – DF to Dataset B	Average	0.819	0.977	0.890	0.566	0.750	0.562	0.726	N/A
		Std Dev	0.004	0.070	0.031	0.240	0.250	0.084	0.038	
50.00%	T- Dataset A	Average	0.810	0.937	0.859	0.449	0.750	0.545	0.702	8.86%
		Std Dev	0.025	0.190	0.123	0.125	0.250	0.146	0.088	
	T- Dataset B	Average	0.819	0.638	0.715	0.481	0.800	0.580	0.648	7.77%
		Std Dev	0.031	0.088	0.061	0.121	0.218	0.113	0.056	
	TL- Dataset A to B	Average	0.811	0.622	0.687	0.508	0.800	0.613	0.650	3.11%
		Std Dev	0.036	0.210	0.142	0.132	0.187	0.130	0.105	
	TL- Dataset B to A	Average	0.796	0.729	0.737	0.447	0.750	0.533	0.635	11.89%
		Std Dev	0.042	0.259	0.162	0.123	0.250	0.113	0.103	
	MD	Average	0.818	1.000	0.900	0.404	0.725	0.514	0.707	4.45%
		Std Dev	0.000	0.000	0.000	0.100	0.236	0.140	0.070	
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.600**	**0.735**	**0.818**	-0.46%
		Std Dev	**0.000**	**0.000**	**0.000**	**0.000**	**0.166**	**0.143**	**0.072**	
	TL – DF to Dataset A	Average	0.816	0.989	0.894	0.380	0.800	0.492	0.693	3.40%
		Std Dev	0.005	0.033	0.017	0.104	0.218	0.066	0.037	
	TL – DF to Dataset B	Average	0.816	0.951	0.872	0.522	0.850	0.629	0.751	3.46%
		Std Dev	0.008	0.147	0.084	0.189	0.255	0.190	0.108	
25.00%	T- Dataset A	Average	0.821	0.966	0.884	0.258	0.475	0.332	0.608	-5.70%
		Std Dev	0.007	0.103	0.049	0.133	0.261	0.172	0.086	
	T- Dataset B	Average	0.812	0.623	0.704	0.517	0.775	0.585	0.645	7.25%
		Std Dev	0.025	0.053	0.038	0.211	0.208	0.136	0.074	
	TL- Dataset A to B	Average	0.801	0.609	0.682	0.470	0.650	0.520	0.601	-4.67%
		Std Dev	0.040	0.165	0.119	0.196	0.122	0.097	0.066	
	TL- Dataset B to A	Average	0.783	0.620	0.664	0.277	0.400	0.311	0.488	-14.08%
		Std Dev	0.039	0.272	0.183	0.145	0.229	0.150	0.105	
	MD	Average	0.818	0.957	0.878	0.349	0.600	0.429	0.653	-3.46%
		Std Dev	0.006	0.123	0.066	0.063	0.200	0.085	0.046	
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.550**	**0.697**	**0.799**	**-2.78%**
		Std Dev	**0.000**	**0.000**	**0.000**	**0.000**	**0.150**	**0.131**	**0.065**	
	TL – DF to Dataset A	Average	0.796	0.768	0.758	0.417	0.750	0.496	0.627	-6.45%
		Std Dev	0.031	0.273	0.170	0.208	0.250	0.119	0.093	
	TL – DF to Dataset B	Average	0.814	0.906	0.850	0.420	0.725	0.504	0.677	-6.68%
		Std Dev	0.019	0.167	0.094	0.189	0.325	0.188	0.126	
12.50%	T- Dataset A	Average	0.818	1.000	0.900	0.341	0.425	0.355	0.627	-2.70%
		Std Dev	0.000	0.000	0.000	0.269	0.275	0.219	0.110	
	T- Dataset B	Average	0.792	0.510	0.614	0.519	0.625	0.546	0.580	-3.47%
		Std Dev	0.031	0.121	0.098	0.228	0.230	0.192	0.094	
	TL- Dataset A to B	Average	0.781	0.466	0.574	0.539	0.650	0.550	0.562	-10.78%
		Std Dev	0.039	0.136	0.119	0.196	0.200	0.132	0.062	
	TL- Dataset B to A	Average	0.818	1.000	0.900	0.355	0.300	0.288	0.594	4.62%
		Std Dev	0.000	0.000	0.000	0.353	0.245	0.222	0.111	
	MD	Average	0.818	0.998	0.899	0.279	0.450	0.337	0.618	-8.72%
		Std Dev	0.001	0.007	0.003	0.166	0.312	0.205	0.102	
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.550**	**0.690**	**0.795**	**-3.25%**
		Std Dev	**0.000**	**0.000**	**0.000**	**0.000**	**0.187**	**0.168**	**0.084**	
	TL – DF to Dataset A	Average	0.816	0.882	0.829	0.464	0.775	0.542	0.685	2.23%
		Std Dev	0.016	0.236	0.142	0.208	0.175	0.111	0.102	
	TL – DF to Dataset B	Average	0.822	0.953	0.880	0.553	0.700	0.561	0.721	-0.68%
		Std Dev	0.014	0.095	0.048	0.301	0.292	0.206	0.106	
6.25%	T- Dataset A	Average	0.818	1.000	0.900	0.418	0.600	0.468	0.684	6.14%
		Std Dev	0.000	0.000	0.000	0.249	0.255	0.202	0.101	
	T- Dataset B	Average	0.777	0.341	0.472	0.600	0.550	0.539	0.506	-15.87%
		Std Dev	0.039	0.049	0.047	0.268	0.269	0.217	0.094	
	TL- Dataset A to B	Average	0.757	0.382	0.504	0.565	0.750	0.624	0.564	-10.59%
		Std Dev	0.035	0.080	0.073	0.144	0.274	0.184	0.092	
	TL- Dataset B to A	Average	0.814	0.892	0.838	0.353	0.425	0.369	0.604	6.35%
		Std Dev	0.014	0.207	0.121	0.228	0.297	0.236	0.132	
	MD	Average	0.819	0.994	0.898	0.380	0.550	0.429	0.664	-1.97%
		Std Dev	0.002	0.017	0.006	0.070	0.218	0.085	0.042	
	DF (Proposed)	**Average**	**0.813**	**0.849**	**0.815**	**1.000**	**0.625**	**0.762**	**0.788**	**-4.04%**
		Std Dev	**0.019**	**0.223**	**0.134**	**0.000**	**0.125**	**0.095**	**0.083**	
	TL – DF to Dataset A	Average	0.812	0.876	0.830	0.451	0.850	0.563	0.696	3.88%
		Std Dev	0.011	0.200	0.120	0.130	0.200	0.092	0.084	
	TL – DF to Dataset B	Average	0.823	0.913	0.852	0.427	0.900	0.568	0.710	-2.19%
		Std Dev	0.012	0.189	0.109	0.056	0.166	0.049	0.056	

TABLE 12. One-way ANOVA test results for Experiment 2 - Varying data volume.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F-stat	p-value
Between Groups	39	2.389	0.061	6.341	1.110×10^{-16}
Within Groups	362	3.497	0.010		
Total	401	5.886			

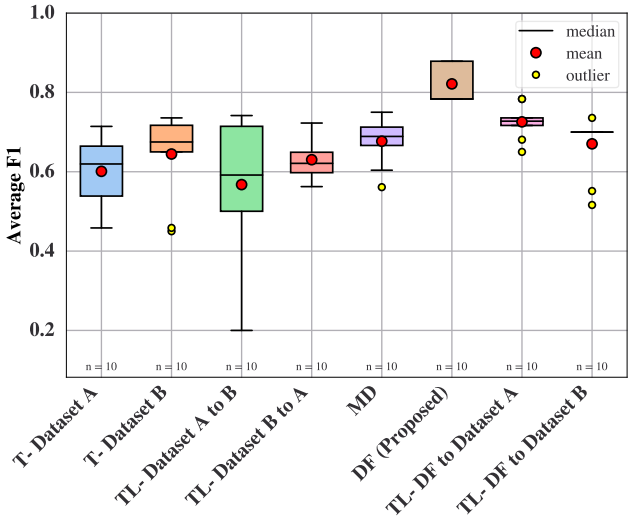


FIGURE 4. Box and whisker plot comparing the results for Experiment 1 - Training methods comparison.

Table 13 shows a breakdown of the experimental settings used. In the case of traditional training, the anomaly detector model will be trained on the reduced dataset, similar to experiment 2. However, transfer learning approaches will make use of both datasets.

The tabulated results for experiment 3 can be found in Table 14 for results from 10:90 to 50:50 (Dataset A: Dataset B), and in 15 for results from 60:40 to 90:10. The tabulated results are summarised in a box and whisker plot, shown in Figure 6. The One-way ANOVA table for the Average F1 scores is shown in Table 16.

TABLE 13. Experiment 3 - Varying dataset ratio details.

Dataset Ratio (A:B)	Dataset A samples	Dataset B samples
10:90	400,000	3,600,000
20:80	800,000	3,200,000
30:70	1,200,000	2,800,000
60:40	1,600,000	2,400,000
50:50	2,000,000	2,000,000
40:60	2,400,000	1,600,000
30:70	2,800,000	1,200,000
20:80	3,200,000	800,000
90:10	3,600,000	400,000

V. DISCUSSION

A. EXPERIMENT 1 - TRAINING METHODS COMPARISON ANALYSIS

Examining the experimental results from Experiment 1, as presented in Table 8, the DF method is empirically proven to consistently deliver the best performance in terms of F1 score for both Dataset A and Dataset B, as well as the average F1 score across both datasets. With the best Average F1 score of 0.879 and a 10-run average of 0.821, DF surpasses the other methods. These findings suggest that the DF approach effectively captures the salient features in both datasets, resulting in consistently strong performance across the datasets compared to the compared methods. Moreover, the results imply a considerable advantage in fusing the datasets, as the performance on individual datasets significantly exceeds that of models specialised in each respective datasets.

In comparison, the traditional training approach on Dataset A (T-Dataset A) exhibits better performance on Dataset A with a mean F1 score of 0.867 but suffers from poor results on Dataset B with a weaker score of 0.423, consequently lowering the average F1 score and making this model unsuitable for use across homogeneous datasets. The superior performance of the DF method on both Dataset A and Dataset B, along with the average F1 score across both datasets, underscores the algorithm's effectiveness in fulfilling the need for a single neural network adaptable to data from various sources within the same problem domain. This outcome aligns with the algorithm's proposed benefits, which aim to eliminate the need for multiple NNs and reduce data requirements from individual sources.

Furthermore, the results indicate that the traditional training approach, while effective for the dataset it was trained on, falls short in terms of generalisability across homogeneous datasets. In contrast, the DF method not only provides a more robust solution for handling data from multiple sources but also proves to be consistent in performance across multiple runs.

Utilising transfer learning, even when first training with the fused dataset, does not lead to better overall performance, even on the dataset that was trained on in the second phase. However, the preprocessing methods employed in the DF algorithm play a crucial role in enhancing performance, particularly for Dataset B, which required downsampling to meet the algorithm's requirements. This demonstrates the algorithm's adaptability to variations in data characteristics and its potential for handling real-world scenarios where data collection specifications may be inconsistent.

Although the performance across different training approaches on Dataset B is lower than on Dataset A, using transfer learning from the fused dataset to Dataset B yields results that are on par with training solely on Dataset B using the traditional approach. Simultaneously, this approach achieves strong performance on Dataset A. However, the DF algorithm still outperforms all transfer learning

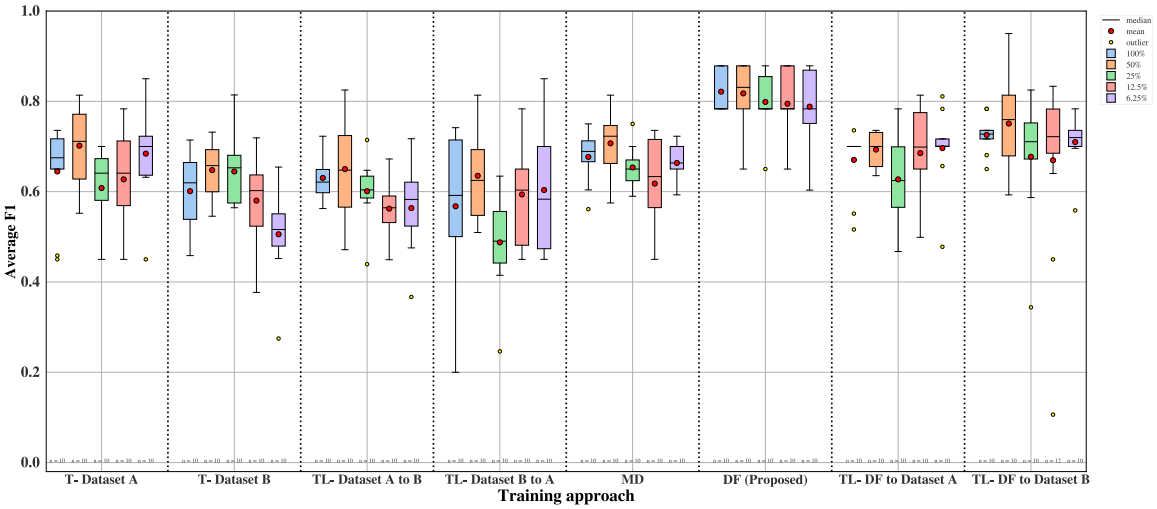


FIGURE 5. Box and whisker plot showcasing the impact of varying data volumes on each training method, grouped by training approach and plotted against average F1 score (higher is better).

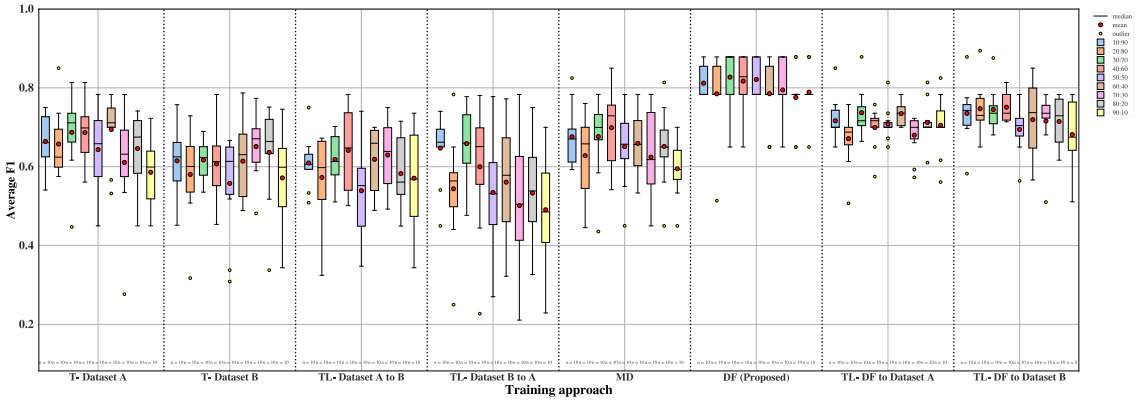


FIGURE 6. Box and whisker plot showcasing the impact of varying dataset ratios on each training method, grouped by training approach and plotted against average F1 score (higher is better).

approaches in terms of overall performance on both test datasets.

The box and whisker plot in Figure 4, illustrating the spread of the results, shows that the DF approach did not produce any outlying results from the 10 runs, indicating high levels of consistency. However, it is also evident that the transfer learning approaches exhibit the lowest deviation in results compared to a single training phase on one dataset, whether using traditional training approaches or the proposed DF method. This highlights the potential benefits of combining transfer learning with the DF approach to achieve even more consistent and reliable performance across datasets. However, the results show that transfer learning may not be the best approach, especially in this problem domain, to achieve the best performance.

The ANOVA Analysis presented in Table 9 demonstrates that the experimental results yield a statistically significant outcome, with a p-value of 1e-16. This highlights the relevance of the differences observed among the various training approaches.

B. EXPERIMENT 2 - VARYING DATA VOLUME ANALYSIS

The results of Experiment 2, presented in Table 11 and visualised in Figure 5, demonstrate that the DF approach significantly outperforms other training approaches across varying volumes of training data. Furthermore, when utilising only 6.25% of the training data, the model still surpasses other training approaches that use 100% of the data, achieving an Average F1 score of 0.788. As expected, most approaches experience decreased performance when using less data, with the proposed DF approach following this trend closely. However, it is worth noting that the performance reduction is minimal, with only a 4.04% drop compared to the baseline results.

Interestingly, not all approaches follow this pattern. For instance, transfer learning from the fused dataset to dataset B performs best when using 50% of the data, while transfer learning from the fused dataset to dataset A achieves optimal results with only 6.25% of the data. For these approaches, there does not appear to be a clear advantage to using more data, leading to the conclusion that, for training unsupervised

TABLE 14. Results for Experiment 3 - Dataset Ratio from 90:10 to 50:50 (Dataset A: Dataset B).

Data Ratio (A:B)	Experiment	Score	Dataset A Precision	Dataset A Recall	Dataset A F1	Dataset B Precision	Dataset B Recall	Dataset B F1	Average F1
10:90	T - Dataset A	Average	0.818	1.000	0.900	0.375	0.525	0.428	0.664
		Std Dvn	0.000	0.000	0.000	0.143	0.175	0.137	0.069
	T - Dataset B	Average	0.790	0.587	0.667	0.492	0.700	0.563	0.615
		Std Dvn	0.034	0.134	0.093	0.144	0.269	0.173	0.093
	TL- Dataset A to B	Average	0.799	0.663	0.710	0.394	0.800	0.509	0.609
		Std Dvn	0.038	0.206	0.137	0.059	0.292	0.121	0.062
	TL- Dataset B to A	Average	0.822	0.958	0.882	0.338	0.600	0.413	0.647
		Std Dvn	0.022	0.086	0.036	0.170	0.320	0.182	0.084
	MD	Average	0.820	0.828	0.814	0.495	0.775	0.538	0.676
		Std Dvn	0.028	0.180	0.095	0.206	0.284	0.126	0.074
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.575**	**0.724**	**0.812**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.115**	**0.087**	**0.044**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.456	0.875	0.533	0.717
		Std Dvn	0.000	0.000	0.000	0.208	0.256	0.112	0.056
	TL - DF to Dataset B	Average	0.815	0.911	0.852	0.582	0.800	0.619	0.735
		Std Dvn	0.015	0.168	0.092	0.227	0.218	0.104	0.070
20:80	T - Dataset A	Average	0.818	1.000	0.900	0.366	0.525	0.415	0.657
		Std Dvn	0.000	0.000	0.000	0.156	0.236	0.164	0.082
	T - Dataset B	Average	0.786	0.548	0.641	0.417	0.750	0.520	0.580
		Std Dvn	0.061	0.123	0.112	0.149	0.250	0.157	0.108
	TL- Dataset A to B	Average	0.809	0.632	0.700	0.388	0.575	0.445	0.573
		Std Dvn	0.029	0.155	0.094	0.163	0.297	0.177	0.107
	TL- Dataset B to A	Average	0.800	0.694	0.720	0.325	0.525	0.368	0.544
		Std Dvn	0.018	0.248	0.155	0.131	0.361	0.173	0.131
	MD	Average	0.824	0.812	0.810	0.429	0.500	0.447	0.628
		Std Dvn	0.040	0.176	0.102	0.126	0.158	0.113	0.099
	DF (Proposed)	**Average**	**0.813**	**0.953**	**0.873**	**1.000**	**0.550**	**0.697**	**0.785**
		Std Dvn	**0.017**	**0.140**	**0.082**	**0.000**	**0.150**	**0.131**	**0.100**
	TL - DF to Dataset A	Average	0.812	0.902	0.846	0.385	0.800	0.496	0.671
		Std Dvn	0.032	0.184	0.115	0.102	0.218	0.060	0.067
	TL - DF to Dataset B	Average	0.817	0.990	0.895	0.463	0.900	0.599	0.747
		Std Dvn	0.003	0.020	0.010	0.134	0.166	0.121	0.061
30:70	T - Dataset A	Average	0.818	0.999	0.899	0.415	0.625	0.475	0.687
		Std Dvn	0.001	0.003	0.002	0.169	0.301	0.189	0.095
	T - Dataset B	Average	0.816	0.633	0.713	0.476	0.675	0.521	0.617
		Std Dvn	0.016	0.034	0.023	0.197	0.195	0.085	0.051
	TL- Dataset A to B	Average	0.806	0.591	0.681	0.518	0.750	0.556	0.619
		Std Dvn	0.023	0.049	0.038	0.203	0.250	0.120	0.061
	TL- Dataset B to A	Average	0.812	0.822	0.803	0.498	0.650	0.514	0.659
		Std Dvn	0.039	0.212	0.128	0.197	0.278	0.133	0.085
	MD	Average	0.817	0.904	0.848	0.429	0.700	0.505	0.676
		Std Dvn	0.006	0.183	0.100	0.162	0.269	0.155	0.096
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.625**	**0.754**	**0.827**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.168**	**0.146**	**0.073**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.507	0.850	0.574	0.737
		Std Dvn	0.000	0.000	0.000	0.252	0.200	0.113	0.056
	TL - DF to Dataset B	Average	0.823	0.967	0.886	0.498	0.875	0.604	0.745
		Std Dvn	0.016	0.096	0.041	0.188	0.168	0.113	0.052
40:60	T - Dataset A	Average	0.817	0.990	0.895	0.499	0.550	0.478	0.687
		Std Dvn	0.005	0.030	0.016	0.213	0.269	0.153	0.076
	T - Dataset B	Average	0.803	0.631	0.700	0.432	0.675	0.514	0.607
		Std Dvn	0.028	0.144	0.090	0.136	0.275	0.164	0.099
	TL- Dataset A to B	Average	0.795	0.678	0.715	0.514	0.725	0.568	0.641
		Std Dvn	0.043	0.219	0.147	0.119	0.284	0.139	0.101
	TL- Dataset B to A	Average	0.793	0.670	0.707	0.394	0.725	0.492	0.600
		Std Dvn	0.046	0.244	0.164	0.168	0.344	0.206	0.154
	MD	Average	0.815	0.979	0.889	0.415	0.725	0.509	0.699
		Std Dvn	0.010	0.056	0.030	0.131	0.284	0.170	0.092
	DF (Proposed)	**Average**	**0.818**	**0.998**	**0.899**	**1.000**	**0.600**	**0.735**	**0.817**
		Std Dvn	**0.001**	**0.007**	**0.003**	**0.000**	**0.166**	**0.143**	**0.071**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.435	0.800	0.499	0.700
		Std Dvn	0.000	0.000	0.000	0.196	0.292	0.099	0.049
	TL - DF to Dataset B	Average	0.820	0.998	0.900	0.476	0.900	0.602	0.751
		Std Dvn	0.007	0.004	0.003	0.104	0.166	0.069	0.035
50:50 (**Baseline**)	T - Dataset A	Average	0.818	1.000	0.900	0.335	0.475	0.387	0.643
		Std Dvn	0.000	0.000	0.000	0.169	0.261	0.198	0.099
	T - Dataset B	Average	0.802	0.544	0.648	0.362	0.700	0.467	0.558
		Std Dvn	0.023	0.038	0.032	0.208	0.367	0.249	0.125
	TL- Dataset A to B	Average	0.791	0.589	0.658	0.356	0.550	0.420	0.539
		Std Dvn	0.038	0.213	0.140	0.153	0.245	0.165	0.123
	TL- Dataset B to A	Average	0.798	0.748	0.746	0.278	0.425	0.323	0.535
		Std Dvn	0.027	0.283	0.178	0.172	0.317	0.206	0.141
	MD	Average	0.817	0.994	0.897	0.326	0.550	0.405	0.651
		Std Dvn	0.003	0.017	0.008	0.162	0.269	0.193	0.095
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.600**	**0.743**	**0.821**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.122**	**0.093**	**0.047**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.475	0.775	0.518	0.709
		Std Dvn	0.000	0.000	0.000	0.205	0.261	0.082	0.041
	TL - DF to Dataset B	Average	0.818	0.929	0.866	0.444	0.675	0.523	0.695
		Std Dvn	0.006	0.122	0.059	0.103	0.160	0.084	0.055

TABLE 15. Results for Experiment 3 - Dataset Ratio from 40:60 to 10:90 (Dataset A: Dataset B).

Data Ratio (A:B)	Experiment	Score	Dataset A Precision	Dataset A Recall	Dataset A F1	Dataset B Precision	Dataset B Recall	Dataset B F1	Average F1
60:40	T - Dataset A	Average	0.820	0.969	0.886	0.418	0.675	0.502	0.694
		Std Dvn	0.008	0.083	0.036	0.090	0.251	0.129	0.078
	T - Dataset B	Average	0.810	0.582	0.677	0.547	0.650	0.552	0.614
		Std Dvn	0.019	0.052	0.040	0.242	0.278	0.183	0.095
	TL- Dataset A to B	Average	0.815	0.721	0.757	0.453	0.625	0.480	0.619
		Std Dvn	0.017	0.150	0.080	0.227	0.280	0.147	0.084
	TL- Dataset B to A	Average	0.801	0.633	0.675	0.410	0.575	0.447	0.561
		Std Dvn	0.035	0.285	0.178	0.247	0.317	0.217	0.140
	MD	Average	0.828	0.903	0.853	0.359	0.675	0.465	0.659
		Std Dvn	0.018	0.168	0.084	0.120	0.195	0.138	0.076
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.525**	**0.670**	**0.785**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.175**	**0.158**	**0.079**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.477	0.900	0.570	0.735
		Std Dvn	0.000	0.000	0.000	0.199	0.200	0.061	0.030
	TL - DF to Dataset B	Average	0.818	0.904	0.851	0.543	0.800	0.588	0.719
		Std Dvn	0.008	0.159	0.082	0.202	0.269	0.137	0.095
70:30	T - Dataset A	Average	0.805	0.843	0.807	0.350	0.525	0.415	0.611
		Std Dvn	0.029	0.241	0.153	0.152	0.261	0.184	0.130
	T - Dataset B	Average	0.809	0.636	0.709	0.487	0.800	0.594	0.651
		Std Dvn	0.026	0.108	0.069	0.162	0.218	0.163	0.077
	TL- Dataset A to B	Average	0.800	0.671	0.719	0.434	0.750	0.541	0.630
		Std Dvn	0.020	0.182	0.110	0.083	0.224	0.122	0.088
	TL- Dataset B to A	Average	0.778	0.557	0.618	0.325	0.500	0.385	0.501
		Std Dvn	0.038	0.282	0.185	0.174	0.316	0.216	0.177
	MD	Average	0.823	0.991	0.899	0.308	0.425	0.350	0.624
		Std Dvn	0.013	0.027	0.004	0.190	0.317	0.231	0.116
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.550**	**0.690**	**0.795**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.187**	**0.168**	**0.084**
	TL - DF to Dataset A	Average	0.813	0.962	0.881	0.355	0.800	0.479	0.680
		Std Dvn	0.010	0.069	0.036	0.038	0.245	0.084	0.052
	TL - DF to Dataset B	Average	0.815	0.954	0.877	0.423	0.850	0.555	0.716
		Std Dvn	0.006	0.100	0.051	0.055	0.229	0.104	0.073
80:20	T - Dataset A	Average	0.824	0.974	0.892	0.338	0.525	0.401	0.646
		Std Dvn	0.012	0.063	0.025	0.175	0.284	0.193	0.092
	T - Dataset B	Average	0.801	0.610	0.688	0.561	0.725	0.585	0.636
		Std Dvn	0.021	0.111	0.069	0.243	0.325	0.219	0.119
	TL- Dataset A to B	Average	0.793	0.611	0.681	0.454	0.575	0.485	0.583
		Std Dvn	0.022	0.161	0.098	0.141	0.251	0.152	0.093
	TL- Dataset B to A	Average	0.787	0.680	0.696	0.371	0.450	0.370	0.533
		Std Dvn	0.039	0.309	0.206	0.261	0.269	0.194	0.124
	MD	Average	0.818	1.000	0.900	0.342	0.500	0.403	0.651
		Std Dvn	0.000	0.000	0.000	0.159	0.250	0.188	0.094
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.500**	**0.651**	**0.776**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.158**	**0.146**	**0.073**
	TL - DF to Dataset A	Average	0.815	0.959	0.878	0.411	0.875	0.547	0.712
		Std Dvn	0.008	0.123	0.067	0.081	0.168	0.078	0.052
	TL - DF to Dataset B	Average	0.817	0.994	0.897	0.580	0.625	0.532	0.715
		Std Dvn	0.003	0.017	0.008	0.231	0.256	0.115	0.059
90:10	T - Dataset A	Average	0.818	0.942	0.871	0.251	0.400	0.301	0.586
		Std Dvn	0.016	0.138	0.079	0.137	0.255	0.173	0.090
	T - Dataset B	Average	0.790	0.513	0.616	0.511	0.600	0.527	0.572
		Std Dvn	0.023	0.103	0.090	0.252	0.300	0.228	0.115
	TL- Dataset A to B	Average	0.791	0.598	0.668	0.434	0.600	0.473	0.571
		Std Dvn	0.043	0.191	0.135	0.248	0.320	0.220	0.125
	TL- Dataset B to A	Average	0.782	0.687	0.703	0.257	0.325	0.278	0.491
		Std Dvn	0.040	0.291	0.193	0.154	0.195	0.161	0.133
	MD	Average	0.819	0.977	0.890	0.265	0.375	0.300	0.595
		Std Dvn	0.004	0.070	0.031	0.116	0.202	0.136	0.068
	DF (Proposed)	**Average**	**0.818**	**1.000**	**0.900**	**1.000**	**0.525**	**0.678**	**0.789**
		Std Dvn	**0.000**	**0.000**	**0.000**	**0.000**	**0.135**	**0.119**	**0.060**
	TL - DF to Dataset A	Average	0.818	1.000	0.900	0.444	0.750	0.511	0.705
		Std Dvn	0.000	0.000	0.000	0.235	0.250	0.143	0.072
	TL - DF to Dataset B	Average	0.811	0.940	0.862	0.618	0.650	0.529	0.696
		Std Dvn	0.020	0.180	0.113	0.259	0.300	0.117	0.085

TABLE 16. One-way ANOVA test results for Experiment 3 - Varying dataset ratio.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F-stat	*p*-value
Between Groups	71	4.546	0.064	6.935	1.110×10^{-16}
Within Groups	646	5.964	0.009		
Total	717	10.510			

anomaly detectors, using more data is not always beneficial. While this may not hold true for other tasks such as fault classification, those tasks are beyond the scope of the present study and will be explored in future works.

The box and whisker plot in Figure 5 reveals that the performance of the anomaly detection model trained with DF is less consistent and has a larger spread when using less data. However, examining the results from other training approaches shows that this is not always the case. In some instances, such as Transfer Learning from Dataset A to Dataset B, the spread is largest when training with the full dataset. Additionally, as the ANOVA table in Table 12 shows, the p-value for this experiment suggests the results are statistically significant.

By outperforming all other models while using only a small fraction of the training data, the DF method addresses the common challenge of not having sufficient training data from each data source. In this experiment, as shown in Table 10, the estimated FLOPs needed for training the model with 6.25% of the data dramatically decreased from 4.92×10^{12} to 3.08×10^{11}, representing a 93.7% reduction in computational power. This significant decrease is especially noteworthy when contrasted with the minor 4.04% reduction in performance. The DF approach effectively utilises less data, showcasing its potential to contribute to more sustainable and environmentally friendly AI development. Aligned with the principles of Green AI, which emphasise efficiency and reducing the environmental impact of training AI models, the results of Experiment 2 highlight the superior performance of DF. Its crucial implications for real-world applications demonstrate its ability to address the issue of limited training data while promoting more sustainable practices in AI development, providing a valuable contribution towards SDG goals 12 and 13 with reduced computational power requirements and hence energy consumption.

C. EXPERIMENT 3 - VARYING DATASET RATIO ANALYSIS

The results of the experiment, shown in Table 14 and Table 15, reveal that the DF algorithm outperforms all other training approaches in terms of Average F1 score. The best performance is achieved with a 30:70 ratio (Dataset A: Dataset B), resulting in an Average F1 score of 0.827, closely followed by the baseline experiment (50:50) with an Average F1 of 0.821. In comparison, the next best performing training approach was transfer learning from the fused dataset to Dataset B (TF - DF to Dataset B), which had an Average F1 score of 0.751. The ANOVA in Table 16 confirms that these values are statistically significant.

The box and whisker plot in Figure 6 indicates that the transfer learning approaches involving the fused dataset generally exhibit less spread compared to other methods. Furthermore, the spread of the DF results appears to be the most consistent across different experimental settings. One might hypothesise that the box plot trend would indicate an increase in generalised performance as the balance between samples from each dataset increases. Interestingly, this is not the case.

While the results empirically demonstrate that an imbalance in datasets mostly does not affect the stability of the results for DF, a clear trend is observed, wherein the average F1 score decreases as more of Dataset A is used for training. This trend is more evident in some of the other approaches, such as T-Dataset A, TL-Dataset B to A, and the mixed dataset (MD). For instance, training with the Mixed Dataset at a 40:60 ratio yields an F1 score of 0.889 on Dataset A, 0.509 on Dataset B, and an overall average of 0.699. Conversely, training with a 90:10 ratio results in an F1 score of 0.890 on Dataset A, 0.300 on Dataset B, and an overall average of 0.595. These results show that the performance on Dataset A can be maintained with less data while improving the performance on Dataset B with more data.

The observed trend leads to an intriguing conclusion: there is a higher benefit to training with Dataset B compared to Dataset A to achieve a more generalised anomaly detection performance. Although both Dataset A and Dataset B represent healthy data, Dataset B exhibits a higher level of noise, which can be clearly seen in Figure 2h, the raw FFT of the signal. Additionally, there is a slightly higher level of spread on the PCA shown in Figure 3. These observations suggest that Dataset B is a more difficult dataset to learn and requires more training data compared to Dataset A. This also implies that a larger variety of healthy data in the training dataset contributes to an overall better anomaly detection performance.

D. FURTHER REMARKS AND LIMITATIONS

In the context of Transfer Learning, it is worth noting that better results on the dataset trained on in the second phase are not guaranteed. The results of the experiments demonstrate that swapping the transfer learning training phases produces different outcomes, which indicates that there is no clear approach to determine which dataset should be used in each phase without conducting additional testing. Dataset Fusion mitigates this issue by training on both datasets simultaneously.

The conclusions drawn from Experiment 3 suggest that there is a greater benefit to training on a higher number of samples from Dataset B, as opposed to Dataset A, in order to achieve a more generalised performance. However, identifying this preference presents a challenge, as a similar experiment must be conducted to reach this determination. Despite this issue, the DF approach offers a solution by providing a more stable performance across multiple experimental settings. This advantage becomes particularly significant in practical applications where time and resource constraints may render extensive experimentation unfeasible. By addressing these concerns and offering more consistent results, DF shows potential as an effective method for training unsupervised anomaly detection models, particularly in situations where the optimal dataset and training phase cannot be easily determined. The robust performance of the DF approach, combined with its ability to accommodate various experimental settings, positions it as a valuable tool for practical use cases. It is currently unclear whether the findings from these experiments will hold true for other tasks, such as fault classification. Addressing this question is beyond the scope of this paper, but it presents an intriguing avenue for future research.

In tasks requiring long-term degradation or forecasting, where the sequential nature of the data is paramount, our batching and fusion method may not be optimal. In such tasks, training data are primarily aimed at identifying the trend of data behaviour over time from a specific source, which may be obscured when alternately mixing datasets. However, for other tasks where the focus is on identifying data source invariant changes - those consistent regardless of the source - our DF algorithm proves to be highly beneficial. This methodology enables the creation of a balanced training set that is more representative of the population distribution of the problem at hand. This approach, therefore, demonstrates substantial value in tasks such as classification or anomaly detection, where the detection of irregularities or deviations is more critical than following a specific trend. In contrast, in forecasting tasks, where the continuity and trend of a single data source are paramount, alternative methods preserving the sequential integrity within each source may yield better models. Future work may explore additional strategies for DF tailored specifically for different task types, offering further flexibility and adaptability in tackling diverse data science problems.

The homogeneity assumption in the proposed dataset fusion technique presents a limitation, particularly when handling real-world datasets that often exhibit some degree of non-homogeneity such as in this case, motors of different sizes and rotational speeds. To mitigate this challenge, future research could look into the incorporation of domain adaptation methods, which allow for the handling of datasets that differ slightly in their distributions or characteristics.

VI. CONCLUSION

This paper presents a time-series dataset composition approach called the DF algorithm, designed to address challenges in achieving generalised anomaly detection performance across multiple homogeneous data sources. The proposed algorithm was validated using a case study involving motor current signals, demonstrating that the fused dataset retains salient features from both source datasets while clustering in the middle of both datasets when PCA is applied. The algorithm was then tested on an anomaly detection task and compared to conventional training approaches, with empirical results showing that the DF algorithm significantly outperforms other methods in terms of average performance across both datasets.

Additionally, further experiments were conducted to assess the performance of the proposed approach under non-ideal conditions. Experimental results indicate that the DF approach remains superior even when reducing the number of data samples, with only a 4.04% reduction in performance despite using only 6.25% of the training data, resulting in a 93.7% reduction in computational power required for training. When evaluating the model's performance with imbalanced numbers of samples from each dataset, the proposed approach proved stable across different sample ratios. These findings highlight significant benefits in the context of Green AI, which emphasises sustainable AI model development, as well as practical feasibility due to the algorithm's resilience under non-ideal conditions.

Several research directions could be explored based on the present study. For instance, future works might investigate the applicability of the algorithm for training classifier models or examine whether using the frequency domain representation of the fused dataset could enhance performance. Furthermore, it would be worthwhile to explore the relationship between dataset ''complexity'' and the ''usefulness'' of data from each fused dataset, enabling the development of a more systematic approach to dataset fusion. Furthermore, adapting the proposed method to enable compatibility with non-homogeneous datasets could potentially be accomplished using domain adaptation techniques, which would further strengthen the applicability of the proposed method.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Voltvision for their substantial support of this research. Their sponsorship has significantly contributed to the progress and achievements of this work.

REFERENCES

[1] G. Jin, X. Yi, L. Zhang, L. Zhang, S. Schewe, and X. Huang, ''How does weight correlation affect the generalisation ability of deep neural networks?'' in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2020, pp. 21346–21356.

[2] D. Partridge and N. Griffith, ''Strategies for improving neural net generalisation,'' *Neural Comput. Appl.*, vol. 3, no. 1, pp. 27–37, Mar. 1995.

[3] S. Lawrence and C. L. Giles, ''Overfitting and neural networks: Conjugate gradient and backpropagation,'' in *Proc. Int. Joint Conf. Neural Netw.*, 2000, pp. 114–119.

[4] A. Elhalwagy and T. Kalganova, ''Multi-channel LSTM-capsule autoencoder network for anomaly detection on multivariate data,'' *Appl. Sci.*, vol. 12, no. 22, p. 11393, Nov. 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/22/11393

[5] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, ''Dropout: A simple way to prevent neural networks from overfitting,'' *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[6] R. Caruana, S. Lawrence, and L. Giles, ''Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 402–408.

[7] H. Wang, C. Qin, Y. Zhang, and Y. Fu, ''Neural pruning via growing regularization,'' 2020, *arXiv:2012.09243*.

[8] G. An, ''The effects of adding noise during backpropagation training on a generalization performance,'' *Neural Comput.*, vol. 8, no. 3, pp. 643–674, Apr. 1996. [Online]. Available: https://direct.mit.edu/neco/article/8/3/643/5975/The-Effects-of-Adding-Noise-During-Backpropagation

[9] J. Chen, X. Feng, L. Jiang, and Q. Zhu, ''State of charge estimation of lithium-ion battery using denoising autoencoder and gated recurrent unit recurrent neural network,'' *Energy*, vol. 227, Jul. 2021, Art. no. 120451.

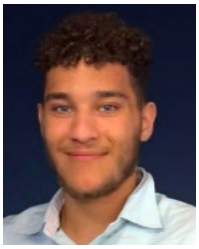
[10] G. Salimi-Khorshidi, G. Douaud, C. F. Beckmann, M. F. Glasser, L. Griffanti, and S. M. Smith, ''Automatic denoising of functional MRI data: Combining independent component analysis and hierarchical fusion of classifiers,'' *NeuroImage*, vol. 90, pp. 449–468, Apr. 2014.

[11] C. Shorten and T. M. Khoshgoftaar, ''A survey on image data augmentation for deep learning,'' *J. Big Data*, vol. 6, no. 1, pp. 1–48, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[12] M. A. Tanner and W. H. Wong, ''The calculation of posterior distributions by data augmentation,'' *J. Amer. Stat. Assoc.*, vol. 82, no. 398, pp. 528–540, Jun. 1987.

[13] A. Byerly and T. Kalganova, ''Towards an analytical definition of sufficient data,'' 2022, *arXiv:2202.03238*.

[14] A. Byerly and T. Kalganova, "Class density and dataset quality in high-dimensional, unstructured data," 2022, *arXiv:2202.03856*.
[15] H. Raza, G. Prasad, and Y. Li, "Dataset shift detection in non-stationary environments using EWMA charts," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 3151–3156.
[16] T. Guo, Z. Xu, X. Yao, H. Chen, K. Aberer, and K. Funaya, "Robust online time series prediction with recurrent neural networks," in *Proc. IEEE Int. Conf. Data Sci. Adv. Anal. (DSAA)*, Oct. 2016, pp. 816–825.
[17] H. Raza, G. Prasad, and Y. Li, "Adaptive learning with covariate shift-detection for non-stationary environments," in *Proc. 14th U.K. Workshop Comput. Intell. (UKCI)*, Sep. 2014, pp. 1–8.
[18] Z. Cai, O. Sener, and V. Koltun, "Online continual learning with natural distribution shifts: An empirical study with visual data," 2021, *arXiv:2108.09020*.
[19] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Dec. 2015, pp. 1135–1143.
[20] X. Qian and D. Klabjan, "A probabilistic approach to neural network pruning," 2021, *arXiv:2105.10065*.
[21] S. Urolagin and N. V. S. Reddy, "Generalization capability of artificial neural network incorporated with pruning method," in *Proc. Int. Conf. Adv. Comput., Netw. Secur.*, P. S. Thilagam, A. R. Pais, K. Chandrasekaran, and N. Balakrishnan, Eds. Heidelberg, Germany: Springer, 2012, pp. 171–178.
[22] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, "The effects of data quality on machine learning performance," 2022, *arXiv:2207.14529*.
[23] Z. C. Lipton, Y. X. Wang, and A. J. Smola, "Detecting and correcting for label shift with black box predictors," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 7, 2018, pp. 4887–4897.
[24] S. Rabanser, S. Günnemann, and Z. C. Lipton, "Failing loudly: An empirical study of methods for detecting dataset shift," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1394–1406.
[25] R. Wu, C. Guo, Y. Su, and K. Q. Weinberger, "Online adaptation to label distribution shift," 2021, *arXiv:2107.04520*.
[26] B. Neal, S. Mittal, A. Baratin, V. Tantia, M. Scicluna, S. Lacoste-Julien, and I. Mitliagkas, "A modern take on the bias-variance tradeoff in neural networks," 2018, *arXiv:1810.08591*.
[27] L. Fan, F. Zhang, H. Fan, and C. Zhang, "Brief review of image denoising techniques," *Vis. Comput. Ind., Biomed.*, vol. 2, no. 1, pp. 1–12, Dec. 2019.
[28] D. Amodei and D. Hernandez. (May 2018). *AI and Compute*. [Online]. Available: https://openai.com/research/ai-and-compute
[29] D. Yalalov. (Feb. 2023). *AI Model Training Costs Are Expected to Rise From $100 Million to $500 Million by 2030*. [Online]. Available: https://mpost.io/ai-model-training-costs-are-expected-to-rise-from-100-million-to-500-million-by-2030/
[30] S. Bozinovski, "Reminder of the first paper on transfer learning in neural networks, 1976," *Informatica*, vol. 44, no. 3, pp. 291–302, Sep. 2020.
[31] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.
[32] Y. Xian and H. Hu, "Enhanced multi-dataset transfer learning method for unsupervised person re-identification using co-training strategy," *IET Comput. Vis.*, vol. 12, no. 8, pp. 1219–1227, Dec. 2018, doi: 10.1049/iet-cvi.2018.5103.
[33] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1623–1637, Mar. 2022.
[34] Y. Yao, Y. Wang, Y. Guo, J. Lin, H. Qin, and J. Yan, "Cross-dataset training for class increasing object detection," 2020, *arXiv:2001.04621*.
[35] X. Zhou, V. Koltun, and P. Krähenbühl, "Simple multi-dataset detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 7561–7570.
[36] X. Jin, Y. Park, D. C. Maddix, H. Wang, and Y. Wang, "Domain adaptation for time series forecasting via attention sharing," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 10280–10297.
[37] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 1, no. 1, pp. 2030–2096, 2016.
[38] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool, "Domain adaptive faster R-CNN for object detection in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3339–3348.
[39] A. E. Treml, R. A. Flauzino, M. Suetake, and N. A. R. Maciejewski, "Experimental database for detecting and diagnosing rotor broken bar in a three-phase induction motor," IEEE Dataport, Sep. 2020, doi: 10.21227/fmnm-bn95.
[40] R. G. C. Cunha, E. T. D. S. Junior, and C. M. D. S. Medeiros. *Inter-Turn Short-Circuit in Induction Motor*. Accessed: Apr. 23, 2022. [Online]. Available: https://www.kaggle.com/datasets/4d260937c50d483e5a888c3d587180a3c53c4c962313c52829de7ec45c383b94
[41] S. Ahmad, K. Styp-Rekowski, S. Nedelkoski, and O. Kao, "Autoencoder-based condition monitoring and anomaly detection method for rotating machines," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Mar. 2020, pp. 4093–4102.
[42] B. Maschler, T. Knodel, and M. Weyrich, "Towards deep industrial transfer learning for anomaly detection on time series data," in *Proc. 26th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2021, pp. 1–8.
[43] C. R. Harris et al., "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.
[44] W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, S. van der Walt and J. Millman, Eds. 2010, pp. 56–61, doi: 10.25080/Majora-92bf1922-00a.
[45] N. Chinchor, "MUC-4 evaluation metrics," in *Proc. 4th Conf. Message Understand.*, 1992, pp. 22–29, doi: 10.3115/1072064.1072067.
[46] J. Sevilla, L. Heim, M. Hobbhahn, T. Besiroglu, A. Ho, and P. Villalobos. (2022). *Estimating Training Compute of Deep Learning Models*. Accessed: Apr. 23, 2023. [Online]. Available: https://epochai.org/blog/estimating-training-compute



AYMAN ELHALWAGY (Graduate Student Member, IEEE) received the B.Eng. degree (Hons.) in electronic and computer engineering from Brunel University London, Uxbridge, in 2021, where he is currently pursuing the Ph.D. degree in electronic and electrical engineering. His current research interests include applied machine learning, fault detection and classification, and intelligent systems.

TATIANA KALGANOVA (Member, IEEE) received the B.Sc. (Hons.) and Ph.D. degrees. She is currently a Professor of intelligent systems and the Electronic and Computer Engineering Postgraduate Research Director of Brunel University London, Uxbridge, U.K. She has more than 25 years of experience in the design and implementation of applied intelligent systems.