


RESEARCH ARTICLE

The transfer learning of uncertainty quantification for industrial plant fault diagnosis system design

J. Blair^{1,2} , O. Amin^{1,2}, B. D. Brown¹, S. McArthur¹, A. Forbes² and B. Stephen¹

¹Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, UK

²Data Science Department, National Physical Laboratory, Teddington, UK

Corresponding author: J. Blair; E-mail: j.blair@strath.ac.uk

Received: 08 March 2024; **Revised:** 19 September 2024; **Accepted:** 31 October 2024

Keywords: uncertainty; data pipelines; machine learning; fault diagnosis; engineering

Abstract

The performance and confidence in fault detection and diagnostic systems can be undermined by data pipelines that feature multiple compounding sources of uncertainty. These issues further inhibit the deployment of data-based analytics in industry, where variable data quality and lack of confidence in model outputs are already barriers to their adoption. The methodology proposed in this paper supports trustworthy data pipeline design and leverages knowledge gained from one fully-observed data pipeline to a similar, under-observed case. The transfer of uncertainties provides insight into uncertainty drivers without repeating the computational or cost overhead of fully redesigning the pipeline. A SHAP-based human-readable explainable AI (XAI) framework was used to rank and explain the impact of each choice in a data pipeline, allowing the decoupling of positive and negative performance drivers to facilitate the successful selection of highly-performing pipelines. This empirical approach is demonstrated in bearing fault classification case studies using well-understood open-source data.

Impact statement

This work contributes to identifying, explaining, and leveraging known uncertainty drivers to the design of similar industrial fault detection and diagnostic system data pipelines in three ways:

- Demonstrating how the uncertainty from the imposed pipeline design constraints can compound to create improved or deteriorated performance in fault diagnostic systems.
- Identifying highly or under-performing system design options using a human-readable XAI framework, leading to better or worse system performance, respectively.
- Identification of uncertainty sources learned from a fully-observed pipeline system design (source) to an unseen pipeline (target). This allows insight into the target system's pipeline without the computational overhead of fully observing all possible pipelines or fully redesigning the new pipeline.

1. Introduction

In energy sectors, downtime in key assets can be costly for operators, incurring lost generation revenues and associated penalties. To mitigate these costs, many operators have turned to prognostic and health management (PHM) and condition monitoring (CM) techniques to monitor the health of assets more closely [1]. PHM and CM techniques can utilize sensor data and machine learning (ML) models to detect

the onset of faults [2], and when these diagnostic approaches are applied to critical operational components, they can decrease maintenance efforts and expedite return to service—but only if they are known to deliver high predictive accuracy. Data pipelines describe the flow and transformation of data through important stages in a data acquisition system, from collection by the measurement system through to life-long storage and management. Accuracy in data pipelines can be eroded by data quality, which can be circumvented with appropriate system design and calibration [3].

Many industries require a high level of transparency when utilizing data analytics to ensure plant operation is well informed and traceable to trustworthy evidence for subsequent reasoning and decision-making. It is vital to know when to trust the output of analytics and to understand where the largest uncertainty contributions occur along the data pipeline. Knowledge of the uncertainty present in a data pipeline de-risks the system by providing the operator with confidence in the quality of data and decisions being made by a fault detection and diagnostic system. The motivation for this paper is to demonstrate how the uncertainty associated with data pipeline design choices can be identified, quantified (in terms of the choice's contribution to analytic performance), and leveraged as transferable knowledge when designing pipelines for similar engineering applications.

Industrial fault diagnosis has evolved through several stages. Initially, fault diagnosis relied entirely on expert knowledge until data collection and processing became more widely available. Access to historical, labeled data allowed the development and resultant popularity of Machine Learning-based, and then Deep Learning-based (DL) approaches as Big Data became possible. Both ML and DL approaches are heavily impacted by the availability of data and labels and incur high computational overheads when retraining on new datasets, pushing the need for transfer learning (TL). Transfer learning can allow available data or pre-trained models to be adapted to a new compatible task or application [4].

1.1. Uncertainty quantification in data pipelines

Uncertainty is often characterized into two broad categories: Type A (aleatoric), where the uncertainties are driven by randomness, or Type B (epistemic), where the uncertainties are driven by a lack of knowledge and so can feasibly be driven down through improved understanding or measurement [5–7]. Examples of this in a data acquisition system could be sensor noise (aleatoric), where further measurement can reduce the uncertainty to a certain extent by averaging out random effects, versus increased fault observations (epistemic), where more examples would provide a ML model with more characteristic information [8].

The general components of the data pipeline in industrial process condition monitoring, shown in Figure 1, are the sensor systems, data communications channels, data processing and storage, data analytics, and decision metrics. At each of these stages, uncertainty sources impact the quality of the derived information. Quality issues can arise from sensor calibration issues [9], the method and standards for the data communication channels [10], and possible problems due to data storage and bandwidth limitations [11], all of which increase uncertainty contributions associated with the data. These

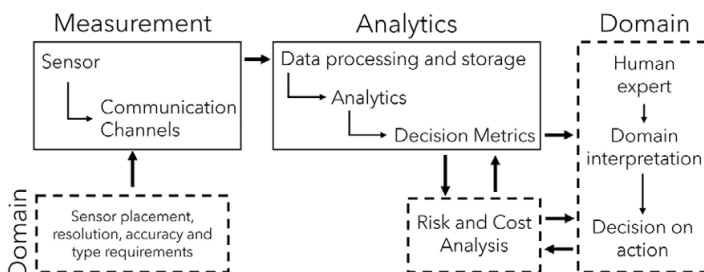


Figure 1. Illustration of the major stages and flow of data in a simplified industrial data acquisition pipeline.

uncertainty contributions often cannot be accurately corrected for, resulting in a change in the confidence interval on the final output [12]. Additionally, many ML models cannot attribute a confidence margin to an output decision, which makes risk and cost assessments involving the output of machine learning models problematic [13]. This inability to quantify prediction confidence results in either a lack of adoption of machine learning techniques or poorly informed decisions being made due to the poor quality of the data sources.

Many legacy industrial process plants (such as power plants), even when retrofitted with modern digital data collection and acquisition systems, are likely to have suboptimal data pipelines, which limits the information available for diagnostic and fault detection systems. Hence, the automated design of data pipelines for machine learning applications in data science spaces is an active area of study [14–16]. This has arisen due to the increasing popularity and desire for “off the shelf” machine learning techniques. Feurer et al. [14] proposes “Auto-sklearn” and improves on automated pipeline design by taking account of historical performance and constructing ensembles evaluated during Bayesian optimization. Liu et al. [15] builds on ADMM (Alternating Direction Method of Multipliers) optimization by decomposing the pipeline optimization problem into easier sub-problems and incorporating constraints on the objective, resulting in performance improvements. Yang et al. [16] introduces a new greedy design protocol to gather information about a new pipeline dataset efficiently and proposes “TensorOboe,” which uses low rank tensor decomposition as a surrogate for efficient pipeline search. However, these state-of-the-art techniques still require calibration, albeit with less expert intervention. Additionally, these techniques tend to focus solely on delivering the “best” performing pipeline within a given search space but do not provide the developer with an understanding of **why** the proposed pipelines should be accepted or rejected. An associated explanation of the interactions between engineering sub-systems and how they drive the performance of the overall system is also lacking. The work in this paper aims to provide insight into these elements of pipeline design.

1.1.1. Explainable AI

Explainable AI (XAI) tools are being adopted into industrial fault diagnosis systems as a way of improving the transparency of ML outputs for maintenance applications [17]. XAI tools can be integrated into a ML model pipeline at several stages and used to provide different levels of explanations. The most flexible approach is applying “post-hoc” techniques where the decision-making process for a trained model is explained after training and producing predictions. This includes techniques such as surrogate modeling (such as in [18], where an inherently transparent model is used to approximate the behavior of the model of interest) or perturbation-based methods (such as Shapley Additive exPlanations (SHAP) [19]). SHAP is based on Shapley values [20], a coalition game theory method that computes the contribution of model input features to the obtained output to provide insight into the importance of each feature. This importance can be provided on a local or global scale, where global explanations provide an overview of the models learned relationship between all input variable instances and resulting output predictions, and local explanations provide insight into the contributions of one observation or region in the model space [21]. Due to its flexibility in explanation levels and robust theoretical grounding, SHAP is becoming an increasingly popular and widely adopted XAI approach [22]. SHAP is used in [23], where a human-readable XAI framework utilizes SHAP to explain the outputs of 3 regression models predicting: power output in a combined cycle gas plant; gearbox vibration; and bearing wear in feed-water pumps. The human-readable outputs are generated through encoding text explanations of SHAP outputs to aid non-ML experts in engaging with their predictive tools in a more intuitive manner. In [17] SHAP is used and then compared with Local Depth-based Feature Importance for Isolation Forest (Local-DIFFI) to explain the feature importance of machine learning-based fault diagnosis of rotating machinery. More generally, Shukla et al. [24] investigates the application of XAI techniques when using deep neural nets in predictive maintenance within the field of aerospace integrated vehicle health management.

The literature has a general consensus that XAI techniques can contribute towards establishing trust in automated machine learning processes. However, the literature also denotes limitations in state-of-the-art

XAI techniques related to the complexity and human biases inherent in the XAI models themselves, in addition to XAI end-users' ability to understand the explanations provided. However, these limitations are not the focus of this paper, where SHAP, due to its robust theoretical foundation, is used to explain the outputs of a decision tree trained to identify which features (pipeline design options) lead to the success or failure of fault detection and diagnostic systems, as measured through the model classification error.

1.2. Transfer Learning

The lack of sufficient labeled data is a common problem in supervised learning. However, the utilization of knowledge gained from available (source domain) data can often improve model performance in a situation where the new application (target domain) may lack sufficient labeled data [25]. TL can be categorized in many ways, from considering this lack of sufficient labeled data to also considering the similarities between data domains or modeling tasks [26]. TL is usually defined by three high-level categories: inductive, transductive, and unsupervised, with some authors [26] including a fourth to consider negative transfer.

Inductive TL covers scenarios where a degree of labeling is available in both domains or where tasks and data modalities are the same. Transductive TL is where labeled data is only available from the source domain or where the tasks are different but relevant to each other. Unsupervised TL is where there is no labeled data from either domains or the domains and tasks are different. Lastly, negative transfer considers when the transfer of knowledge hinders the performance of the model in the target domain. TL can be further broken down into the types of learning conducted and whether the transfer of knowledge focuses solely on data [25], such as learning on instances. Learning on instances applies where there is access to a small sample of labeled data in the target domain, with relevant data able to be adapted from the source domain [26], that is, a sub-category of inductive TL. Additionally, further sub-categories for TL can be based upon data heterogeneity [27]. Finally, in [28], a different approach to categorization is presented where four industrial transfer learning concepts are proposed, which focus on the description of the intended application instead of the technique used to conduct the transfer.

As this paper focuses on the industrial engineering application of transfer learning, the concept of "cross-entity" transfer learning (from [28]) will be adopted. Cross-entity transfer specifically focuses on the scenario where knowledge is transferred between two similar assets with similar functionality and faults. Inductive TL relates to this concept as labeled data is available for both source and target assets/domain. However, the labeled data in question varies in terms of different data quantities and data diversity.

Other related works, which combine TL and uncertainty quantification, mainly focus on the uncertainty of available data or model predictions. In [29], the authors use the variance in the outputs of an ensemble of pre-trained, reweighted convolutional neural networks to calculate epistemic uncertainty in the diagnosis of COVID-19 from medical images. In [30], the authors use uncertainty quantification to identify the most uncertain samples in available industrial elevator usage data while transferring their hybrid digital twin and Neural Network architecture to new elevator usage scenarios. However, in [31], the meta data for simulations used to characterize the design of a previous crash box is used to predict the behavior of a new, similar, early-stage design with limited simulations. The TL in [31] was designed to permit further uncertainty quantification in new designs but does not perform this analysis or explore how potential designs may be compared.

2. Methodology

To improve the trustworthiness of a data-driven ML system, it is desirable to reduce uncertainty during the data pipeline design phase, as design changes are easiest at this time. During the initial stages of the pipeline design, a developer will be presented with many design choices, each incurring a different performance trade-off. Firstly, inter-stage pipeline uncertainties may compound and propagate to mislead the analytic in unforeseen ways, preventing a developer from initially identifying underperforming

choices. Additionally, pipeline design choices may be constrained by the desired system functionality (consider the purpose of a fault diagnostic rather than a fault detection system), and some sources of data may degrade the performance (e.g. due to poor sensor positioning or calibration). Rapid design or automation of design for analytic pipelines can assist in presenting a developer with explanations for the uncertainty sources in possible designs to help them identify important drivers of system performance. This can identify areas to focus investment to reduce the overall uncertainty, and so risk, in the system. A flowchart of the full approach proposed in this work to support pipeline design is shown in [Figure 2](#)

2.1. Identifying key pipeline stages and design options

As identified in [32], the discretization of a pipeline into stages may be dictated by the project, budget, or domain expertise of the contributing engineers. Considering the industrial data acquisition pipeline in [Figure 1](#) as the basis for a fault detection or diagnosis system, there are several key stages along with the consequences of the resulting design decisions and trade-offs. These are summarized in [Table 1](#) for the case study considered in this work.

Options requiring consideration are:

2.1.1. Sensors and process variable measurements

This stage is primarily concerned with the fidelity of instrumentation/sensing coverage on the asset and how the associated physical phenomena are measured, taking into account the cost of installing and maintaining the measurement system. For example, a rotating plant asset may be monitored by vibration, temperature, current, oil, or acoustic sensors [33], all of which provide different levels of insight into faults of interest.

2.1.2. Data collection

This stage covers the trade-off between the running and upfront cost of the sensors against the resolution and sampling rate of the data acquired [10]. A high precision sensor or high sampling rate may allow fault behavior to be observed in great detail, potentially allowing early fault detection, but will result in the collection of a large amount of non-fault data, which must also be handled/stored.

2.1.3. Data transmission

This stage covers the amount of data that can be reliably transmitted by the chosen data transmission system against the cost of increased bandwidth [11]. Even if the measurement system is capable of generating high-frequency, high-fidelity measurements, the data transmission system may not be able to transmit this with sufficient speed or quality to a centralized storage location.

2.1.4. Data processing and storage

This stage is mainly concerned with the trade-off between the computational cost of processing and storing data against the quantity and type of useful information preserved within the data [11]. This can include the format the data will be presented to the analytic stage, such as a time series signal being translated to time, time-frequency, or frequency domain [34]. Also, the tradeoff between the amount of preserved information and the dimensionality of the data during dimensionality reduction procedures should be considered. Furthermore, some data resolution or meta data concerning the data that was collected may be lost during conversion to a designated storage format, impacting its usefulness and trustworthiness. In [35], the high sampling rate time series data used to capture electrical faults actually carried its predictive value in the proportion of energy in frequency domain subsets at relatively low resolution. In this instance, this stage of the pipeline would have featured sharply characterized uncertainty across the choices of preprocessing.

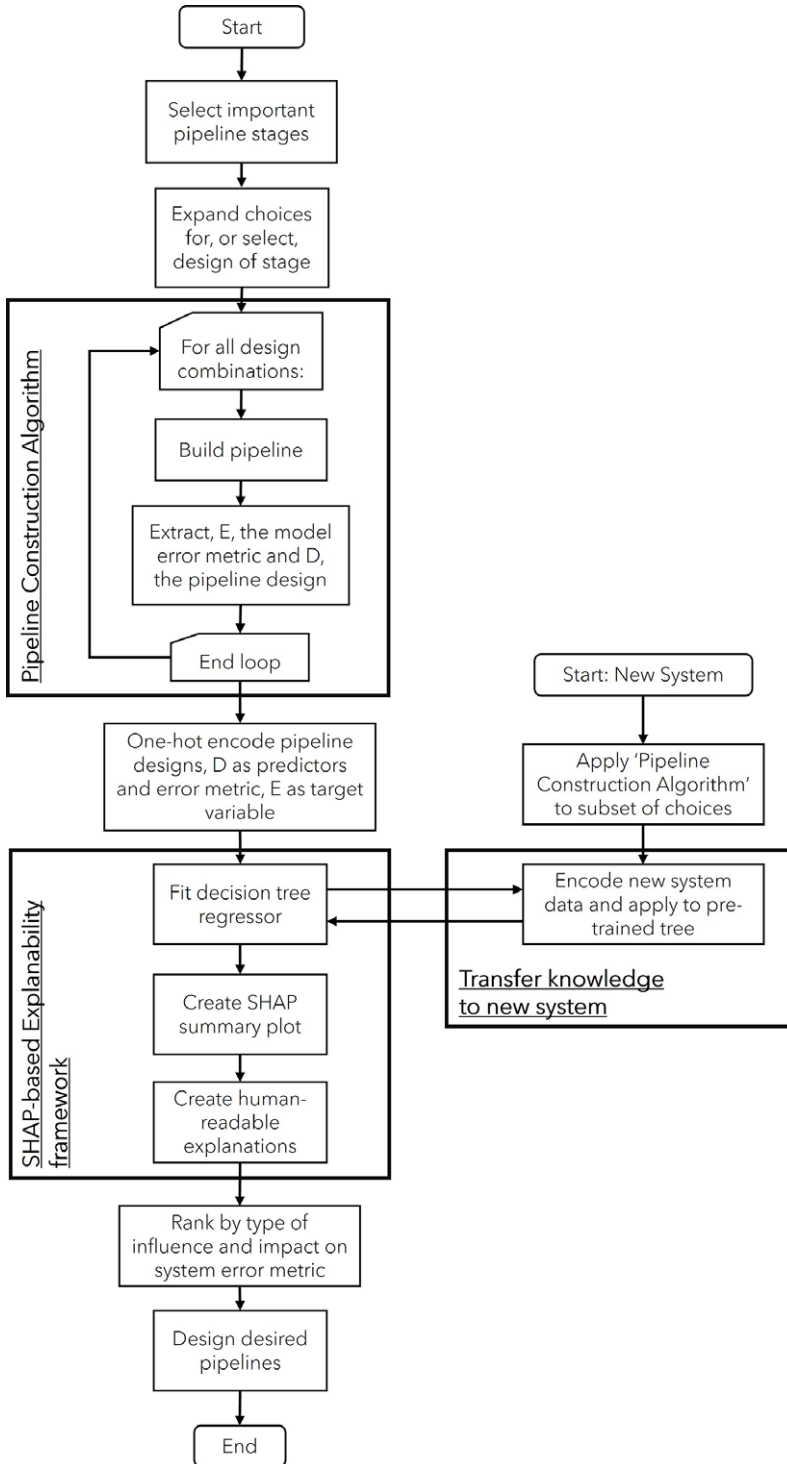


Figure 2. Flowchart of the pipeline design, construction, explanation and transfer to new systems.

Table 1. Summary of pipeline design choices

Pipeline stage	Choices	Choice details
Sensors	6	Fan end only Fan end and base plate Drive end only Drive end and base plate Drive end and fan end All sensors
Window	2	0.5 or 1 second
Data allocation	2	Percentage label prevalence or random
Data domain	3	Timeseries Statistics (mean, median, standard deviation, RMS, peak, skewness, kurtosis, crest, shape and impulse) Frequency (PSD) with PCA* Wavelet scattering with PCA*
Classification task	4	Binary (0, healthy or 1, fault) End (healthy, Fan end or Drive end faults) Location (healthy, ball, inner race or outer race faults) All (healthy, fan end ball, inner race, outer race or drive end ball, inner race, outer race faults)
Classification model**	22	ESD, GNB, CGSVM, CT, CSVM, FKNN, MKNN, MGSVM, KNN, CsKNN, WKNN, CbKNN, EBgT, CKNN, LSVM, QSVM, MT, FT, ESKNN, ERUSBT, EBoT, FGSVM

*Number of principle components chosen to explain at least 95% variance.

**Acronyms provided in Table 2.

2.1.5. Analytics

This stage covers the trade-off between more informative model tasks (such as detailed fault diagnosis) and model family bias against the required model accuracy, model complexity, and computational limitations [36]. Different model families identify relationships within provided data by different means, all of which incur specific biases that may only be suitable for limited applications. The computational requirements for model hyperparameter tuning differ depending on the amount of data available and type of model used. A developer may require the model choice to be transparent to allow the decisions made by the model to be explainable. Additionally, the model task can deteriorate the model performance in different ways. Some faults may be more difficult to identify or separate from others, and the labels used to group data may cause dissimilar samples to be grouped, which can also deteriorate model performance.

2.1.6. Decision metrics

The most informative decision metrics are generally application-specific, with their selection usually driven by the cost of different types of model failure. False alarms can cause healthy equipment to be taken offline to perform unnecessary maintenance, while missed faults can cause unexpected outages of equipment. Both reduce trust in the fault diagnostic system and can eventually result in the model recommendations being ignored altogether. The utilization of different metrics for model selection can lead to different models being favored or even unacceptable models being selected if the validation requirements are not carefully considered [37].

2.1.7. Domain interpretation

The outputs of the ML-based system should be compatible with the linguistic terms and procedures used in the intended application domain to support communication and build trust with the system [38]. A

system that can provide different levels of explanation alongside the ML outputs can give the user more agency in engaging with the system, better informing their resultant decisions.

2.2. Uncertain pipeline construction

Once the relevant pipeline stages and the choices of interest at each of these stages have been identified, pipeline designs observing all combinations of each choice are constructed following Algorithm 1. Algorithm 1 is an exhaustive/greedy search and is needed to provide full observation of the system and the interactions between stages and choices. This permits the prominent and important relationships and dependencies to be uncovered in the model errors. Note that the stages must be built sequentially in order of the flow of data to ensure the pipeline is built in the required order (that is the input sensor data must be chosen before the input data can be pre-processed) and for stages that have a fixed design, the loops in Algorithm 1 will have only one iterative loop. Lastly, the term Error, E , is the validation metric of choice. This provides full observation of the system and the interactions between stages and choices, allowing important relationships and dependencies to be uncovered in the model errors. The design, D , and the model metric, E , now form a new set of data to perform meta-analysis on.

Algorithm 1 Pipeline construction

Variables:

Stage, $S(1,N)$.

Choices, $C((1, C_{S_1}), (1, C_{S_2}), \dots, (1, C_{S_N}))$ where C_s is a vector of $(1, C_{S_s})$, choices for stage, s .

Error, $E(1,Z)$ where $Z = \prod_{i=1}^N \sum_{j=1}^{C_{S_i}} j$.

Design, $D(1,Z)$.

procedure BUILDPIPELINES(S, C).

 % Construct all design combinations.

$z = 1$.

for choice $k, 1 \leftarrow C_{S_1}$ **do**.

for choice $m, 1 \leftarrow C_{S_2}$ **do**

 ...

for choice $p, 1 \leftarrow C_{S_N}$ **do**.

 % Build pipeline of design D_z .

 % Get model error, E_z .

$D_z = (S_1(k), S_2(m), \dots, S_N(p))$

$E_z = f(S_1(k), S_2(m), \dots, S_N(p))$

$z += 1$.

end for

 ...

end for.

end for.

return D, E .

end procedure.

2.3. Explanations of pipeline design uncertainty

To enhance the domain interpretation of the system, a SHAP-based human-readable explanation framework was constructed, as in [35]. This was created to explain how the design choices in the pipeline relate to the uncertainty in the form of the classification error produced by the model in that pipeline. The design of the pipeline is “one hot encoded,” where categorical variables with n classes are converted into n separate binary variables, with a 0 representing an absence of the choice in the design and 1 representing

the inclusion of the choice in the design. A decision tree regressor is fit over all available data [39] with the encoded pipeline design as predictors and the model error as the target. The decision tree and SHAP framework are built in Python using the scikit-learn¹ and SHAP² libraries.

To aid the understanding of SHAP plots produced by the SHAP-based framework, human-readable explanations are generated to explain the significance and type of influence each choice has on the detection or diagnostic system performance. Explanations provided by the framework elaborate on the choice's individual impact on the system error using domain-relevant language. In cases where the SHAP tools have identified the choice as a 100% positive or negative influence, further information is provided using the information from the raw pipeline design data. For such cases, the framework allows for local explanations, which provide further analysis of anomalous results. From the human-readable explanations, the most impactful positive and negative performance drivers for each pipeline stage can be selected with the aim of producing the “best” and “worst” performing pipelines to validate the SHAP-based framework. To construct a highly performing pipeline, design decisions with a positive influence and highest impact are selected for each pipeline stage.

2.4. Transferring knowledge of design uncertainty to new systems

There is potential for knowledge gained from one system to be transferred to gain insight into the uncertainty drivers in the design of a similar, less observed system. Fully observing one system to an extent that provides enough information to meaningfully compare pipeline designs and the drivers of uncertainty can require significant overhead. Suitable sensor coverage, fault measurements, and computational resources to compute all design combinations must be provided, which may be possible on a test rig but may be difficult to translate into a practical environment.

To transfer instances from the target domain to the source domain, there must be alignment potential between the pipeline classification error between the source and target domains and the alignment potential between the pipeline design choices, ensuring the design of the pipelines should have the same or fewer design choices. This can be determined through various metrics or visualizations, as demonstrated in the Case Studies later in this work. Instance transfer is conducted by converting the design of the target domain pipeline to align with that of the source domain. Having the same or fewer choices allows the instances from the target domain to align with the source domain, as design options that are not observed in the source domain cannot be leveraged to the target domain. This instance transfer will allow the pre-trained tree model in the SHAP-based framework from the source domain to be applied to any of the target domain pipelines with consistent encoding of the model inputs. As the model has learned from the source domain, this knowledge can provide insight into the behavior of the target system pipelines.

3. Case Study 1: Rotating plant bearing fault diagnosis in an unexplained data pipeline

Bearings are prolific in industrial applications and are a common point of failure in rotating plants [33]. Accordingly, two well-understood open-source bearing fault test rig data sets will form the basis of a transfer learning task, with this section covering the creation of the source domain pipelines and their diagnostic process choices. For the source domain, the publicly available Case Western Reserve (CW) rotating plant dataset [40] was chosen, which includes a selection of seeded bearing faults introduced to key locations in a motor. Faults are present in the ball, inner race, and three locations in the outer race at both ends of the motor. The faults are monitored by vibration sensors present at the motor fan end, drive end, and base plate. The motor operating conditions are varied between 0–3 HP, with data collected at 12 or 48 kHz (down-sampled to 12 kHz for consistency). This dataset has been extensively utilized to demonstrate machine learning models for engineering applications, and in this instance acts as a

¹ scikit-learn Package: <https://scikit-learn.org/stable/>

² SHAP Package: <https://shap.readthedocs.io/en/latest/>

demonstrative baseline to showcase the framework rather than the classification capability of a particular machine learning model.

3.1. Source domain pipeline design stages and decisions

Six key pipeline stages and their potential design decisions are identified with the choices at these stages given in Table 2. The chosen stages reflect key areas in the pipeline design for this dataset where a diverse range of decisions are available and the optimal solution is not immediately clear. The model choices are chosen similar to [36] due to the range of classification boundary attribution methods and built using MATLAB's classification libraries³, while the window length of the vibration data and diagnostic task is decided based on the number of samples and included fault types. The row ordering in Table 2 reflects the ordering of pipeline stages that was applied in Algorithm 1.

3.2. Uncertain pipeline construction and explanation

Classification testing error was collected for all combinations of variable settings for pipeline design variables, with 5-fold cross-validation for each combination resulting in 31,680 total pipelines. Varying the chosen design variable settings resulted in a wide range of performance output, as shown by the testing error distribution in Figure 3. Two dominant modes are present in the error distribution at ~0% and ~40% testing error, with a heavy tail towards the upper end of the error range, representing setting combinations that contribute to improved or degraded performance, respectively. The fitted decision tree regressor model used to identify pipeline-to-error relations in the SHAP-based framework resulted in a R2 score of 0.9788, with an ideal value of 1, showing that the model fit the data well but still incurs some error. To account for this, the model is retrained 5 times with each trained model being passed through the human-readable XAI framework. Applying the SHAP tools within the human-readable XAI framework generated magnitude plots, which describe the ranking of design choices based on the magnitude of their impact on the bearing fault system error (taken from the trained decision tree output), and summary plots, which show the direction of this impact (that is do the design choices drive the system error up or down). Example figures (Figure C2 and C3) are included within the supplementary materials. The SHAP tools and human-readable explanation tools within the human-readable XAI framework are used to order the design options based on the strength of their impact on the performance outcome (which could be positive or negative). Each design choice is ranked 1 to 39 (the total number of choices) for each retrain of the tree, which is then averaged to give an overall score. The average ranking of all pipeline choices across all stages for 5 decision tree model runs is shown in Table 2.

The high importance and positive influence of "TB" (Task: Binary) and "TE" (Task: End) choices mean models are more likely to perform well on the simpler binary (TB) fault detection case and can differentiate if the faults are present on the drive or fan end of the motor (TE), suggesting the data is quite easily separable using these groupings. Generally, the models seem to struggle with the "All" (TA) choice, the task containing the highest diagnostic information and complexity, suggesting that separating the data based on both the motor cross section and the end of the motor is difficult for the models to separate accurately. The K-Nearest Neighbour (KNN) models are the most important, positively performing model group, which attributes classification boundaries by clustering, suggesting this method is the most appropriate for this case study. The least successful models (ESD, GNB, CGSVM, CT, CSVM) have diverse boundary attribution methods but tend to be the "coarse" equivalent, which restricts the amount of detail that can be captured by the model. This suggests more complex models are required to adequately capture the fault characteristics. The "Wavelet" (time-frequency domain) method was the most impactful and positive influencing choice for the data domain stage of the pipeline. This supports standard engineering understanding that providing models with time and frequency information of a developing fault leads to the strongest identification of the type of

³ MATLAB Classification Models: <https://uk.mathworks.com/help/stats/classification.html>

Table 2. Summary of pipeline stages, choices and their rankings for source and target datasets (averaged over 5 runs)

Pipeline stage (num. choices)	Pipeline choice	Rank (source dataset)	Rank (target dataset)
Sensor (4 (2*))	Fan end only sensor	6.0+	(N-A)
	Drive end only sensor*	13.0+	8.8+
	All sensors	15.4–	(N-A)
	Drive end and base plate sensors	24.6–	(N-A)
	Fan end and base plate sensors	29.4+	(N-A)
	Drive end and fan end sensor	34.4–	(N-A)
Window (2)	0.5 s window	26.8+	20.6+
	1 s window	30.4–	22.0–
Model Task (6 (1*))	Task: Binary (TB) (0, healthy or 1, fault)*	1.0+	1.0+
	Task: End (TE) (healthy, Fan end or Drive end faults)	3.0+	(N-A)
	Task: Location (TL) (healthy, ball, inner race or outer race faults)*	31.0+	20.4+
	Task: All (TA) (healthy, fan end ball, inner race, outer race or drive end ball, inner race, outer race faults)	34.6–	(N-A)
Data Domain (3)	Wavelet scattering with principle component analysis (PCA)**	2.0+	2.0+
	Frequency (Power Spectral Density) with principle component analysis (PCA)**	7.6–	6.6–
	Timeseries Statistics (mean, median, standard deviation, root mean squared (RMS), peak, skewness, kurtosis, crest, shape and impulse)	20.8+	23.4+
Data Alloc. (2)	Random Allocation	38.4–	38.4–
	Prevalence Allocation	38.6+	38.6+
Model, (22)	ESD (Ensemble subspace discriminant model)	4.0–	4.8–
	GNB (Gaussian Naive Bayes model)	5.0–	6.0–
	CGSVM (Coarse Gaussian support vector machine model)	7.8–	10.2–
	CT (Coarse Tree) model	9.0–	19.0–
	CSVM (Coarse support vector machine) model	10.2–	17.2–
	FKNN (Fine K-Nearest Neighbours) model	11.2+	7.2+
	MKNN (Medium K-Nearest Neighbours) model	12.4+	9.0+
	MGSVM (Medium Gaussian support vector machine) model	15.4–	22.6–
	KNB (Kernel Naive Bayes) model	15.6–	26.6–
	CsKNN (Cosine K-Nearest Neighbours) model	16.6+	12.0+
	WKNN (Weighted K-Nearest Neighbours) model	18.0+	13.2+
	CbKNN (Cubic K-Nearest Neighbours) model	19.6+	15.2+
	EBgT (Ensemble Bagged Tree) model	20.6+	15.8+
CKNN (Coarse K-Nearest Neighbours) model	22.6–	23.4–	
LSVM (Linear support vector machine) model	23.4–	35.0–	
QSVM (Quadratic support vector machine) model	26.2+	34.0+	

Continued

Table 2. Continued

Pipeline stage (num. choices)	Pipeline choice	Rank (source dataset)	Rank (target dataset)
	MT (Medium tree) model	27.0–	33.4–
	FT (fine tree) model	27.6+	21.2+
	ESKNN (Ensemble subspace K-Nearest Neighbours) model	28.8+	15.8+
	ERUSBT (Ensemble Random undersampling Boosted Tree) model	31.6–	23.0+
	EBoT (Ensemble Boosted Tree) model	34.2–	26.2+
	FGSVM (fine Gaussian support vector machine) model	36.2–	30.6–

*Subset of choices used for the target dataset, for all other stages the choices are the same.

**Number of principle components chosen to explain at least 95% variance. Choices are marked to indicate if they have an overall (+) positive or (–) negative influence on the system performance. (N-A) marks choices unavailable to the target dataset.

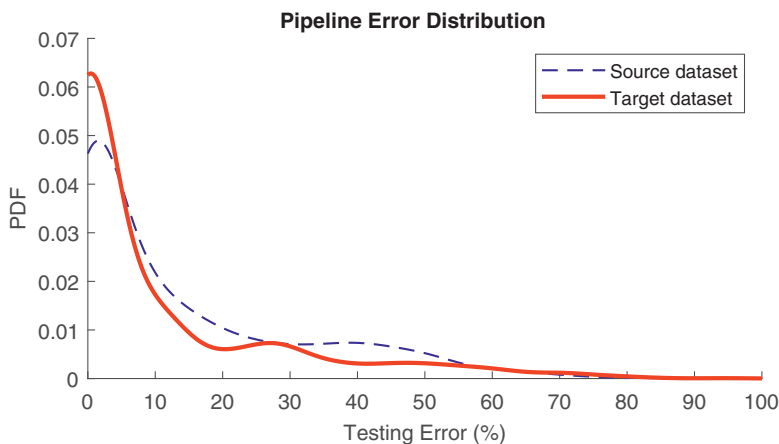


Figure 3. Probability distribution of classification errors across the source and target data sets. The source data set consists of 31,680 CW pipelines. Many pipelines result in low, near 0% errors and the distribution has a heavy tail at higher errors with a peak around 40%. The target data set consists of 2640 generated pipelines. Many pipelines result in low errors, with a heavy tail towards high errors. There is another peak near 30% error, similar to the source domain.

developing fault, instead of using only time or frequency domain representations [34]. For the sensors, the combinations containing the drive end sensor or baseplate sensor in tandem with other sensors tend to perform negatively, and the most important sensor combinations (fan end only with positive performance, drive end only with positive performance, all sensors with negative performance) also have the most extreme positive or negative influences, which drives up their impact on the system performance. The fan end sensor may experience less noise due to being slightly removed from the driving force, which may explain its generally positive influence on the system performance. Lastly, the least significant contributing features towards pipeline error were the design choices for the training and testing set data allocation method stage (Random or Prevalence allocation). Due to the large supply of fault examples in this curated data set, both methods may ensure a variety of fault types would be visible to the models to allow for more consistent training.

Table 3. Summary of “best” and “worst” pipeline choices per stage for source and target datasets

Pipeline stage	CW dataset		Target dataset (MFPT)	
	Best choices	Worst choices	Best choices	Worst choices
Sensor	Fan end sensor only	All sensors	Drive end only*	Drive end only*
Window	0.5 seconds	1 second	0.5 seconds	1 second
Data		Allocation**	Both (label prevalence and Random)	Both (label prevalence and Random)
	Both (label prevalence and Random)	Both (label prevalence and Random)		
Data Domain	Wavelet (time-frequency)	Frequency	Wavelet (time-frequency)	Frequency
Classification task	Binary (fault detection)	All-specific motor cross section location and motor end	Binary (fault detection)	Motor cross section location
Classification Models	Fine, Medium, Cosine, Cubic and Weighted KNN models	Ensemble Subspace Discriminant, Gaussian Naive Bayes, Coarse Gaussian SVM, Coarse Tree and Coarse SVM models	Fine, Medium, Cosine, Cubic and Weighted KNN models	Ensemble Subspace Discriminant, Gaussian Naive Bayes, Coarse Gaussian SVM, Coarse Tree and Coarse SVM models
Error	0.057% (max), 0% (min)	82.4% (max), 50.8% (min)	0% (max), 0% (min)	70% (max), 0% (min)

*There is only one sensor option, so must be selected for both best and worst pipelines.

**The data allocation stage was equally unimportant with a neutral influence on performance between both datasets, so both options are included for selecting best and worst pipelines.

3.3. Selecting improved pipeline designs

Designing the best pipeline based upon the human-readable explanations from the SHAP plots identified the top choices as shown in Table 3. The original pipeline data was filtered using the chosen design choices, and the resulting error histogram for the top 50 pipeline designs is shown in Figure 4 with the maximum error of 0.057%.

4. Case Study 2: Transferring learned design uncertainty to operational systems

A second system may be under-observed, as the time and cost overhead required to augment the asset to collect the same diversity of measurements and recertify the asset may be infeasible. Instead, with the construction of a source system completed, the knowledge of the present uncertainty drivers can be transferred to the under-observed target system. The open-source bearing test rig data set for the target system is the Society For Machinery Failure Prevention Technology (MFPT) dataset [41].

The target dataset contains bearing faults introduced into the ball, inner race, or outer race of a motor-driven bearing housing, which is monitored by one vibration sensor. The operating loading conditions are varied for each bearing health state: healthy data is collected at 270 lbs (sampled at 97.656 kHz), inner race faults are collected for 0–300 lbs (sampled at 48.828 kHz), and outer race faults are collected for 25–300

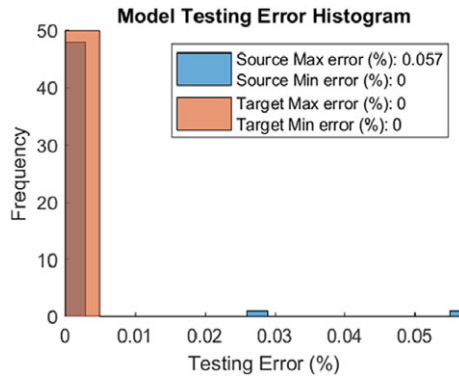


Figure 4. Histogram of classification errors from SHAP recommended “best” pipeline design choices for the source and target domains. All chosen source domain and target domain pipelines have very low classification error of maximum 0.057% and 0% error, respectively.

lbs (sampled at 48.828 kHz) and 270 lbs (sampled at 97.656 kHz). The input shaft is driven at 1500 rpm, with each experiment lasting 3 or 6 seconds. Due to being a smaller dataset, the 48 kHz cases were upsampled for consistency.

While including similar bearing faults, the target dataset includes less sensor coverage and fault locations than the source system, which limits the amount of pipelines that can be designed. This specifically affects the sensor and classification task options, as there is no access to fan end and base plate sensors or fan end faults. The summary of pipeline stages and choices for the target dataset is shown in Table 2.

4.1. Target data pipeline design

To validate the methodology, the pipelines for the combination of all available choices were constructed with the classification error collected for each pipeline with 5-fold cross-validation, as in Section 3, resulting in 2640 total pipelines for the target case. A subset of this will be provided to the human-readable XAI framework trained in Section 3. The histogram of errors over all pipelines is shown in Figure 3, with a peak at ~0% and ~30% and a long tail towards high classification errors. The histogram demonstrates the target domain behaves similarly to the source domain. A Quantile-Quantile plot in Figure 5 is used as a comparison between the source and target error histograms over all pipeline designs and captures the errors at the design extremity. The Pearson correlation coefficient between the two distributions is 0.97, showing a strong linear correlation between the quantiles of the two error distributions. The similar behavior (identical distributions follow the plotted theoretical line) suggests that the knowledge in the source pipelines is a good candidate to support the explanations of the target case.

The data is not encoded to align with the source data, and a subset of 110 pipelines is provided to the human-readable XAI framework (that has been trained on pipeline data from Section 3) to explain the impact of each choice. The subset chosen were those with a window length of 1 s, the classification task of cross-section location (TL), the data allocation method of random, and the data domain of frequency with all models and cross-validation samples taken.

The average ranking of each choice from the human-readable XAI framework is shown in Table 2, allowing for a comparison with the source case.

Many of the recommendations align with the source system, such as the allocation methods being equally unimportant due to the balanced ratio of faulty to healthy data; the high performance on the binary classification task; and processing data in the wavelet (time-frequency) domain providing more useful information to the models. The target case has several choice limitations compared to the source system, such as the limited diagnostic tasks and sensor combinations; however, the influence of the available

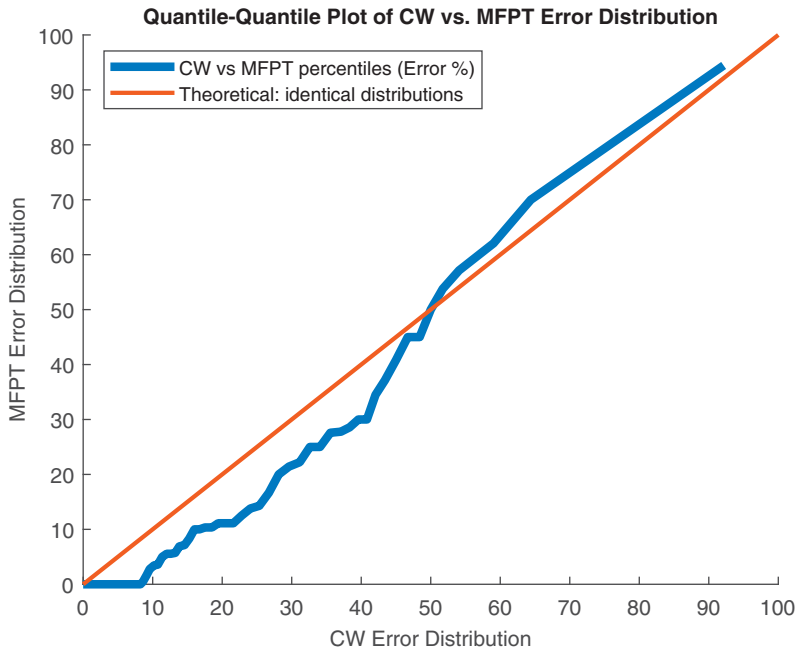


Figure 5. *Quantile-Quantile Plot of the source and target pipeline error distributions. The Pearson correlation coefficient between the two distributions is 0.97 showing a strong linear correlation between the quantiles of the two error distributions. As the two systems align well with the theoretical plot, information gained from one system would provide useful insight into the behaviour of the other. This is effectively transference of the expected errors between the source and target systems.*

choices is in agreement with the source case (both contributing positively to system performance). Due to the limited samples provided to the human-readable XAI framework, the influence of some model types has changed, such as the Ensemble Boosted Tree and Ensemble RUS Boosted Tree, which perform well for the limited pipeline samples provided for the target case. Lastly, the window lengths have increased in importance but result in the same influence. The target dataset has less data samples than the source domain, and the use of the 0.5 second window length can increase the number of observations used to train the models.

4.2. Selecting improved pipeline choices for the target system

Using the human-readable explanations to rank the choices in each stage for the target case, the “best” pipelines can be extracted from the all target pipeline data, including the held-out design data. For the “best” pipelines, the choices presented in Table 3 are chosen. Aside from the limited sensor options and classification task, the rest of the recommendations for the “best” pipeline choices align with the recommendations for the source system. The histogram of errors for the “best” 50 pipeline designs is shown in Figure 4, with the maximum error of 0%. Applying the “worst” choices highlighted by the source domain does lead to degraded performance in the target domain data. Figures demonstrating this (Figure C1) are available in the supplementary materials.

5. Conclusion

Highly regulated industries require a complete understanding of the level of trust they can have in their data acquisition and information systems. Trustworthiness can be eroded along the entire length of a data pipeline, from sensor placement to the analytics performing the fault diagnostics. The empirical study

presented in this paper has contributed a means of explaining how uncertainties in each design stage of a pipeline influence the overall error and how this understanding can be transferred to new target systems that may lack abundant labeled data. Once a source domain is created, the explanation framework can then be used to prescribe high-performing pipeline designs in other, similar monitoring situations without repeating the exhaustive evaluation of design choices. The SHAP-based framework explanations were able to adequately identify choices that lead to the construction of better bearing fault classification pipeline performance, even across difficult-to-compare data processing stages, providing an operator with insight into the uncertainty drivers in their data acquisition systems during the design stage.

The approach presented is flexible; it can allow developers to consider where design decisions are made in their proposed pipelines and to investigate the impact their choices can have on a data acquisition pipeline performance. In this presentation of the work, no constraints were placed on the design of the pipeline; however, if a developer must work to certain specifications, this methodology could allow them to design around these constraints in order to improve system performance and understand the resultant limitations. Additionally, the proposed methodology can be expanded to handle more stages (or settings per stage) to suit the application, and the human-readable explanations can be customized to present end users with more familiar language to further enhance understanding of the outputs. Lastly, there is the potential to transfer learned uncertainty drivers with human-readable explainability frameworks trained on fully observed systems to gain insight into new, comparable cases, which may allow experimental test rigs to translate to practical assets. This was demonstrated successfully by decoupling the positive performance drivers in a target system using a smaller subset of pipeline designs.

There are several notable limitations to the work, which would be a high priority for future work. Firstly, investigating if sufficient pipeline performance alignment can be detected from the source and target input datasets would provide an early warning system for the potential of negative transfer, where the insights from the source domain may not apply to the target domain before the pipelines are constructed. Additional analysis on the sensitivity of the method to the alignment between the pipeline design performance would further support the detection of negative transfer, which could be conducted through sourcing a third, less applicable dataset or investigating the impact of noise on the currently applied datasets. The SHAP framework applied in this work depends on high accuracy from the decision tree used to explain the impact of the pipeline designs. More investigation into the failure modes of this model type, with specific attention paid to local-level explanations where pipeline design performance may deviate from identified patterns, would provide more insight and control to those responsible for pipeline design. Currently, to demonstrate the explanations of the pipelines apply to all designs explored within this work, an exhaustive search was used to ensure all design choices were compared. This can be computationally expensive, and depending on the impact of the insights, it may not provide additional benefits if the design choices have little impact. In the future, more intelligent algorithms can be explored with a more cyclical approach, whereby the methodology discussed in this work could provide updates when new pipeline design results are available to test the impact of the new choices before they are exhaustively compared. A more intelligent search algorithm could highlight the most informative design choices to create pipelines for, making more effective use of the search space.

The application of this approach could be useful in cases requiring rapid design or automation of design for analytic pipelines for similar assets in a fleet, while providing developers with information on where to focus resources to reduce diagnostic system risk. Future work could involve investigating the trade-off between the cost of implementing recommended design choices and savings generated by enhanced performance of maintenance tools over time.

Supplementary material. The supplementary material for this article can be found at <http://doi.org/10.1017/dce.2024.54>.

Data availability statement. All data used in this work is open source, and can be found at <https://engineering.case.edu/bearingdatacenter> [40] for the source domain data and <https://www.mfpt.org/fault-data-sets/> [41] for the target domain data.

Author contributions. J Blair—Investigation; J Blair, O. Amin—Software; J Blair, B Stephen, B Brown, A Forbes—Methodology; B Stephen, B Brown, A Forbes—Supervision; B Stephen, B Brown, A Forbes, S McArthur—Funding acquisition;

J Blair—Writing—original draft; J Blair, O. Amin, B Stephen, B Brown, A Forbes, S McArthur—Writing—reviewing and editing. All authors approved the final submitted draft.

Funding statement. This work is part of an EPSRC funded project, code EP/V519777/1.

Competing interest. None.

MSC codes. Primary—00
Secondary—00A06

Abbreviations.

- Explainable AI (XAI)
- Prognostics and health management (PHM)
- Condition monitoring (CM)
- Machine learning (ML)
- Deep Learning (DL)
- Transfer learning (TL)
- Shapley Additive exPlanations (SHAP)
- Task: Binary (TB)
- Task: End (TE)
- Task: Location (TL)
- Task: All (TA)
- Principle component analysis (PCA)
- Root mean squared (RMS)
- Ensemble subspace discriminant model (ESD)
- Gaussian Naive Bayes model (GNB)
- Coarse Gaussian Support vector machine model (CGSVM)
- Coarse Tree model (CT)
- Coarse support vector machine (CSVM)
- Fine K-Nearest Neighbours model (FKNN)
- Medium K-Nearest Neighbours model (MKNN)
- Medium Gaussian support vector machine (MGSVM)
- Kernel Naive Bayes model (KNB)
- Cosine K-Nearest Neighbours model (CsKNN)
- Weighted K-Nearest Neighbours model (WKNN)
- Cubic K-Nearest Neighbours model (CbKNN)
- Ensemble Bagged Tree model (EBgT)
- Coarse K-Nearest Neighbours model (CKNN)
- Linear support vector machine (LSVM)
- Quadratic support vector machine (QSVM)
- Medium tree model (MT)
- Fine tree model (FT)
- Ensemble Subspace K-Nearest Neighbours model (ESKNN)
- Ensemble Random Undersampling Boosted Tree model (ERUSBT)
- Ensemble Boosted Tree model (EBoT)
- Fine Gaussian support vector machine model (FGSVM)
- Case Western (CW)
- Society For Machinery Failure Prevention Technology (MFPT)

Ethical standards. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

References

- [1] **L. J. Bond et al.**, “Prognostics and life beyond 60 years for nuclear power plants”, in *IEEE Conference on Prognostics and Health Management*, 2011, pp.1–7
- [2] **J. Coble et al.**, “A Review of Prognostics and Health Management Applications in Nuclear Power Plants”, in *International Journal of Prognostics and Health Management*, 2020, Vol. 6
- [3] **M. C. Kennedy, and A. O’Hagan.** “Bayesian calibration of computer models”, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* Vol. 63, Issue 3, Pg 425–464, (2001), <https://doi.org/10.1111/1467-9868.00294>
- [4] **Liu, G., et al.** “Knowledge transfer in fault diagnosis of rotary machines”. *IET Collab. Intell. Manuf.* 4(1), 17–34 (2022). <https://doi.org/10.1049/cim2.12047>

- [5] **BIPM and IEC and IFCC and ILAC and ISO and IUPAC and IUPAP and OIML**, “Type A evaluation of standard uncertainty” in “Evaluation of measurement data - Guide to the expression of uncertainty in measurement”, Joint Committee for Guides in Metrology, JCGM 100:2008, Chpt 4.2, Pg 10–11, <https://doi.org/10.59161/JCGM100-2008E>
- [6] **BIPM and IEC and IFCC and ILAC and ISO and IUPAC and IUPAP and OIML**, “Type B evaluation of standard uncertainty” in “Evaluation of measurement data - Guide to the expression of uncertainty in measurement”, Joint Committee for Guides in Metrology, JCGM 100:2008, Chpt 4.3, Pg 11–14, <https://doi.org/10.59161/JCGM100-2008E>
- [7] **A. D. Kiureghian, O. Ditlevsen**, “Aleatory or epistemic? Does it matter?”, *Structural Safety*, Volume 31, Issue 2, 2009, Pages 105–112, ISSN 0167-4730, <https://doi.org/10.1016/j.strusafe.2008.06.020>.
- [8] **E. Hüllermeier, W. Waegeman**, “Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods”. *Mach Learn* 110, 457–506 (2021). <https://doi.org/10.1007/s10994-021-05946-3>
- [9] **J. Hines and D. Garvey**, “Process and equipment monitoring methodologies applied to sensor calibration monitoring”, in *Quality and Reliability Engineering International*, Vol. 23, pp. 123–135, 2007
- [10] **H. Harb and A. Makhoul**, “Energy-Efficient Sensor Data Collection Approach for Industrial Process Monitoring,” in *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 661–672, Feb. 2018, doi: 10.1109/TII.2017.2776082.
- [11] **M. Z. A. Bhuiyan, J. Wu, G. Wang, Z. Chen, J. Chen and T. Wang**, “Quality-Guaranteed Event-Sensitive Data Collection and Monitoring in Vibration Sensor Networks,” in *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 572–583, April 2017, doi: 10.1109/TII.2017.2665463.
- [12] **BIPM and IEC and IFCC and ILAC and ISO and IUPAC and IUPAP and OIML**, “Uncertainty” in “Evaluation of measurement data - Guide to the expression of uncertainty in measurement”, Joint Committee for Guides in Metrology, JCGM 100:2008, <https://doi.org/10.59161/JCGM100-2008E>
- [13] **Y. -H. Lin and G. -H. Li**, “A Bayesian Deep Learning Framework for RUL Prediction Incorporating Uncertainty Quantification and Calibration,” in *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7274–7284, Oct. 2022, doi: 10.1109/TII.2022.3156965.
- [14] **M. Feurer, A. Klein, K. Eggenberger, J. Springenberg, M. Blum, and F. Hutter**. “Efficient and robust automated machine learning”. In *Advances in neural information processing systems*. 2962–2970. (2015)
- [15] **S. Liu, P. Ram, D. Vijaykeerthy, D. Bouneffouf, G. Bramble, H. Samulowitz, D. Wang, A. Conn, and A. Gray**, “An ADMM Based Framework for AutoML Pipeline Configuration” in *Proceedings of the AAAI Conference on Artificial Intelligence* 34(04):4892–4899, (2020), DOI:10.1609/aaai.v34i04.5926
- [16] **C. Yang, J. Fan, Z. Wu, and M. Udell**. “AutoML Pipeline Selection: Efficiently Navigating the Combinatorial Space”. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20)*. New York, NY, USA: Association for Computing Machinery, 1446–1456. (2020) <https://doi.org/10.1145/3394486.3403197>
- [17] **L. Brito, G. A. Susto, J. N. Brito, M. A.V. Duarte**, “An explainable artificial intelligence approach for unsupervised fault detection and diagnosis in rotating machinery”, *Mechanical Systems and Signal Processing*, Volume 163, 2022, 108105, ISSN 0888-3270, <https://doi.org/10.1016/j.ymssp.2021.108105>.
- [18] **H. Tan and H. Kotthaus**, “Surrogate Model-Based Explainability Methods for Point Cloud NNs,” *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2022, pp. 2927–2936, doi: 10.1109/WACV51458.2022.00298.
- [19] **Scott M. Lundberg and Su-In Lee**, “A Unified Approach to Interpreting Model Predictions”, in *Advances in Neural Information Processing Systems*, volume 30, 2017, https://proceedings.neurips.cc/paper_files/paper/2017
- [20] **L. Shapley**, “A Value for n-Person Games”. In: Kuhn, H. and Tucker, A., Eds., *Contributions to the Theory of Games II*, Princeton University Press, Princeton, 307–317. (1953) <https://doi.org/10.1515/9781400881970-018>
- [21] **I. Ahmed, G. Jeon and F. Piccialli**, “From Artificial Intelligence to Explainable Artificial Intelligence in Industry 4.0: A Survey on What, How, and Where,” in *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5031–5042, Aug. 2022, doi: 10.1109/TII.2022.3146552.
- [22] **G. Van Den Broeck, A. Lykov, M. Schleich, D. Suciu**, “On the Tractability of SHAP Explanations”, in *Journal of Artificial Intelligence Research*, 74 (2022) 851–886, <https://doi.org/10.1613/jair.1.13283>.
- [23] **O. Amin, B. Brown, B. Stephen, and S. McArthur**, “A Case-study Led Investigation of Explainable AI (XAI) to Support Deployment of Prognostics in the industry”, in *PHM Society European Conference*, 7(1), 9–20.(2022). <https://doi.org/10.36001/phme.2022.v7i1.3336>
- [24] **B. Shukla, I.-S. Fan, and I. Jennions**, “Opportunities for explainable artificial intelligence in aerospace predictive maintenance,” *PHM Society European Conference*, vol. 5, no. 1, p. 11, 2020
- [25] **F. Zhuang et al.**, “A Comprehensive Survey on Transfer Learning”, in *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.
- [26] **S. Niu, Y. Liu, J. Wang and H. Song**, “A Decade Survey of Transfer Learning (2010–2020)”, in *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, Oct. 2020, doi: 10.1109/TAI.2021.3054609.
- [27] **O. Day and T. M. Khoshgoftaar**, “A survey on heterogeneous transfer learning,” *J. Big Data*, vol. 4, no. 1, Dec. 2017.
- [28] **B. Maschler and M. Weyrich**, “Deep Transfer Learning for Industrial Automation: A Review and Discussion of New Techniques for Data-Driven Machine Learning”, in *IEEE Industrial Electronics Magazine*, vol. 15, no. 2, pp. 65–75, June 2021, doi: 10.1109/MIE.2020.3034884.

- [29] **A. Shamsi, H. Asgharnezhad, S. Jokandan, A. Khosravi, P. Kebria, D. Nahavandi, S. Nahavandi and D. Srinivasan**, “An Uncertainty-Aware Transfer Learning-Based Framework for COVID-19 Diagnosis”, in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1408–1417, April 2021, doi: [10.1109/TNNLS.2021.3054306](https://doi.org/10.1109/TNNLS.2021.3054306).
- [30] **Q. Xu, S. Ali, T. Yue, and M. Arratibel**. “Uncertainty-Aware Transfer Learning to Evolve Digital Twins for Industrial Elevators”. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '22)*, Singapore. ACM, New York, NY, USA, November 14–18, 2022, 12 pages. <https://doi.org/10.1145/3540250.3558957>
- [31] **G. Colella, V. Lange and F. Duddeck**, “Transfer learning for metamodel construction to enable uncertainty quantifications in crash design based on scarce data availability”, ISMA-USD 2022, September 2022, <https://doi.org/10.5281/zenodo.7564630>
- [32] **Mohammad Saeid Mahdavejrad, Mohammadreza Rezvan, Mohammadamin Barekatin, Peyman Adibi, Payam Barnaghi, Amit P. Sheth**, Machine learning for internet of things data analysis: a survey, *Digital Communications and Networks*, Volume 4, Issue 3, 2018, Pages 161–175, <https://doi.org/10.1016/j.dcan.2017.10.002>.
- [33] **N. Jammu, and P. Kankar**, “A review on prognosis of rolling element bearings”. *International Journal of Engineering Science and Technology*, 3., (2011)
- [34] **Swapnil K. Gundewar and Prasad V. Kane**, “Bearing fault diagnosis using time segmented Fourier synchrosqueezed transform images and convolution neural network”, *Measurement*, Volume 203, 2022, 111855, <https://doi.org/10.1016/j.measurement.2022.111855>.
- [35] **P. C. Baker, B. Stephen, M. D. Judd**, “Compositional Modeling of Partial Discharge Pulse Spectral Characteristics,” in *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 7, pp. 1909–1916, July 2013, doi: [10.1109/TIM.2013.2247711](https://doi.org/10.1109/TIM.2013.2247711).
- [36] **M. Z. Ali, M. N. S. K. Shabbir, X. Liang, Y. Zhang and T. Hu**, “Machine Learning-Based Fault Diagnosis for Single- and Multi-Faults in Induction Motors Using Measured Stator Currents and Vibration Signals”, in *IEEE Transactions on Industry Applications*, vol. 55, no. 3, pp. 2378–2391, May–June 2019, doi: [10.1109/TIA.2019.2895797](https://doi.org/10.1109/TIA.2019.2895797).
- [37] **J. Blair, B. Stephen, B. Brown, A. Forbes, and S. McArthur**, “Hybrid Fault Prognostics for Nuclear Applications: Addressing Rotating Plant Model Uncertainty”. *PHM Society European Conference*, 7(1), 58–67. (2022) <https://doi.org/10.36001/phme.2022.v7i1.3321>
- [38] **S. Mohseni, N. Zarei, and E. D. Ragan**, “A survey of evaluation methods and measures for interpretable machine learning”. (2018). ArXiv, abs/1811.11839.
- [39] **A. Medina-Borja**, “*Uncovering Complex Relationships in System Dynamics Modeling: Exploring the Use of CART, CHAID and SEM.*” (2007).
- [40] **Seeded Fault Test Data**, Case Western Reserve University Bearing Data Center, Accessed: Jul. 24, 2024 [Online]. Available: <https://engineering.case.edu/bearingdatacenter>
- [41] **Bearing Fault Dataset**, Condition Based Maintenance Fault Database for Testing of Diagnostic and Prognostics Algorithms, MFPT, Accessed: May 13, 2021 [Online]. Available: <https://www.mfpt.org/fault-data-sets/>