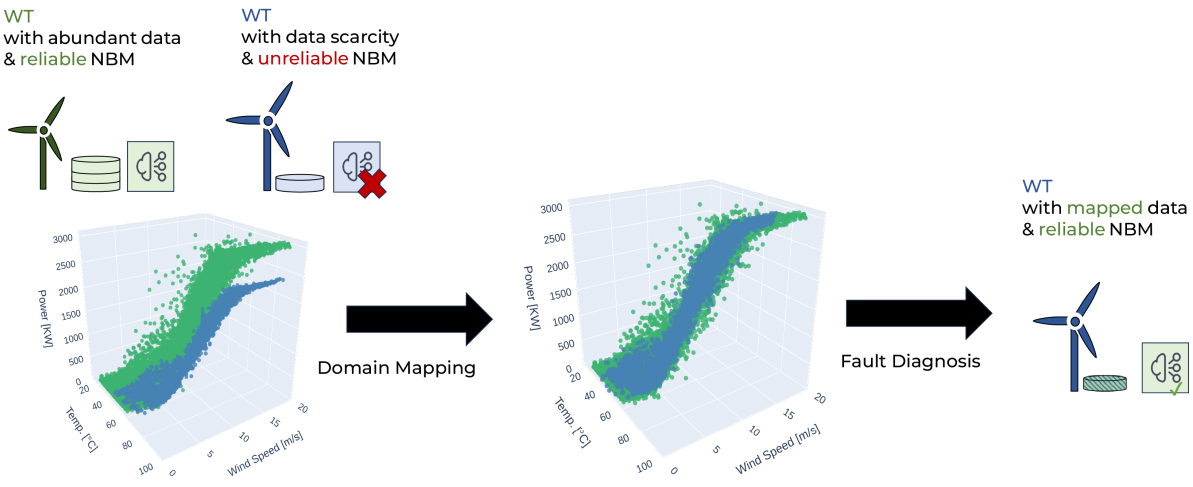
Fault Diagnosis in New Wind Turbines using Knowledge from Existing Turbines by Generative Domain Adaptation

Stefan Jonas[*1, 2], **Angela Meyer**[1,3]
[1]School of Engineering and Computer Science, Bern University of Applied Sciences, Biel, Switzerland
[2]Faculty of Informatics, Università della Svizzera italiana, Lugano, Switzerland
[3]Department of Geoscience and Remote Sensing, Delft University of Technology, Delft, The Netherlands

Abstract

Intelligent condition monitoring of wind turbines is essential for reducing downtimes. Machine learning models trained on wind turbine operation data are commonly used to detect anomalies and, eventually, operation faults. However, data-driven normal behavior models (NBMs) require a substantial amount of training data, as NBMs trained with scarce data may result in unreliable fault diagnosis. To overcome this limitation, we present a novel generative deep learning approach to make SCADA samples from one wind turbine lacking training data resemble SCADA data from wind turbines with representative training data. Through CycleGAN-based domain mapping, our method enables the application of an NBM trained on an existing wind turbine to one with severely limited data. We demonstrate our approach on field data mapping SCADA samples across 7 substantially different WTs. Our findings show significantly improved fault diagnosis in wind turbines with scarce data. Our method achieves the most similar anomaly scores to an NBM trained with abundant data, outperforming NBMs trained on scarce training data with improvements of +10.3% in F1-score when 1 month of training data is available and +16.8% when 2 weeks are available. The domain mapping approach outperforms conventional fine-tuning at all considered degrees of data scarcity, ranging from 1 to 8 weeks of training data. The proposed technique enables earlier and more reliable fault diagnosis in newly installed wind farms, demonstrating a novel and promising research direction to improve anomaly detection when faced with training data scarcity.



*stefan.jonas@bfh.ch

Keywords wind turbine · condition monitoring · fault diagnosis · deep learning · transfer learning · generative domain adaptation · domain mapping

1 Introduction

Wind energy plays an essential role in the global shift towards renewable energy. With the increasing role of wind energy, improving the reliability of wind turbines (WTs) is of crucial importance [1]. The resulting supervisory control and data acquisition (SCADA) data has facilitated the application of deep learning methods for data-driven condition monitoring tasks in WTs [2], [3]. Classification models for WT fault diagnosis represent a major share of these models [4]. However, these supervised classification methods require thoroughly labeled WT maintenance records and a fault database containing SCADA data for a wide variety of fault types, which is often unavailable in practice. Normal behavior models (NBMs) [2], [5] can overcome this limitation. NBMs are based on anomaly detection and model the fault-free operation behavior of WT components. As opposed to classification models, NBMs only require a training dataset comprising historical, fault-free operation measurements. Deviations from the expected value obtained from the NBM at test time, i.e., when the trained model is used in an operational context, can be viewed as an anomaly score, with continuous deviations indicating possible incipient faults.

A major limitation of NBMs is that they require a large training set that is representative of the various WT operation and environmental conditions. Training NBMs with scarce or non-representative data (for instance, only records at low wind speeds) may lead to unreliable fault diagnosis models [6], [7]. Large training sets are sometimes unavailable as scarcity of representative training data can occur, for instance, in newly installed WTs, after maintenance, part replacements, or as a result of aging, rendering previously collected data unrepresentative of the future WT operational states. Data scarcity is further exacerbated by the lack of data sharing in the wind industry, which is inhibiting research and development [8]. A scarcity in training data poses a serious challenge for WT fault diagnosis tasks.

Transfer learning and, in particular, domain adaptation, aims to overcome the limitations caused by data scarcity by transferring knowledge and adapting models from data of a different but related domain [9], [10]. In our fault diagnosis context, domains are typically represented by a WT and its associated data, knowledge, and NBMs. The goal is to use a source domain where data is abundantly available, represented by a WT with abundant data, to improve a task on a target domain, represented by a WT with scarce training data. Domain adaptation has shown success in numerous WT-related fault classification tasks for classifying unlabeled data (e.g., [11]–[13]). However, domain adaptation for unsupervised anomaly detection has hardly been investigated [14]. This is likely a result of multiple challenges faced in applying domain adaptation to unsupervised anomaly detection. First, it usually involves time series data as opposed to the emphasis on image data. Further, by definition, only normal data is available for training anomaly detection models, excluding supervision-based methods. Moreover, unlike the typical setting with abundant but unlabeled target domain data, we are faced with data scarcity in the target domain (i.e., the target wind turbine). Available studies using transfer learning or domain adaptation for improving unsupervised anomaly detection through NBMs are limited to fine-tuning [15] or simple corrections of the NBM [16].

Performing fault diagnosis for a WT with data scarcity by simply employing an NBM trained on another WT with abundant data is generally not feasible. Since the WTs vary in, for instance, power generation and drive train characteristics, the NBM trained on another WT will predict an incorrect expected state due to the different characteristics learned from its training set. However, if an NBM were employed from a WT exhibiting a minimal domain shift [17], i.e., if the 2 WT's test data distributions would match, it could be used for fault diagnosis for the WT with scarce training data. This would be the case if the two WTs shared the same specifications, operational behavior, and weather conditions. To this end, our work demonstrates how to make the WTs resemble each other by transforming their SCADA data into one another. We employ domain mapping [10] to map SCADA samples from one WT to resemble the corresponding SCADA data of another WT. Through this mapping, we can enable fault diagnosis using another WT's NBM. A critical component herein is that the mapping should preserve the sample's content when translating it to another domain. For example, a SCADA measurement capturing a WT running idly should remain in an idle state, and critically, anomalous behavior should be mapped to anomalous behavior across WT domains.

The authors of [18] proposed a generative neural network that synthesizes SCADA data resembling another WT but did not employ or consider any content preservation. Without explicitly enforcing a preservation of WT operational states, as employed in our study, anomalies may be mapped to healthy states, ultimately leading to unreliable fault diagnosis. Pattnaik et al. [19] presented a content-preserving CycleGAN[20]-based domain mapping approach for mapping industrial time series data across motors for bearing fault classification. While their work demonstrated the promising potential of mapping data between different but related machines, it followed the usual domain adaptation workflow of abundant but unlabeled data for classification, whereas our study investigates domain mapping for an unsupervised anomaly detection task using limited training data.

We propose a novel fault diagnosis approach based on WT domain mapping with CycleGAN that learns to map SCADA data between the source and the target WT domain in a content-preserving manner. At test time, the target data is mapped to the source domain, thereby enabling fault diagnosis using the source WT's NBM. To our knowledge, this study is the first to demonstrate a CycleGAN-based framework for unsupervised anomaly detection with time series as well as in wind energy. Our contributions are as follows:

i) Our study is the first to investigate and demonstrate domain mapping for unsupervised anomaly detection with time series for WT fault diagnosis.

ii) We show how our domain mapping technique can be used for data scarce WTs to improve NBM performance in fault diagnosis applications.

iii) We outline future research directions for domain mapping-based anomaly detection in wind energy.

2 Related Work

2.1 Domain adaptation

Formally, a domain $D = \{\mathcal{X}, P(X)\}$ consists of a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$ [9]. We aim to learn a task $T = \{\mathcal{Y}, P(Y|X)\}$, where $\mathcal{Y}$ is the label space and $P(Y|X)$ is the conditional probability. We differentiate between the source domain $\mathcal{D}_\mathcal{S}$ with its dataset $D_s = (x_{1s}, y_{1s}), \ldots, (x_{ns}, y_{ns})$ and the target domain $\mathcal{D}_T$ with its corresponding dataset $D_T = (x_{1t}, y_{1t}), \ldots, (x_{nt}, y_{nt})$, where usually $0 < n_t \ll n_s$. In our WT fault diagnosis context, the source domain is represented by a source WT with representative SCADA data ($n_s \gg 0$) and the target domain by a target WT with scarce, non-representative data ($n_t \ll n_s$). Transfer learning aspires to use knowledge extracted from D_S and by T_S to improve learning the task T_T in the target domain $\mathcal{D}_T \neq \mathcal{D}_\mathcal{S}$. Domain adaptation is a particular type of transductive transfer learning, a setting in which the task remains the same but across different domains [9], [10].

As acquiring labels for readily available large scale vision datasets is a prohibitively time intensive and expensive task, much attention has been devoted to unsupervised domain adaptation (labeled source domain and unlabeled target domain data) for vision related classification tasks [10]. Initial approaches were focused on learning domain-invariant representations of source and target features (e.g., [21]–[23]).

More recently, generative methods were able to improve performance in unsupervised domain adaptation [24], [25]. These methods extend domain adaptation with a generative component. The main approach is domain mapping (domain translation), which aims to map samples from one domain to another. Domain mapping can allow, e.g., the use of a pretrained source domain classifier for unlabeled target data which has been mapped to the source domain. An essential challenge herein is to retain important characteristics and semantics of the mapping input such as class characteristics, to ensure a correct and consistent classification. For instance, a handwritten digit of class 1 must at the very least remain of class 1 when mapped into a corresponding street view house numbers domain for a correct classification.

Unpaired image-to-image translation methods (e.g., [20], [26]) are fundamental to achieve such a content-preserving mapping. CycleGAN [20] enables the translation of unpaired samples into another domain with generative adversarial networks through a cycle-consistency loss, which enforces the preservation of content. Domain mapping based on CycleGAN has been used in various applications, e.g. in medical settings [27], voice conversion [28], as well as machine fault diagnosis [19]. For a more comprehensive review of discrepancy-based domain adaptation and domain mapping we refer to [24], [25].

2.2 Domain adaptation for wind turbine fault diagnosis

The potential of domain adaptation has been demonstrated with time series data from sensors across multiple applications, including non-WT industrial fault detection [19], [29], [30]. These applications primarily rely on classification through domain alignment with abundant target domain data lacking fault labels. For WTs specifically, discrepancy-based domain adaptation for fault classification is presented in [11]–[13], [31]. Results show significant gains in accuracy on the unlabeled target domains, even when target domain data is scarce [12]. An alternative approach to improve fault classification performance, namely by generative data augmentation, is proposed in [32] to generate missing working conditions for vibration data.

Yet, the available literature for our relevant anomaly detection task using NBMs is scarce. In our unsupervised anomaly detection setting, there is typically a target data scarcity, coupled with no available supervision (e.g., no fault labels), thereby rendering many previously presented domain adaptation techniques unsuitable for this task. Few conventional transfer learning approaches have been proposed: In [15], fine-tuning is successfully applied to an NBM pretrained on physics-informed simulation data. While the results show significantly improved performance for when only one month of SCADA data is available, the physics-informed simulation restricts the models to active power NBMs. A simple correction based on linear regression of the source domain NBM output is proposed in [16]. The proposed correction achieves slightly improved results over fine-tuning but is primarily implemented to avoid overfitting an NBM on a specific season, represented by a target training set comprising considerable 3 months of data.

The first generative domain adaptation work for WT NBMs is presented by Jin et al. [18]. Specifically, a generative adversarial network (GAN) is proposed to map samples from the scarce target domain, consisting of two weeks of SCADA data, to the source domain comprising data from a different WT. The GAN is conditioned on target data, that is, the input to the generator are SCADA samples from the target domain, rather than random noise. In doing so, the target domain input is mapped to samples matching the marginal probability distribution of the source domain. Finally, the mapped target data is used with an autoencoder-based NBM pretrained on the source domain for anomaly detection. Presented results demonstrate reliable anomaly detection despite target data scarcity. While this approach resembles domain mapping, it substantially deviates from previously outlined approaches. First, it consists of only one-directional target-to-source mappings. Crucially, the mapping network is also unconstrained, i.e., there is no constraint to preserve specific WT states (e.g., maximal power generation or anomalous operation behavior) to the corresponding state in the other domain. This may result in anomalous data being mapped to arbitrary healthy operation states. This lack of content preservation can therefore lead to unreliable fault diagnosis. Our study overcomes such deficiencies by employing domain mapping based on CycleGAN with consistency losses to preserve the SCADA content during domain translation.

As the only study to investigate constrained domain mapping for WTs, Chatterjee et al. [33] show only a marginal and negligible accuracy gain in blade icing detection when using CycleGAN to map images of plain ice-free rotor blades across a dataset of rotor blades with icing for generative data augmentation. The authors attributed the poor performance to possibly the small dataset size, large class imbalance, or their image generation process. Their work utilized images of rotor blades with all modes present and for synthetic data augmentation (i.e., convert images of plain blades to images of blades affected by icing), whereas our study investigates the translation of non-representative SCADA time series data for anomaly detection.

2.3 Domain mapping for unsupervised anomaly detection

Few studies exist on domain mapping applications for unsupervised anomaly detection, even beyond the wind energy research field. Hardly any domain adaptation studies exist for anomaly detection tasks as in the present study where only normal training samples are available combined with limited data in the target domain [14]. To our knowledge, our study is the first to propose a CycleGAN-based approach to unsupervised anomaly detection using domain mapping.

Moreover, only few works have even performed the underlying domain translation with time series data, i.e., mapping time series, regardless of the task and outside the scope of wind turbine SCADA data. CycleGAN is proposed for data augmentation by generating artificial damaged states of acceleration data in [34] through a mapping of undamaged-to-damaged conditions. Pattnaik et al. [19] translates time series across machines for bearing fault diagnosis using a CycleGAN framework. Unlike our anomaly detection study with scarce target data, their model maps abundant but unlabeled target domain data to the source domain where it is subsequently used in a fault classifier pretrained on the source domain. Results showed a significant outperformance compared to conventional domain adaptation methods when the domain shift is large, i.e., when mapping from substantially different but related machines.

3 Dataset

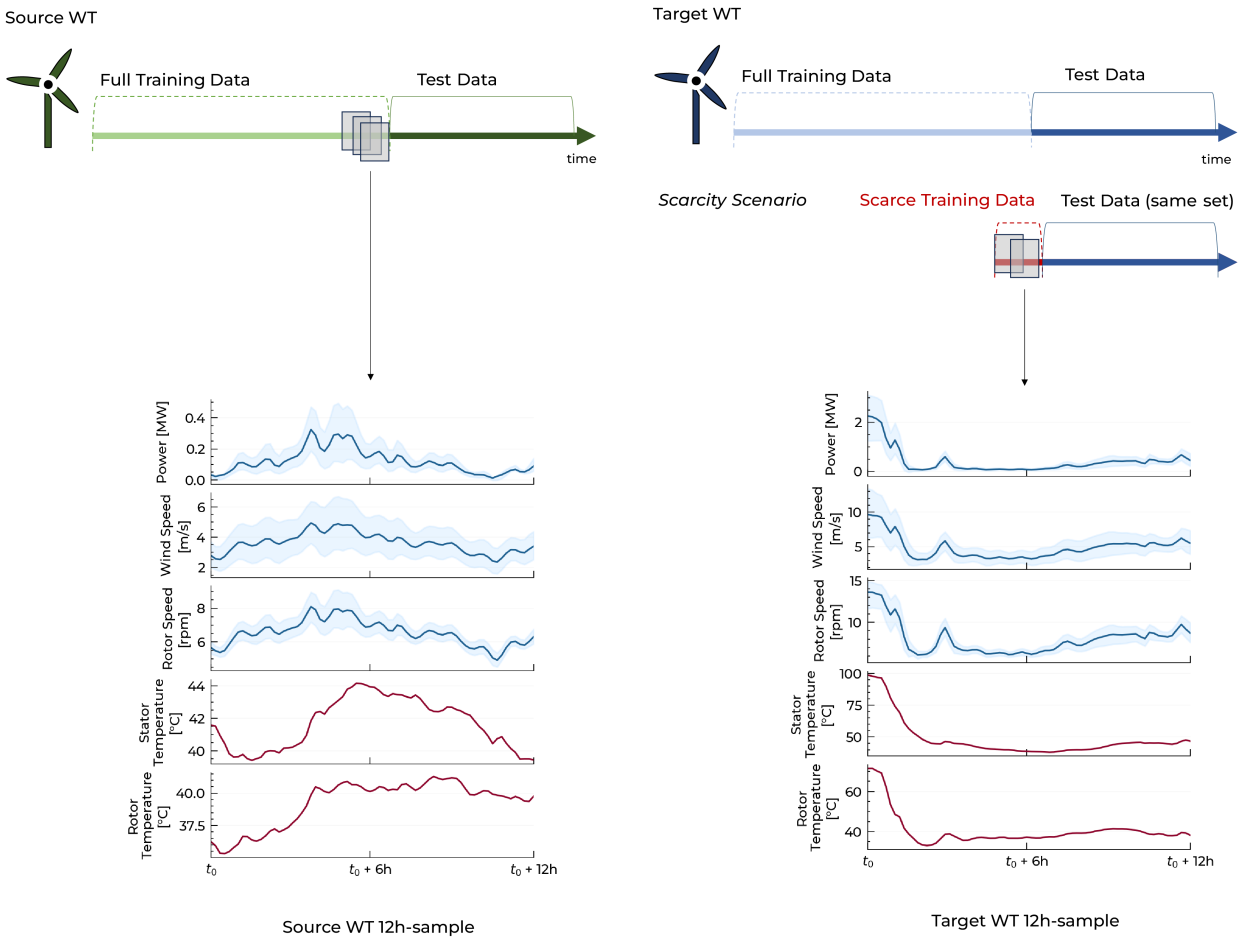


Figure 1: On the left, a source WT is selected with its data split into a training/validation and test set. On the right, we select a target WT which is also split into a preliminary training/validation and test set. For the target WT, a data scarcity scenario is then applied, in which only 1-8 weeks immediately preceding the target test set (which remains the same) are considered as the new training/validation set. For both WTs, we apply a sliding window approach to extract 12-hour SCADA samples from the sets, consisting of 11 channels in total. The shaded areas represent the corresponding maximum and minimum feature values.

Our available raw data comprises 10-minute averaged SCADA data of 7 wind turbines, each from a distinct wind farm. While the turbines share the same manufacturer to ensure a matching SCADA variable system, they exhibit significantly different characteristics, model types, power rating, maintenance and fault history, and geographical location. An incident log is further provided for each WT. These logs contain a binary flag for each 10-minute measurement, describing time frames during which unknown and unspecified incidents occurred in the WT operation. While these incidents do not necessarily represent anomalous behavior or faults, we consider no measurements during these marked time frames as coming from normal operation behavior. A more detailed overview and an illustration of the 7 WTs is provided in Appendix D.

Our study considers source-target pairs of WTs for domain adaptation, with one WT defining the source domain and the pair's second WT defining the target domain. Our goal is to map SCADA samples between the source domain, consisting of a randomly selected WT, and the target domain, represented by another WT of a different WT type from a different wind farm. The data from the source domain WT is split into a training set (first 70% of the data), a validation set (30% of the training data), and a test set (the last 30% of data). For the target domain WT, starting with its full dataset, we first apply the same 70-30-30 split as with the source WT. As a next step we apply a data scarcity scenario, by artificially shortening the training and validation set. Specifically, we shorten the training/validation set to

consecutive time frames of 1 to 8 weeks immediately preceding the test set, which remains the same across all scarcity scenarios. To ensure dataset size consistency when comparing across different WTs, we always refer to one week of data to comprise 1008 (6 10-minute sample per hour * 24 * 7 days) SCADA samples instead of selecting by calendar week, as maintenance, data gaps, or incidents could cause strongly changing dataset sizes across turbines.

These datasets are further normalized according to statistics calculated over the training set of the source domain such that each SCADA variable falls into a value range of $[-1, 1]$. The target domain data is normalized using the same min-max normalization formula while using source domain statistics and therefore not confined within this range.

From each set, we apply a sliding window approach to extract a sample dataset consisting of 12-hour samples of consecutive SCADA measurements with 11 variables. Selected were the mean, maximum, and minimum wind speed, rotor speed, power output values, as well as two temperatures from internal components, namely the mean stator and rotor temperature. Due to the one-to-one mapping inherent with CycleGAN [35], we refrained from using variables with a larger stochastic component such as the ambient temperature. For instance, an idle WT operational state, represented through our highly correlated 11 input features, could exist with numerous possible ambient temperature variations, which may necessitate probabilistic domain mapping (many-to-many mappings, e.g. [35]). The chosen 12-hours time frame allows us to better capture temporal dynamics within a sample, for instance thermal processes. An illustration of our dataset splits and resulting SCADA samples is shown in Figure 1.

Normal behavior models require that only normal samples, i.e., only healthy WT operation states, are used for training. We apply WT-specific filtering to exclude possibly abnormal samples from the training and validation sets. First, we exclude all measurements that fall into a time frame marked in the incident logs. Moreover, we apply a rated power filter rule and Mahalanobis distance-based filtering based on [36] to remove outliers, measurements from curtailment, and other outliers. No filtering is performed on the test sets.

4 Methodology

Our study aims to improve the reliability of fault diagnosis models for WTs with unrepresentative training data by leveraging models and data from similar, but not identical, WTs for which representative training data is available. To achieve this, we employ a generative domain adaptation approach, namely domain mapping. This section outlines our machine learning models, domain mapping losses, and evaluation metrics used. Our proposed framework consists of a domain mapping network that can translate source domain samples (consecutive 12-hour SCADA samples) into the target domain and vice versa. Despite the translation being learned cyclically and in both directions during training, we are ultimately interested in mapping (scarce) target domain data to the source domain for subsequent anomaly detection. The anomaly detection underlying the fault diagnosis of the mapped target domain data is finally achieved with an autoencoder-based NBM pretrained on the source domain. The proposed workflow is shown in Figure 2.

4.1 Autoencoder-based NBM

We employ an NBM based on autoencoders [37] for anomaly detection and fault diagnosis. Autoencoders are unsupervised neural networks trained to encode input data into a lower dimensional representation (encoder) and to subsequently reconstruct the input data from the compressed representation (decoder). To reconstruct the input as exactly as possible, an autoencoder learns to encode the most important features, thereby learning critical variable and feature relationships. An input sample that significantly deviates from the training set (i.e., an anomaly) will thus result at test time in an elevated reconstruction error, representing an anomaly score. Autoencoders have been employed as WT-specific NBMs (e.g., [38]–[40]). Our autoencoder takes as input 12-hour SCADA samples with 11 channels (792 data points) and learns to reconstruct them based on an encoding size (bottleneck dimension) of 72 units. A model is trained on only normal source domain training samples for each WT domain pair. At test time, we categorize all samples with a reconstruction error (mean absolute error between the autoencoder reconstruction output and its original input) above the defined WT-specific threshold as anomalous. The threshold is set to detect far out outliers [41] for all models as $T = q_3 + 3\,(q_3 - q_1)$, where q_3 and q_1 represent the 75th and 25th percentile of the normal validation data reconstruction errors, respectively. Further information about the autoencoder model is outlined in Appendix A.

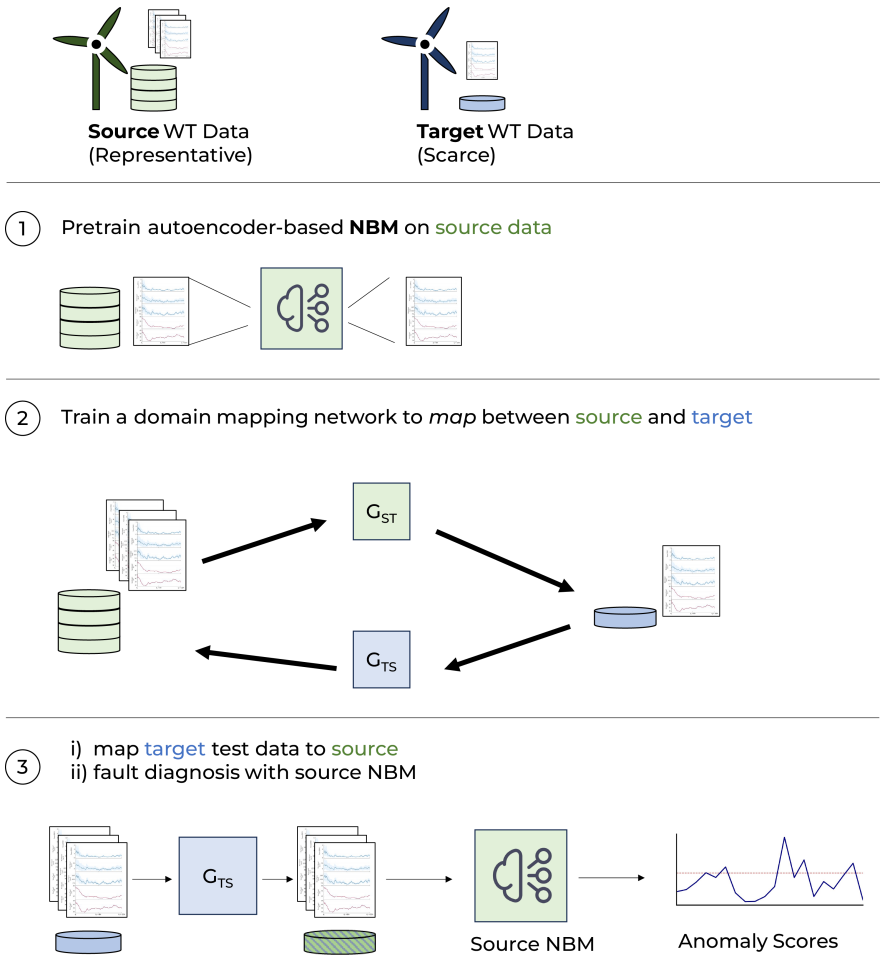


Figure 2: Workflow of the proposed approach. A source domain, represented by a WT with abundant data, and a target domain, represented by a WT with scarce data, are available. In step (1), an NBM is trained on the healthy, abundant source data only. This NBM can detect anomalies for the source WT but cannot be used for the target WT due to the domain shift. In (2), our proposed domain mapping technique using a CycleGAN is applied. The network learns to map source data to match target data and vice versa. In final step (3), the test set data of the target WT is first mapped to the source domain using the trained target-to-source network component (G_{TS}). As it now resembles source data, the pretrained NBM can be applied to obtain anomaly scores on the mapped data.

4.2 Domain mapping model

Our domain mapping network is based on the CycleGAN formulation [20]. The network consists of two one-directional generative adversarial networks (section 4.3) taking as input SCADA samples of one WT to transform them into data resembling the other WT. The generators are trained to synthesize realistic samples in an adversarial way by trying to outperform the WT-specific critics, which try to distinguish between fake (generated) and real samples of their particular WT. Unconstrained generators, as in the case of [18], can however potentially transform an input SCADA sample in any way and disregard the inherent content within the sample, which represents a particular state of the WT, e.g., idle power, maximal power production, or anomalous behavior. We propose and demonstrate that adding consistency losses (section 4.4) to the network enforces the content-preservation for anomaly detection. Most importantly, a cycle-consistency loss enforces that a sample first mapped to the other domain and then back to its original domain should remain similar. Figure 3 illustrates the concept of our domain mapping network.

In the following, we outline the specifications of the generators, discriminators, and content-preserving losses. Formally, we define a source domain $\mathcal{D}_S$ representing the source WT and a target domain $\mathcal{D}_T$ representing the target domain WT with domain datasets D_S, D_T containing respective SCADA samples here labeled as s and t.

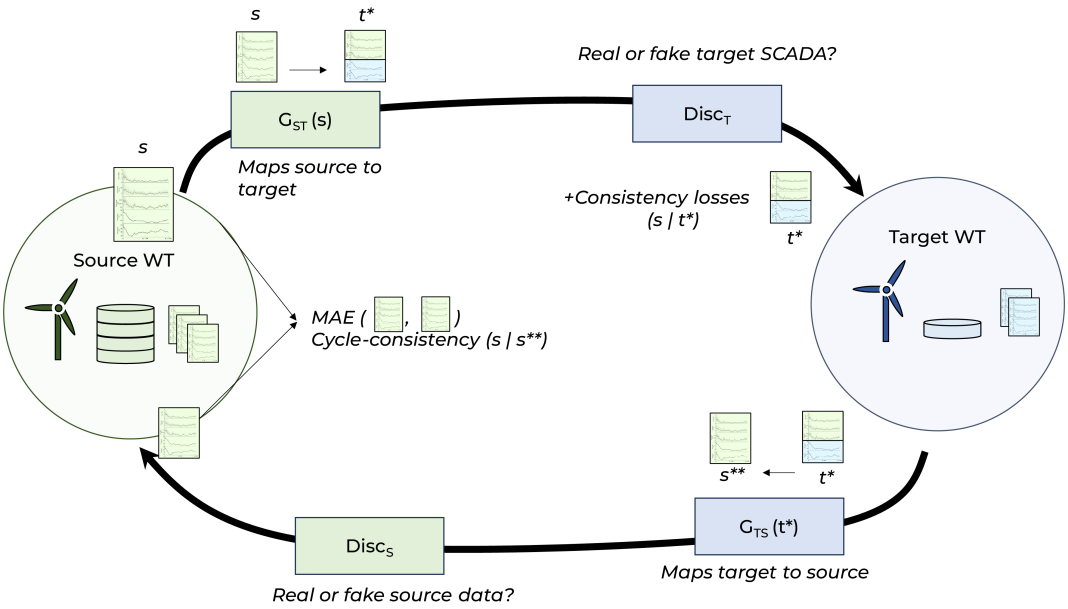


Figure 3: Illustration of the proposed domain mapping network. Visualized here is only the translation of a SCADA sample from the source domain (s) to the target domain and back, while our model maps in both directions. A source sample (s) is mapped by the generator for the respective domain direction (G_{ST}) to resemble data from the target domain under additional content-preservation constraints. The mapped sample (t^*) is mapped back to the source domain using G_{TS}, completing the cycle. The cycle-consistency forces the cycled sample (s^{**}) to closely resemble the original input.

4.3 Generators and critics

The framework consists of two generators G_{ST}, G_{TS} and two discriminators $Disc_T, Disc_S$. The generators are tasked to map a sample from one domain, which is their input, to resemble a sample from the other domain. There is one generator for each domain direction, i.e., $G_{ST} : \mathcal{D}_S \rightarrow \mathcal{D}_T, G_{TS} : \mathcal{D}_T \rightarrow \mathcal{D}_S$. The generator G_{ST} maps an input SCADA sample from $\mathcal{D}_S$ to resemble a sample belonging to domain $\mathcal{D}_T$. As opposed to the standard GAN formulation, the generators take as inputs real samples of a domain to translate, instead of synthesizing samples from random noise. The discriminators generally act as critics, assessing whether a sample belongs to the underlying probability distribution or not. $Disc_T(x)$ is generally a score reflecting whether a sample candidate x belongs to the corresponding domain distribution P_T or not. In our work, we use the GAN-QP framework [42], [43] as in [18] to train the generators and critics to generate realistic samples. Further details about model architectures, hyperparameter optimization, and complete training procedure are presented in Appendix B.

4.4 Content preservation losses

As the goal is to ultimately perform fault diagnosis with mapped data, it is critical to ensure that the generator mapping preserves the SCADA sample content, i.e., the operational state represented by the sample such as constant maximum power output and in particular anomalous behavior. The proposed cycle-consistency loss in CycleGAN [20] constrains the domain mapping to encourage that a mapped sample mapped back to its original domain resembles the original sample. Formally, it encourages $G_{ST}(G_{TS}(t)) \approx t$ and vice versa. This is achieved by adding an L1 loss to the generator loss punishing deviations between original and cycled samples:

$$\mathcal{L}_{cyc} = \lambda_{cyc} \left(MAE\left(t,\, G_{ST}\left(G_{TS}\left(t\right)\right)\right) \,+\, MAE\left(s,\, G_{TS}\left(G_{ST}\left(s\right)\right)\right)\right) \tag{1}$$

However, the cycle-consistency loss can be in our case insufficient to ensure a consistent content mapping across domains. To illustrate this, let us consider adding a hypothetical flipping operation to the generators. Typically, with image data, flipping an input will cause discriminators to reject the mapping. For instance, flipping a horse image upside down before translating it into a zebra will never generate a realistic zebra, whereas "flipping" a SCADA input (e.g., maximum power mapped to zero power) before translation can still generate realistic domain samples. As the same flipping operation is performed in the opposite domain direction, the cycled output will resemble the original

input, thereby still enforcing the cycle-consistency but without actually preserving content. To account for this, we further restrict the mapping space with additional consistency losses.

We add two physics-informed loss functions to construct our content-consistency loss. The first loss encourages the generators to map idle states in the original domain, for instance, zero active power output or no rotor rotation, to idle states in the other domain. Formally, we define all positions of selected channels (in our study: minimum, mean, maximum power and rotor rotation) within a SCADA sample x as a zero state if they are zero valued: $Z := x_{c,i} = 0; i = 1, ..., 72$, where c denotes the channel. The zero loss discourages deviations from the zero state when mapping to a domain:

$$\mathcal{L}_0 = \lambda_0 \left(MAE(G_{ST}(s\mathbb{1}_Z), 0) + MAE(G_{TS}(t\mathbb{1}_Z), 0) \right) \tag{2}$$

The rated power loss encourages that power outputs at a rated WT value remain at a rated value for the corresponding other WT. Let C_S and C_T represent the rated power of the source and target domain WT, respectively. All positions within a SCADA sample x are defined to be at a rated power if they match the WT capacity: $R := x_{c,i} = C_D; D =$ domain (S, T), $i = 1, ..., 72$, where c denotes the channel (in our study: the mean power). The rated power loss is then defined as:

$$\mathcal{L}_\mathcal{R} = \lambda_\mathcal{R} (MAE(G_{ST}(s\mathbb{1}_R), C_T) + MAE(G_{TS}(t\mathbb{1}_R), C_S)) \tag{3}$$

The consistency losses are added to the generator loss with relative weights $\lambda_{cyc}, \lambda_0, \lambda_R$, determined using a hyperparameter search. More details are outlined in Appendix B. We illustrate our consistency losses in Figure 4.

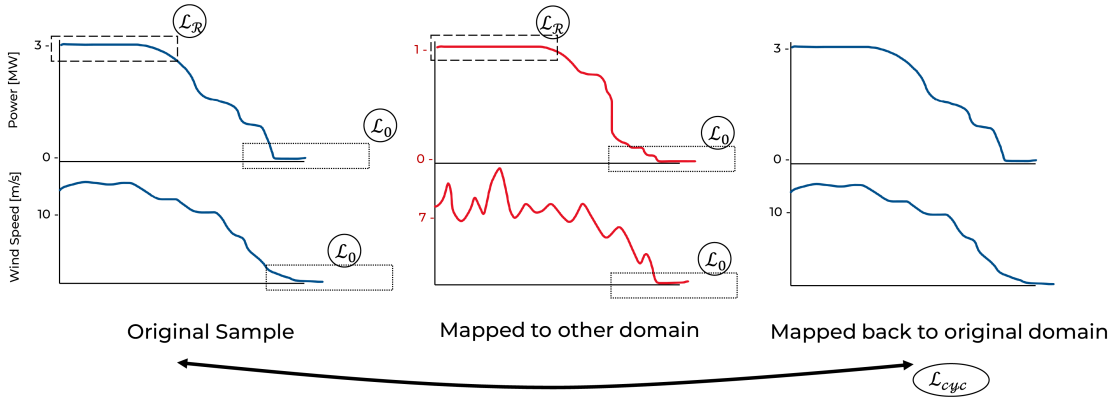


Figure 4: A sketch illustrating our content-consistency losses. An illustrative 12h-sample containing only the WT power output and wind speed is shown on the left, with its mapping to the other domain in the middle and mapped back to the original domain on the right. The zero-loss enforces that zero states (e.g., zero power) should be mapped to zero states in the other domain, shown by the boxed dashed regions marked by $\mathcal{L}_0$. The rated power loss ($\mathcal{L}_R$) enforces that when a WT is running at maximum capacity (in the figure: 3 MW) it should also run at the rated capacity in the other domain (in the figure: 1 MW). Finally, the cycle-consistency loss $\mathcal{L}_{cyc}$ ensures that the cycled sample (mapping a mapped sample back to its original domain) remains similar to its original sample.

4.5 Evaluation and benchmarks

Ultimately, we are interested in obtaining similar anomaly scores with scarce target data as if we had abundant target training data available. Therefore, we consider the test set anomaly scores from an NBM trained on abundant target training data (i.e., without any data scarcity scenario applied) as our ground truth in this study. More specifically for anomaly detection, we are interested in whether compared anomaly scores both exceed their (model-specific) threshold or not (anomalous or normal). We introduce a similarity measure in the following to compare data scarce models to the representative NBM.

Let $a^* = (a_1^*, \ldots, a_n^*)$ be the ground truth test set anomaly scores of the NBM trained on the full target domain training data (i.e., no data scarcity scenario applied), consisting of n test set samples with a model-specific threshold T^*. Let $a = (a_1, \ldots, a_n)$ be the test set anomaly scores of a compared NBM (e.g., anomaly scores of mapped data) with its threshold T. Anomaly scores are converted into a binary value expressing whether the score exceeds the threshold

(positive, 1) or not (negative, 0), i.e., $y^* = a_1 \geq T^*, \ldots, a_n \geq T^*$ and $y = a_1 \geq T, \ldots, a_n \geq T$. We compare the binary values using classification metrics to obtain the performance as the F1-score, defined as:

$$\text{F1-Score} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{4}$$

We compare the performance of our domain mapping technique with two benchmarks: The first one is an NBM trained on scarce target data only, providing a baseline without domain adaptation and SCADA data from existing wind farms. The second one is a fine-tuning benchmark that represents the performance of a conventional and simple domain adaptation approach for this task. Starting with a pretrained NBM trained on SCADA samples of the source domain WT, we fine-tune the model on the available scarce target training data. Our benchmark models and the evaluation are illustrated in Figure 5.

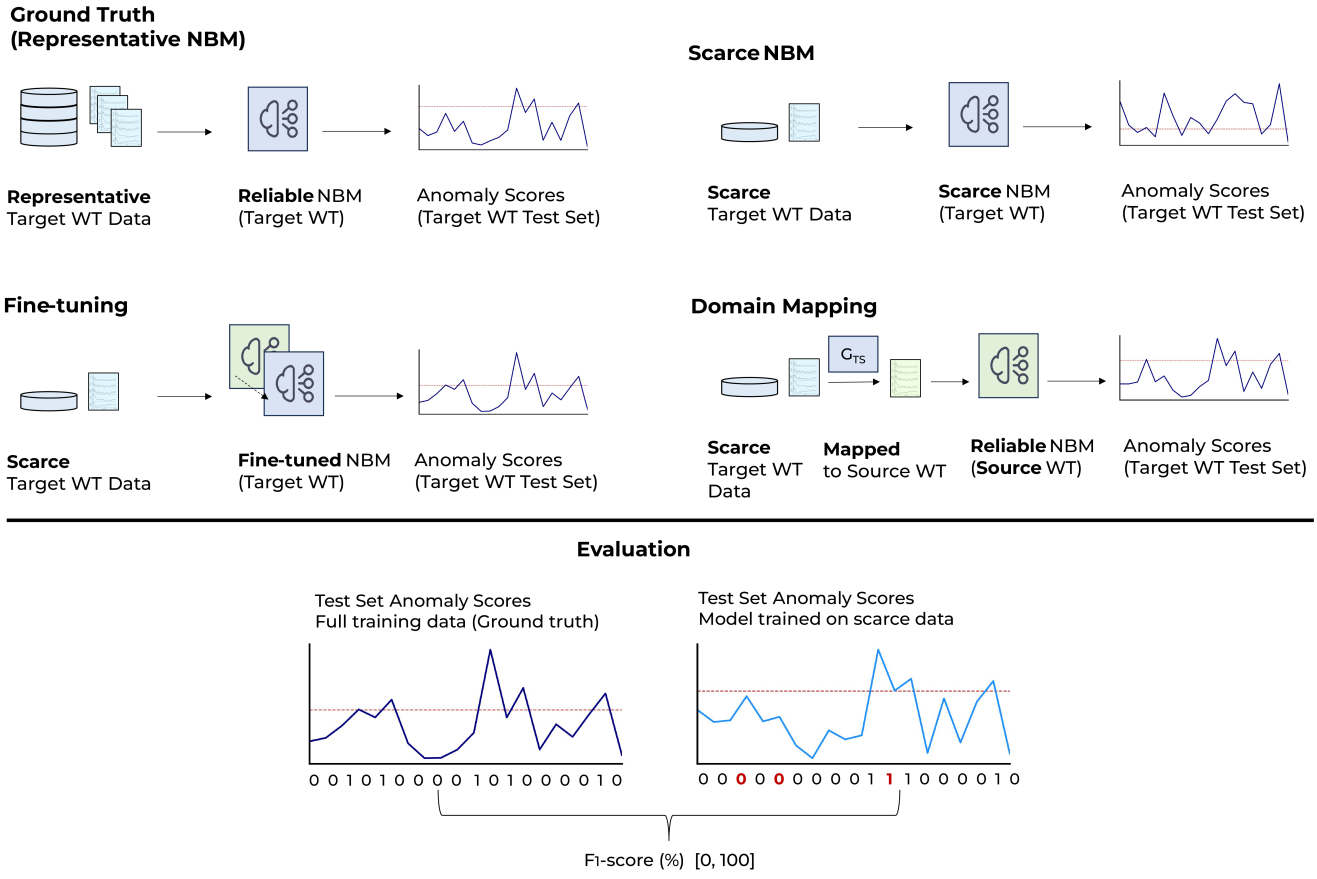


Figure 5: All approaches for comparison are illustrated at the top. In the upper left, we consider an NBM trained on abundant target WT data (no data scarcity) producing anomaly scores on a fixed test set, representing our ground truth. We compare these anomaly scores with an NBM trained on scarce target training data only (upper right), a fine-tuning approach (lower left) and our domain mapping network (lower right). These obtained anomaly scores, sharing the same test set, are compared in terms of a threshold similarity (F1-score) to the ground truth (illustrated at the bottom).

5 Results and Discussion

5.1 Fault diagnosis performance

We assess the fault diagnosis performance of our proposed domain mapping network in terms of producing similar anomaly scores as a target WT NBM trained on abundant data, as outlined in Figure 5. In total, we evaluated 28 source-target WT domain pairs using 6 distinct target domain WTs, as one of the seven WTs (WT07) was used during model selection and hyper-parameter optimization (see Appendix B), with varying degrees of target training data scarcity of 1-8 weeks. All domain pairs consist of a source and target WT differing in turbine specifications in terms of model and having a unique rated power capacity (see Appendix D), farm, and geographical location. For each source-target domain pair, we compared the performance to an NBM trained on scarce data only and a fine-tuning benchmark.

Table 1: F1-scores (in %) across all 6 target wind turbines evaluating the threshold similarity of NBMs trained on various degrees of limited training data to the ground truth of the respective NBMs trained on a full training dataset.

Target WT	2 months	6 weeks	1 month	3 weeks	2 weeks	1 week
WT01	89.0	84.6	77.4	74.4	77.1	66.4
WT02	77.8	62.6	65.3	60.3	73.6	35.2
WT03	84.7	73.8	64.8	38.4	52.6	34.5
WT04	92.6	94.0	88.3	89.8	42.8	34.4
WT05	67.6	55.3	63.9	56.5	31.5	24.3
WT06	94.5	42.4	24.7	31.1	30.3	7.0

Training NBMs on scarce data. For all 6 target WTs considered in our study, we first trained NBMs using only scarce target training data under varying scarcity scenarios. That is, we trained an NBM using 1-8 weeks of training data and evaluated the resulting anomaly scores on the fixed test set in comparison to the ground truth. Our findings in Table 1 show that NBMs trained on limited training sets achieved comparably poor fault diagnosis performance, with the mean F1-score generally decreasing as less training data becomes available. While a training set comprising 2 months of data achieved on average an F1-score of 84.3% across all 6 target WTs, the mean score substantially decreased to 51.3% when only 2 weeks are considered, showing a strong drop in similarity of anomaly scores compared to the NBM trained on a full training dataset. Limited data can lead to unrepresentative training sets with an insufficient coverage of samples from normal operational behavior. An example of representative and unrepresentative training data is shown in Appendix E. Thus, the autoencoder-based NBM becomes incapable of reconstructing normal data at test time that was missing from the training set. Our results indicating that training NBMs under data scarcity can result in unreliable fault diagnosis support the findings of e.g., [6], [7].

Improving fault diagnosis with domain adaptation. To overcome these limitations, we employed domain adaptation techniques. We trained and evaluated our proposed domain mapping technique and fine-tuning as comparison benchmark to overcome these limitations. We present the achieved improvements in F1-score compared to training an NBM on scarce data only (i.e., baseline without domain adaptation) for each domain pair in Table 2, further averaged by data scarcity degree. The absolute F1-scores are available in Table A4 in Appendix C. Both fine-tuning and our domain mapping network achieved an improvement in fault diagnosis performance in scenarios with strong data scarcity, i.e., particularly in situations where the scarce NBM fails to achieve reliable fault diagnosis.

As shown in Table 2, both domain adaptation methods managed to to improve fault diagnosis performance over training on scarce data alone in most domain pairs and across almost all data scarcity degrees. Fine-tuning was able to improve the performance by increasing the F1-score on average by +4.7% points when only 1 month of target training data was available and +9.3% with 2 weeks of data, showing improvements across all scarcity scenarios except 2 months. These results further suggest that fine-tuning can be employed to adapt knowledge embedded in the source domain NBM to a limited target domain.

Notably, our proposed domain mapping approach outperformed fine-tuning across all considered training set sizes, especially in scenarios with severe data scarcity. In the scenarios ranging from 1 week to 1 month of data, our domain mapping approach achieved a substantial performance gain over not only the NBM trained on scarce data but also consistently over fine-tuning, with an increase in mean F1 score by +10.3% points for 1 month and +16.8% for 2 weeks. Our presented domain mapping approach makes use of the entire source WT's dataset to learn mappings and uses its

Table 2: Change in F1-score value compared to the respective NBM trained on scarce data only for all 28 domain pairs and data scarcity degrees.

		2 months		6 weeks		1 month		3 weeks		2 weeks		1 week	
Target	**Source**	**FT**	**Ours**	**FT**	**Ours**	**FT**	**Ours**	**FT**	**Ours**	**FT**	**Ours**	**FT**	**Ours**
WT01	WT07	+0.1	-1.1	+3.9	+2.1	+0.0	+8.6	-1.3	-17.1	-3.1	+2.3	+8.3	+12.4
	WT02	-32.2	-24.6	-7.2	+1.7	-5.3	+12.1	-4.1	+5.2	-4.2	+6.0	+6.0	+10.3
	WT05	-5.8	-6.7	-3.7	+3.7	-3.8	+6.6	+0.7	+13.6	+1.8	+6.4	+8.9	+9.2
	WT04	-3.8	+1.7	-3.9	+5.8	+0.9	+6.6	-2.9	+16.4	-4.4	+11.8	+0.3	+12.1
WT02	WT01	+2.8	-27.7	+21.1	+2.0	+19.0	-9.0	+13.6	-3.7	-40.9	-31.1	-33.3	+6.7
	WT06	-15.6	-6.1	+18.3	+6.4	+18.6	+14.6	+14.6	+16.8	-32.9	-8.9	-26.1	+26.9
	WT05	+4.4	-14.3	+18.2	+1.8	+9.2	-0.2	+17.4	+5.2	+9.0	-12.4	+36.1	+26.3
	WT04	+7.0	-13.0	+21.3	+4.4	+7.3	+1.8	+18.3	+10.4	+7.0	-10.2	+30.9	+19.9
WT03	WT01	-48.6	+1.8	+2.8	+13.0	-7.9	+19.9	+38.4	+34.9	+14.5	+20.9	-17.4	+6.2
	WT06	-41.5	-5.5	+4.6	+5.8	+2.2	+7.2	+36.8	+2.1	+10.1	+32.7	-16.7	+7.2
	WT07	-31.4	+2.1	-2.1	+11.8	+14.4	-16.2	+45.7	+7.6	+23.6	+36.6	+20.3	+15.7
	WT02	-31.4	-6.2	+5.1	+9.6	+21.6	-16.1	+35.3	+37.8	+13.8	+26.0	+6.5	+17.9
	WT05	+4.2	-12.2	+16.1	+11.7	+12.0	+17.6	+40.5	+47.2	+36.1	+20.6	+56.3	+27.0
	WT04	-3.0	-6.9	+16.0	+5.5	+9.9	+1.0	+43.3	+46.2	+33.2	+17.3	+30.6	+14.2
WT04	WT01	-12.7	-17.2	-23.1	-15.1	-9.5	-3.3	-22.5	-7.6	-0.5	+11.7	+2.4	+9.0
	WT06	-6.2	-10.2	-8.3	-21.7	-7.3	-7.8	-25.6	-3.1	+2.5	+6.9	+7.9	+12.6
	WT07	-4.0	-8.3	-1.7	-9.1	+1.6	-5.8	-6.5	-10.9	+17.6	+23.5	+22.9	+26.8
	WT02	-4.1	-0.3	-12.9	-4.4	-0.5	+3.2	-7.1	-4.5	+20.5	+27.1	+18.0	+27.1
	WT05	-0.9	+0.9	-2.2	-0.4	+4.5	+6.4	+2.3	+3.7	+44.5	+18.2	+59.2	+29.0
WT05	WT01	-2.5	-5.7	+9.6	-3.7	-10.1	-1.5	-10.1	-11.3	-5.5	+1.7	+7.6	-4.8
	WT06	-1.8	+10.7	+8.2	+21.3	-9.1	+6.8	-6.5	+24.3	+2.6	+47.2	-11.2	+23.1
	WT07	+19.8	+8.1	+28.2	+20.8	+4.9	+12.6	+12.6	+22.3	+32.9	+16.4	+36.5	+27.8
	WT02	+19.8	+14.2	+31.2	+25.4	+19.3	+22.2	+20.1	+32.4	+40.2	+31.3	+25.1	+28.8
	WT04	+6.1	+25.6	+10.2	+32.8	+6.9	+9.5	+11.8	+27.0	+51.0	+48.0	+26.9	+33.3
WT06	WT07	-27.2	-28.3	+16.8	+18.4	+16.1	+64.9	+5.2	+18.6	+7.7	+20.8	-0.7	+3.8
	WT02	-30.8	-17.8	+18.2	+31.3	+17.8	+49.7	+8.2	+16.0	+2.3	+9.8	-1.0	-0.1
	WT05	-19.6	+0.4	+0.5	+15.4	-4.0	+29.3	-12.8	+44.6	-10.8	+39.7	-0.9	-1.3
	WT04	-15.2	+0.4	+17.3	+34.1	+3.4	+47.5	-15.1	+44.8	-8.7	+48.6	-1.1	+0.0
Average [+- std.]		-9.8 [17.4]	**-5.2** [12.2]	+7.2 [12.9]	**+8.2** [13.6]	+4.7 [9.9]	**+10.3** [18.9]	+8.9 [20.3]	**+15.0** [19.1]	+9.3 [21.4]	**+16.8** [19.2]	+10.8 [22.3]	**+15.2** [11.0]

reliable model for anomaly detection, as opposed to fine-tuning, which exclusively relies on only the trained source NBM parameters being adjusted by a limited target training set, thereby limiting its acquired knowledge of the domain shift. Our findings highlight the effectiveness and potential of our domain mapping method as an alternative approach to conventional fine-tuning to mitigate the challenges of data scarcity in WT fault diagnosis. Our proposed technique can enable more reliable and earlier fault diagnosis, for instance for newly installed wind turbines, by incorporating models and data of only one reference source wind turbine with abundant data.

Performance decreases with abundant data. Conversely however, in scenarios with abundant data (e.g., 2 months), both fine-tuning and domain mapping showed limited improvements or even significant performance drops. A large decrease in performance can particularly be observed when the scarce NBM already achieved comparably high F1-scores, indicating representative training data (e.g., target WT04 in Table 1). This suggests that when enough data is available to train a reliable NBM, the benefits of these domain adaptation techniques may diminish. A drop in performance can likely be attributed to a loss of information through the fine-tuning and domain mapping operation, coupled with our hyperparameter optimization for both methods having been performed for 1 month of available data, which may result in suboptimal training when abundant data is available. That is, our domain adaptation methods were

optimized to be more constrained. In principle, fine-tuning should be able to overfit on the target data (completely disregarding any source knowledge), thereby achieving the same performance as if it were a new model exclusively trained on target data. Equivalently, the content-consistency loss constraints of our domain mapping could be more relaxed, allowing our model to generate samples more freely when enough diverse samples are available. Such adjustments would however require a priori information about the representativeness of the training set compared to the test set (which is by definition unavailable at training time), consequently restricting the possibilities for adequate case-by-case tuning. More research is needed to identify and counteract the loss of information as well as to improve hyperparameter optimization to preemptively detect or mitigate performance decreases in these scenarios.

WT-specific differences in performance changes. Lastly, we note strong differences in performance across target WTs. These could be attributed to the randomness of the selected training set weeks (e.g., contained weather conditions, variety of operational states), the test set characteristics (e.g., number of faults), general turbine behavior, and the domain pair, exhibiting varying domain shifts across different WTs. Further research is needed to investigate these variations and to possibly adjust the network and hyperparameters to the specific characteristics of WTs. For certain domain pairs, fine-tuning may retrospectively turn out to be a better solution compared to domain mapping. In some few other cases, both domain adaptation methods may even lead to a substantial drop in performance even with limited training data. Further investigations are required to clearly identify these causes and for finding mitigation strategies. These variations may cause uncertainties regarding an appropriate method selection, as due to the fundamental lack of knowledge about the test set, it remains unclear which approach to choose at training time.

5.2 Detailed analysis of a domain pair example

We present a detailed comparison of the fault diagnosis performance across all methods for one specific domain pair with the source domain WT "WT07" and target domain WT "WT05" with 1 month of available training data. The performance gains for this domain pair represent approximately the average for this scarcity scenario.

5.2.1 NBM trained on representative data

To obtain our ground truth anomaly scores for the target WT test set, we trained an NBM on the full target training data, which contains measurements recorded over a span of more than 2 years (Appendix D). We visualize the resulting anomaly scores, i.e., the NBM reconstruction errors, for the target WT test set in Figure 6. The scores generally exceed the threshold during time frames when incidents were logged, yet not for all incidents (e.g., in mid-October 2023). As the incidents are unspecified, we are unable to categorize this as a missed fault or correct fault diagnosis. During normal operation, the scores tend to remain below the threshold, although certain short periods exhibit elevated scores (e.g.,in the middle of January 2024), which we cannot determine as false positives or identified but unlogged abnormal states. However, we only focus on comparing these scores to the ones obtained trained with scarce data.

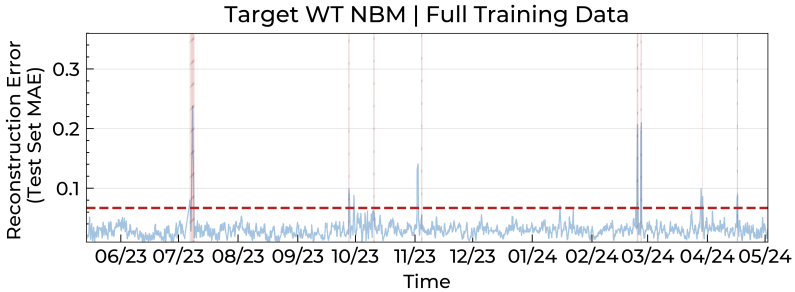


Figure 6: Anomaly scores, i.e., reconstruction errors, of the target WT test set obtained from the representative NBM. The model-specific threshold is shown in the horizontal red dotted line, while time frames with logged incidents are shown by red shaded areas.

5.2.2 NBM trained on scarce data

An NBM was then trained on only 1 month of target training data preceding the identical test set. The anomaly scores of this model are visualized in Figure 7. When comparing the threshold similarity, this model achieved an F1-score of 63.9% (Table A4). Notably, numerous anomaly scores falsely exceed the threshold compared to their counterpart in the ground truth NBM (e.g., January 2024), while scores during incidents largely match the ones from the representative

NBM. As autoencoders learn normal behavior from the training set, this could indicate that normal operational states were missing in the limited training set, causing high reconstruction errors at test time and therefore falsely elevated anomaly scores. Overall, a significant decrease in fault diagnosis performance can be observed when only scarce training data is available.

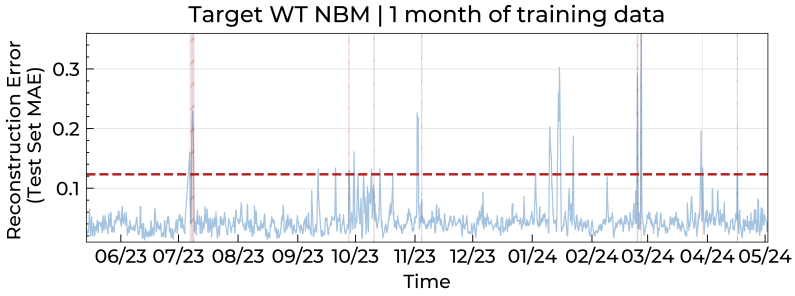


Figure 7: Anomaly scores of the always identical target WT test set obtained from the NBM trained on scarce training data.

5.2.3 Fine-tuning the source domain NBM

As a domain adaptation benchmark, an NBM was first trained on the abundant *source* training data, i.e., the full training set of WT07, and then *fine-tuned* using 1 month of target domain training data. Results are shown in Figure 8. This model achieved an increase in F1-score to 68.8% (+4.9 percentage points) compared to the scarce NBM. In particular, we notice a decrease in falsely elevated scores (for instance, October 2023) while anomaly scores generally remain elevated when they are in the ground truth. Our results suggest that fine-tuning is capable of adjusting information embedded in the source domain NBM to the target domain, resulting in unseen normal data missing in the target WT to be still considered normal, enabling more reliable fault diagnosis when training data is lacking.

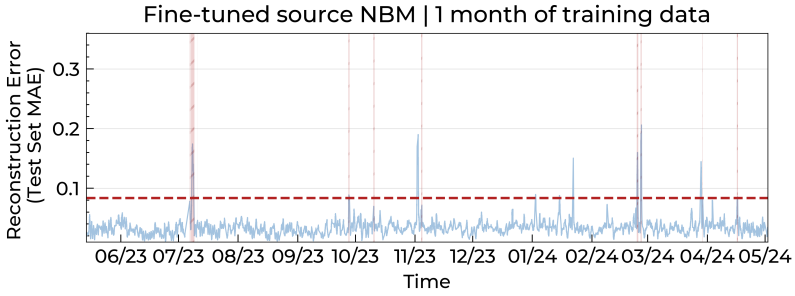


Figure 8: Anomaly scores of the target WT test set from the fine-tuned source NBM using scarce target training data.

5.2.4 Domain mapping

A mapping network was trained to map samples from the source WT to the scarce target WT and vice versa. After training was finished, the target-to-source model (G_{TS}) was used to map the target WT test set samples to the source domain. The mapped samples were subsequently used with the source domain NBM to obtain anomaly scores, which are shown in Figure 9. Our proposed technique achieved further performance gains over fine-tuning. The F1-score was significantly increased to 76.5% (+12.6), achieving the most similar anomaly scores compared to the ground truth.

This strong correspondence in anomaly scores indicates that our mapping network generates mappings closely resembling the source WT, as reconstruction errors for normal samples remain below the threshold, while additionally preserving the content, as anomalies are mapped to anomalous states. An example of a mapping using a target WT test set sample is illustrated in Figure 10, showing preserved operational behavior, i.e., high and low power was mapped to a correspondingly scaled high and low power output with consistent component temperature behavior.

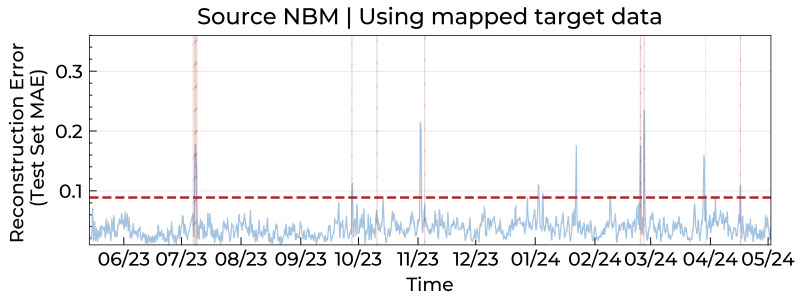


Figure 9: Anomaly scores of the target WT test set. The reconstruction errors are obtained from the pretrained source WT NBM using the mapped target test set data.

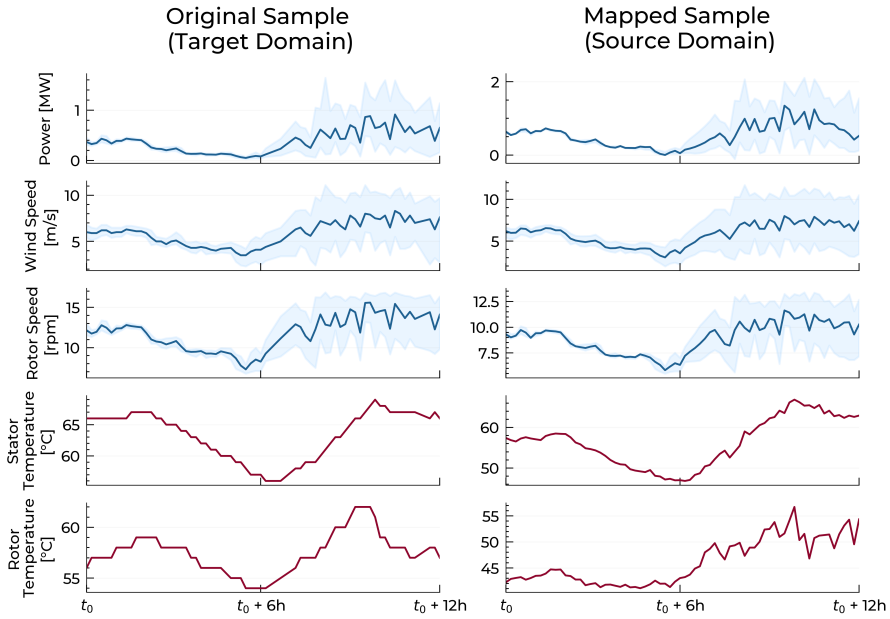


Figure 10: Example of the domain mapping result. A sample from the target WT's test set (left) is mapped by the trained network G_{TS} to the source domain, shown on the right.

6 Conclusion

This study investigated the application of generative domain adaptation, specifically domain mapping, to address the challenge of limited training data when using normal behavior models in wind turbine fault diagnosis. NBMs require substantial normal operation data, which when missing can result in unreliable diagnosis models. To address this challenge, we proposed a novel domain adaptation approach that leverages data from a WT of a different wind farm for which abundant training data is available. Our domain mapping technique transforms SCADA data from a data scarce target domain WT to resemble data from a different WT with representative training data and thus a reliable NBM. This mapping allows for the use of the source WT's NBM for fault diagnosis in the target WT despite limited data. We validated our method by conducting experiments using 28 different WT pairs across varying degrees of data scarcity (1 to 8 weeks). Our findings demonstrate the superior performance of generative domain adaptation compared to training models on limited data and conventional fine-tuning, particularly when faced with severe data scarcity. Our results highlight the significant potential of domain mapping in achieving earlier and more reliable WT fault diagnosis models when only scarce training data is available, for instance in newly installed wind farms.

Our study has several limitations which provide opportunities for future research. First, further exploration of time-series-based model architectures and frameworks, consistency losses, and generalizable hyperparameter optimization would be useful to further investigate the potential of domain mapping for WT fault diagnosis. In our study, hyperparameter

tuning was evaluated on the test set of one selected domain pair using 1 month of target training data. It is possible that certain types of WT pairs exhibiting varying domain shifts may require differently weighted loss constraints and models. More generalizable optimization techniques could be investigated, e.g., based on a quantification of the domain shift. Our domain mapping network furthermore exhibits a significantly higher complexity and a computationally heavier and longer training procedure compared to fine-tuning, representing a consideration for practical applications. While we validated our approach on a very comprehensive dataset comprising real operational data from numerous WTs from different wind farms, our experiments were limited to WTs of the same manufacturer and a selection of SCADA features. Further large-scale experiments with more versatile wind farms and WT types, as well as different SCADA variables and systems, will be valuable to assess the generalizability of our results to other manufacturers. Lastly, evaluating our method under various types of faults (if logs are available) could provide insights in the limitations of mapping types of anomalies and lead to further improvements to the domain adaptation approach.

Our exploratory work suggests a promising potential for further domain mapping applications. For WT fault diagnosis, this includes investigating the use of different architectures and adjusted techniques to establish domain mapping as a novel and effective alternative to fine-tuning. The proposed technique moreover highlights a potential for applications beyond WT systems. Exploring its applicability to diagnosis tasks under data scarcity of other areas, such as for photovoltaic systems, and generally anomaly detection tasks, is an interesting area of future research to explore domain mapping for more accurate and reliable models when faced with limited training data.

Acknowledgements

This research was funded by the Swiss National Science Foundation (grant number 206342). We want to thank aventron AG (Weidenstrasse 27, 4142 Münchenstein, Switzerland, www.aventron.com) for sharing their measurement data enabling this research.

References

[1] S. Faulstich, B. Hahn, and P. J. Tavner, "Wind turbine downtime and its importance for offshore deployment," *Wind Energy*, vol. 14, no. 3, pp. 327–337, Apr. 2011. DOI: 10.1002/we.421.

[2] G. Helbing and M. Ritter, "Deep Learning for fault detection in wind turbines," *Renewable and Sustainable Energy Reviews*, vol. 98, pp. 189–198, Dec. 2018. DOI: 10.1016/j.rser.2018.09.012.

[3] J. Tautz-Weinert and S. J. Watson, "Using SCADA data for wind turbine condition monitoring – a review," *IET Renewable Power Generation*, vol. 11, no. 4, pp. 382–394, Mar. 2017. DOI: 10.1049/iet-rpg.2016.0248.

[4] A. Stetco, F. Dinmohammadi, X. Zhao, *et al.*, "Machine learning methods for wind turbine condition monitoring: A review," *Renewable Energy*, vol. 133, pp. 620–635, Apr. 2019. DOI: 10.1016/j.renene.2018.10.047.

[5] A. Meyer, "Multi-target normal behaviour models for wind farm condition monitoring," *Applied Energy*, vol. 300, p. 117 342, Oct. 2021. DOI: 10.1016/j.apenergy.2021.117342.

[6] A. Grataloup, S. Jonas, and A. Meyer, *Wind turbine condition monitoring based on intra- and inter-farm federated learning*, 2024. DOI: 10.48550/ARXIV.2409.03672.

[7] L. Jenkel, S. Jonas, and A. Meyer, "Privacy-Preserving Fleet-Wide Learning of Wind Turbine Conditions with Federated Learning," *Energies*, vol. 16, no. 17, p. 6377, Sep. 2023. DOI: 10.3390/en16176377.

[8] A. Kusiak, "Renewables: Share data on wind energy," *Nature*, vol. 529, no. 7584, pp. 19–21, Jan. 2016. DOI: 10.1038/529019a.

[9] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010. DOI: 10.1109/TKDE.2009.191.

[10] G. Wilson and D. J. Cook, "A Survey of Unsupervised Deep Domain Adaptation," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 5, pp. 1–46, Oct. 2020. DOI: 10.1145/3400066.

[11] H. Ren, W. Liu, M. Shan, and X. Wang, "A new wind turbine health condition monitoring method based on VMD-MPE and feature-based transfer learning," *Measurement*, vol. 148, p. 106 906, Dec. 2019. DOI: 10.1016/j.measurement.2019.106906.

[12] P. Xie, X. Zhang, G. Jiang, J. Cui, and Q. He, "Investigation of deep transfer learning for cross-turbine diagnosis of wind turbine faults," *Measurement Science and Technology*, vol. 34, no. 4, p. 044 009, Apr. 2023. DOI: 10.1088/1361-6501/acadf7.

[13] Y. Zhu, C. Zhu, J. Tan, Y. Tan, and L. Rao, "Anomaly detection and condition monitoring of wind turbine gearbox based on LSTM-FS and transfer learning," *Renewable Energy*, vol. 189, pp. 90–103, Apr. 2022. DOI: 10.1016/j.renene.2022.02.061.

[14] Z. Yang, I. Soltani, and E. Darve, "Anomaly Detection With Domain Adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2023, pp. 2958–2967.

[15] L. Schröder, N. K. Dimitrov, D. R. Verelst, and J. A. Sørensen, "Using Transfer Learning to Build Physics-Informed Machine Learning Models for Improved Wind Farm Monitoring," *Energies*, vol. 15, no. 2, p. 558, Jan. 2022. DOI: 10.3390/en15020558.

[16] J. Zgraggen, M. Ulmer, E. Jarlskog, G. Pizza, and L. Goren Huber, "Transfer Learning Approaches for Wind Turbine Fault Detection using Deep Learning," *PHM Society European Conference*, vol. 6, no. 1, p. 12, Jun. 2021. DOI: 10.36001/phme.2021.v6i1.2835.

[17] S. Zhao, X. Yue, S. Zhang, *et al.*, "A review of single-source deep unsupervised visual domain adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 473–493, 2022. DOI: 10.1109/TNNLS.2020.3028503.

[18] X. Jin, H. Wang, Z. Kong, Z. Xu, and W. Qiao, "Condition Monitoring of Wind Turbine Generators Based on SCADA Data and Feature Transfer Learning," *IEEE Access*, vol. 11, pp. 9441–9450, 2023. DOI: 10.1109/ACCESS.2023.3240306.

[19] N. Pattnaik, U. S. Vemula, K. Kumar, *et al.*, "CycleGAN Based Unsupervised Domain Adaptation for Machine Fault Diagnosis," in *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, Boston Massachusetts: ACM, Nov. 2022, pp. 973–979. DOI: 10.1145/3560905.3568303.

[20] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[21] Y. Ganin and V. Lempitsky, "Unsupervised Domain Adaptation by Backpropagation," in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 1180–1189.

[22] B. Sun and K. Saenko, "Deep CORAL: Correlation Alignment for Deep Domain Adaptation," in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds., vol. 9915, Cham: Springer International Publishing, 2016, pp. 443–450. DOI: 10.1007/978-3-319-49409-8_35.

[23] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, *Deep Domain Confusion: Maximizing for Domain Invariance*, 2014. DOI: 10.48550/ARXIV.1412.3474.

[24] X. Liu, C. Yoo, F. Xing, *et al.*, "Deep unsupervised domain adaptation: A review of recent advances and perspectives," *APSIPA Transactions on Signal and Information Processing*, vol. 11, no. 1, 2022.

[25] S. Zhao, X. Yue, S. Zhang, *et al.*, "A Review of Single-Source Deep Unsupervised Visual Domain Adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 473–493, 2022. DOI: 10.1109/TNNLS.2020.3028503.

[26] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17, Long Beach, California, USA: Curran Associates Inc., 2017, pp. 700–708.

[27] J. A. Palladino, D. F. Slezak, and E. Ferrante, *Unsupervised Domain Adaptation via CycleGAN for White Matter Hyperintensity Segmentation in Multicenter MR Images*, 2020. DOI: 10.48550/ARXIV.2009.04985.

[28] T. Kaneko and H. Kameoka, "Cyclegan-vc: Non-parallel voice conversion using cycle-consistent adversarial networks," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2100–2104. DOI: 10.23919/EUSIPCO.2018.8553236.

[29] Y. Shi, X. Ying, and J. Yang, "Deep Unsupervised Domain Adaptation with Time Series Sensor Data: A Survey," *Sensors*, vol. 22, no. 15, p. 5507, Jul. 2022. DOI: 10.3390/s22155507.

[30] P. Yan, A. Abdulkadir, P.-P. Luley, *et al.*, "A Comprehensive Survey of Deep Transfer Learning for Anomaly Detection in Industrial Time Series: Methods, Applications, and Directions," *IEEE Access*, vol. 12, pp. 3768–3789, 2024. DOI: 10.1109/ACCESS.2023.3349132.

[31] R. Yue, G. Jiang, X. Jin, Q. He, and P. Xie, "Spatio-Temporal Feature Alignment Transfer Learning for Cross-Turbine Blade Icing Detection of Wind Turbines," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–17, 2024. DOI: 10.1109/TIM.2024.3350147.

[32] X. Liu, H. Ma, and Y. Liu, "A Novel Transfer Learning Method Based on Conditional Variational Generative Adversarial Networks for Fault Diagnosis of Wind Turbine Gearboxes under Variable Working Conditions," *Sustainability*, vol. 14, no. 9, p. 5441, Apr. 2022. DOI: 10.3390/su14095441.

[33] J. Chatterjee, M. T. Alvela Nieto, H. Gelbhardt, *et al.*, "Domain-invariant icing detection on wind turbine rotor blades with generative artificial intelligence for deep transfer learning," *Environmental Data Science*, vol. 2, e12, 2023. DOI: 10.1017/eds.2023.9.

[34] F. Luleci, F. Necati Catbas, and O. Avci, "CycleGAN for undamaged-to-damaged domain translation for structural health monitoring and damage detection," *Mechanical Systems and Signal Processing*, vol. 197, p. 110 370, Aug. 2023. DOI: 10.1016/j.ymssp.2023.110370.

[35] A. Almahairi, S. Rajeshwar, A. Sordoni, P. Bachman, and A. Courville, "Augmented CycleGAN: Learning many-to-many mappings from unpaired data," in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, Oct. 2018, pp. 195–204. [Online]. Available: https://proceedings.mlr.press/v80/almahairi18a.html.

[36] C. McKinnon, K. Tartt, J. Carroll, A. McDonald, C. Plumley, and D. Ferguson, "Comparison of novel SCADA Data Cleaning Technique for Wind Turbine Electric Pitch System," *Journal of Physics: Conference Series*, vol. 2151, no. 1, p. 012 005, Jan. 2022. DOI: 10.1088/1742-6596/2151/1/012005.

[37] P. Li, Y. Pei, and J. Li, "A comprehensive survey on design and application of autoencoder in deep learning," *Applied Soft Computing*, vol. 138, p. 110 176, May 2023. DOI: 10.1016/j.asoc.2023.110176.

[38] S. Jonas, D. Anagnostos, B. Brodbeck, and A. Meyer, "Vibration Fault Detection in Wind Turbines Based on Normal Behaviour Models without Feature Engineering," *Energies*, vol. 16, no. 4, p. 1760, Feb. 2023. DOI: 10.3390/en16041760.

[39] C. M. Roelofs, M.-A. Lutz, S. Faulstich, and S. Vogt, "Autoencoder-based anomaly root cause analysis for wind turbines," *Energy and AI*, vol. 4, p. 100 065, Jun. 2021. DOI: 10.1016/j.egyai.2021.100065.

[40] N. Renström, P. Bangalore, and E. Highcock, "System-wide anomaly detection in wind turbines using deep autoencoders," *Renewable Energy*, vol. 157, pp. 647–659, Sep. 2020. DOI: 10.1016/j.renene.2020.04.148.

[41] N. C. Schwertman, M. A. Owens, and R. Adnan, "A simple more general boxplot method for identifying outliers," *Computational Statistics & Data Analysis*, vol. 47, no. 1, pp. 165–174, Aug. 2004. DOI: 10.1016/j.csda.2003.10.012.

[42] J. Su, *GAN-QP: A Novel GAN Framework without Gradient Vanishing and Lipschitz Constraint*, 2018. DOI: 10.48550/ARXIV.1811.07296.

[43] Z. Li, M. Usman, R. Tao, *et al.*, "A Systematic Survey of Regularization and Normalization in GANs," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–37, Nov. 30, 2023. DOI: 10.1145/3569928.

[44] L. Liu, H. Jiang, P. He, *et al.*, "On the variance of the adaptive learning rate and beyond," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=rkgz2aEKDr.

[45] D. Misra, *Mish: A Self Regularized Non-Monotonic Activation Function*, 2020. DOI: 10.48550/arXiv.1908.08681.

[46] Y. Wu and K. He, "Group normalization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Sep. 2018.

[47] S. Bai, J. Z. Kolter, and V. Koltun, *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*, 2018. DOI: 10.48550/ARXIV.1803.01271.

[48] Y. Yazıcı, C.-S. Foo, S. Winkler, K.-H. Yap, G. Piliouras, and V. Chandrasekhar, "The unusual effectiveness of averaging in GAN training," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=SJgw_sRqFQ.

Appendix A: Autoencoder-based NBM

All our autoencoder-based NBMs follow the model specifications outlined in Table A1, based on a model architecture search on a randomly chosen WT. Each NBM was trained using the RAdam [44] optimizer with a learning rate of 0.003, minimizing the mean squared reconstruction error between model input and autoencoder reconstruction. The models were trained using a batch size of 128 SCADA samples. Training was stopped once the reconstruction error on the validation set stopped improved for several data scarcity-dependent epochs (250 for 2 weeks, 25 for full target data).

Table A1: Architecture of the autoencoder-based NBMs used in our study.

Autoencoder model architecture	
Input	*11 channels x 72 datapoints*
Block 1	Conv1d (32 filters, kernel size 7, stride 1, Mish [45]) x 2
	MaxPool (kernel size 2)
	GroupNorm [46] (1 group, 32 channels)
Block 2	Conv1d (32 filters, kernel size 5, stride 1, Mish) x 2
	MaxPool (kernel size 2)
	GroupNorm (1 group, 32 channels)
Bottleneck	Flatten, FC (32 x 18, 72), FC (72, 8 x 18)
Block 3	Upsample (factor 2)
	Conv1d (32 filters, kernel size 3, stride 1, Mish) x 2
	GroupNorm (1 group, 32 channels)
Block 4	Upsample (factor 2)
	Conv1d (32 filters, kernel size 3, stride 1, Mish) x 2
	GroupNorm (1 group, 32 channels)
Block Out	Conv1d (11 filters, kernel size 1, stride 1, linear)
Output	*11 channels x 72 datapoints*

Appendix B: Domain Mapping Network

The used architecture for the domain mapping discriminators is outlined in Table A2. For the generators, we used a residual temporal convolutional network (TCN) approach [47]. Our non-causal architecture consists of several residual TCN blocks, kernel size, dilation, and normalization, depicted in Figure A1. The full generator is outlined in Table A3.

Table A2: Description of our discriminator model architecture.

Discriminator architecture	
Input	*11 channels x 72 datapoints*
Block 1	Conv1d (128 filters, kernel size 5, stride 2, Mish)
	GroupNorm (1 group, 128 channels)
Block 2	Conv1d (128 filters, kernel size 3, stride 2, Mish)
	GroupNorm (1 group, 128 channels)
Block 3	Conv1d (256 filters, kernel size 3, stride 2, Mish)
	GroupNorm (1 group, 256 channels)
	Flatten
MLP	FC (9 * 256, 1, linear)
Output	*1 value*

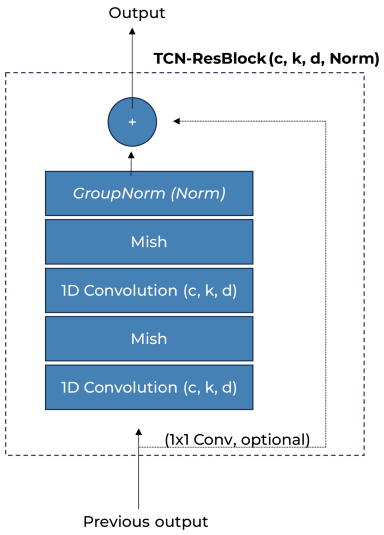


Figure A1: Structure of our TCN residual block, performing non-causal 1D convolutions with kernel size k, dilation d, and with c channels. The final group normalization is optional when *Norm* is set. Based on [47].

Table A3: Description of our generator architecture using TCN-ResBlocks.

Generator Architecture
Input: 11 channels x 72 datapoints
TCN-ResBlock(64, 3, 1, False)
TCN-ResBlock(64, 3, 2, True)
TCN-ResBlock(64, 3, 4, True)
TCN-ResBlock(64, 3, 8, True)
TCN-ResBlock(32, 3, 16, True)
TCN-ResBlock(16, 3, 32, False)
1D Convolution (c = 11, k=1, d=1, stride=1, bias=True, linear)
Output: 11 channels x 72 datapoints

GAN-QP. We follow the GAN-QP formulation to train our generators and discriminators. Let G be a generator, T a discriminator, x_r a real sample and x_f a fake sample (in our case, a mapped sample). The GAN-QP loss is defined as [42]:

$$\begin{aligned} T &= \arg\max_{T} \mathbb{E}_{(x_r,x_f)\sim p(x_r)q(x_f)} \left[T(x_r) - T(x_f) - \frac{(T(x_r) - T(x_f))^2}{2\lambda_{QP} d(x_r, x_f)} \right] \\ G &= \arg\min_{G} \mathbb{E}_{(x_r,x_f)\sim p(x_r)q(x_f)} \left[T(x_r) - T(x_f) \right] \end{aligned} \tag{5}$$

where in our study we use the Euclidean distance as d and set $\lambda_{QP} = 1$.

Training. Our domain mapping network was trained with a batch size of 128 using an Adam optimizer ($\beta_1 = 0.5, \beta_2 = 0.999$, learning rate = 0.0002) for the generators and discriminators each. Early stopping was implemented to stop training at an optimal state and to prevent overfitting. We use a reconstruction error obtained from the source NBM, by mapping the target validation data to the source domain. As only normal data is used for training and validation, we expect the reconstruction error of mapped target-to-source data to be low, representing realistic source WT data familiar to the source NBM. A rising reconstruction error could indicate overfitting on training data and unrealistic sample generation, as the mapped validation data starts to resemble the source domain less. It should be noted that this value cannot be used for model selection and tuning, as it contains no information regarding the content preservation (e.g., mapping all target samples to the same normal source domain sample would result in a very low score). Training was stopped after 1000 batch iterations with no improvement of the early stopping score. Moreover, at evaluation time, we used an Exponential Moving Average (EMA) of the training weights for the generators, which improved training stability and performance (as investigated in [48]).

Anomaly Augmentation. The domain mapping network is trained to translate normal SCADA data resulting in the model inherently learning only the mapping relationships of normal data. During test time, when anomalous samples can be introduced, it may therefore lead to unwanted behavior in the sense that the mapping might "repair" anomalies or cause inconsistent mappings. We therefore add artificially anomalous data to the training set by duplicating each sample from the training batches with a random modification, namely by setting a random part of the 12 hours (40 - 100%) and channels of a random feature group (e.g., power) to zero. Further investigations are needed to determine more augmentation techniques. While we found this step to be critical for performance, adding a further random scaling augmentation did not improve results.

Hyperparameters. A source-target WT domain pair was randomly selected to search for a model architecture and optimal hyperparameters. The target WT used for optimization (WT07) was subsequently excluded as possible target domain from the evaluation. We considered multiple candidates using a target domain scarcity of 1 month and evaluated the F1-score of the resulting threshold scores with ones from an NBM trained on the full representative target domain training data. The same architecture and hyperparameters were used for all other domain pairs, although it is questionable whether these remain optimal across different data scarcity degrees and domain shifts, i.e., different deviations between the source and target WT. Setting hyperparameters based on the domain distance is subject to future research.

The resulting full training procedure is described in Algorithm 1. For more detailed specifications we refer to our provided implementation.

Implementation. This work was implemented in PyTorch and trained using an NVIDIA GPU. Our code implementation is publicly available on GitHub `https://github.com/EnergyWeatherAI/WT_Generative_Domain_Adaptation`.

Algorithm 1 Domain mapping network training algorithm.

Require: Batch size m, generators G_{ST}, G_{TS} and discriminators $Disc_S, Disc_T$, loss weight hyperparameters $\lambda_{cyc}, \lambda_0, \lambda_R$. In our experiments we set $\lambda_{cyc} = 30, \lambda_0 = 0.5, \lambda_R = 0.1$.

while Training not interrupted by early stopping **do**
  for source domain batch b_s and target domain batch b_t **do**
    Generator updates
    Map batches to corresponding other domain: $b_{st} = G_{ST}(b_s), b_{ts} = G_{TS}(b_t)$
    Map batches back to original domain: $b_{sts} = G_{TS}(b_st), b_{tst} = G_{ST}(b_ts)$
    $\mathcal{L}_{GAN_{ST}} \leftarrow \mathcal{L}_{GAN_{QP}}(b_t, b_{st})$
    $\mathcal{L}_{GAN_{TS}} \leftarrow \mathcal{L}_{GAN_{QP}}(b_s, b_{ts})$
    $\mathcal{L}_{GAN} \leftarrow \mathcal{L}_{GAN_{ST}} + \mathcal{L}_{GAN_{TS}} + \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_0\mathcal{L}_0 + \lambda_R\mathcal{L}_R$
    Artificially corrupt batches b_t and b_s; calculate and add $\lambda_{cyc}\mathcal{L}_{cyc}$ to $\mathcal{L}_{GAN}$
    Update weights $\mathbf{w}_{ST}, \mathbf{w}_{TS}$ of G_{ST}, G_{TS} by descending: $\mathbf{w}_{TS}, \mathbf{w}_{ST} \leftarrow \text{Adam}(\nabla_{\mathbf{w}_{TS},\mathbf{w}_{ST}}\mathcal{L}_{GAN})$
    Discriminator updates
    Sample new batches b_s and b_t
    Map batches to corresponding other domain: $b_{st} = G_{ST}(b_s), b_{ts} = G_{TS}(b_t)$
    $\mathcal{L}_{Disc_S} \leftarrow \mathcal{L}_{Disc_{QP}}(b_s, b_{ts})$
    $\mathcal{L}_{Disc_T} \leftarrow \mathcal{L}_{Disc_{QP}}(b_t, b_{st})$
    $\mathcal{L}_{Disc} \leftarrow \mathcal{L}_{Disc_S} + \mathcal{L}_{Disc_T}$
    Update weights $\mathbf{w}_S, \mathbf{w}_T$ of $Disc_S, Disc_T$ by descending: $\mathbf{w}_S, \mathbf{w}_T \leftarrow \text{Adam}(\nabla_{\mathbf{w}_S,\mathbf{w}_T}\mathcal{L}_{Disc})$
  end for
end while

Appendix C: Detailed Results

Table A4: Detailed F1-scores (in %) for all 28 domain pairs

		2 months			6 weeks			1 month			3 weeks			2 weeks			1 week		
Target	**Source**	**Scarce**	**FT**	**Ours**	**Scarce**	**FT**	**Ours**	**Scarce**	**FT**	**Ours**	**Scarce**	**FT**	**Ours**	**Scarce**	**FT**	**Ours**	**Scarce**	**FT**	**Ours**
WT01	WT07	**89.0**	**89.0**	87.9	84.6	**88.5**	86.7	77.4	77.5	**86.1**	**74.4**	73.1	57.3	77.1	74.0	**79.4**	66.4	74.7	**78.8**
	WT02		56.8	64.4		77.4	**86.3**		72.2	**89.5**		70.2	**79.6**		72.9	**83.2**		72.3	**76.7**
	WT05		83.1	82.2		80.9	**88.3**		73.7	**84.0**		75.1	**87.9**		79.0	**83.6**		75.2	**75.5**
	WT04		85.1	**90.6**		80.6	**90.3**		78.3	**84.0**		71.5	**90.8**		72.7	**88.9**		66.6	**78.4**
WT02	WT01	**77.8**	**80.6**	50.0	62.6	**83.7**	64.6	65.3	**84.3**	56.3	60.3	**73.9**	56.6	**73.6**	32.7	42.5	35.2	1.9	**41.8**
	WT06		62.2	71.7		**80.9**	69.0		**84.0**	80.0		74.9	**77.1**		40.7	64.7		9.1	**62.1**
	WT05		**82.1**	63.5		**80.8**	64.3		**74.5**	65.2		**77.7**	65.5		**82.6**	61.2		**71.3**	61.4
	WT04		**84.7**	64.8		**83.9**	67.0		**72.7**	67.1		**78.6**	70.7		**80.7**	63.4		**66.0**	55.1
WT03	WT01	**84.7**	36.1	**86.5**	73.8	76.6	**86.7**	64.8	56.9	**84.8**	38.4	**76.8**	73.3	52.6	67.1	**73.5**	34.5	17.1	**40.7**
	WT06		43.2	79.1		78.4	**79.6**		67.0	**72.1**		**75.2**	40.5		62.8	**85.4**		17.8	**41.7**
	WT07		53.2	**86.8**		71.7	**85.6**		**79.3**	48.7		**84.1**	46.0		76.3	**89.3**		**54.8**	50.2
	WT02		53.3	78.4		78.9	**83.4**		**86.5**	48.7		73.7	**76.2**		66.5	**78.6**		41.0	**52.5**
	WT05		**88.8**	72.4		**89.9**	85.5		76.9	**82.4**		78.9	**85.6**		**88.7**	73.2		**90.8**	61.6
	WT04		81.7	77.8		**89.8**	79.3		**74.8**	65.9		81.7	**84.6**		**85.9**	70.0		**65.1**	48.7
WT04	WT01	**92.6**	79.8	75.4	**94.0**	70.9	78.8	**88.3**	78.8	85.0	**89.8**	67.4	82.3	42.8	42.3	**54.5**	34.4	36.8	**43.4**
	WT06		86.3	82.3		85.7	72.3		81.0	80.5		64.3	86.7		45.3	**49.7**		42.3	**47.0**
	WT07		88.6	84.3		92.3	84.9		**89.9**	82.5		83.3	78.9		60.4	**66.3**		57.3	**61.3**
	WT02		88.5	92.3		81.1	89.6		87.8	**91.5**		82.8	85.4		63.3	**69.9**		52.4	**61.5**
	WT05		91.7	**93.4**		91.8	93.5		92.8	**94.7**		92.1	**93.5**		**87.3**	61.1		**93.6**	63.4
WT05	WT01	**67.6**	65.1	61.9	55.3	**64.9**	51.6	**63.9**	53.8	62.4	**56.5**	46.4	45.2	31.5	26.0	**33.2**	24.3	**31.9**	19.5
	WT06		65.8	**78.3**		63.5	**76.6**		54.8	**70.7**		50.0	**80.8**		34.1	**78.7**		13.1	**47.4**
	WT07		**87.4**	75.7		**83.5**	76.1		68.8	**76.5**		69.0	**78.8**		**64.5**	47.9		**60.8**	52.1
	WT02		**87.4**	81.8		**86.5**	80.7		83.2	**86.1**		76.5	**88.9**		**71.8**	62.9		49.4	**53.1**
	WT04		73.7	**93.2**		65.5	**88.1**		70.7	**73.4**		68.2	**83.5**		**82.5**	79.5		51.2	**57.6**
WT06	WT07	**94.5**	67.3	66.2	42.4	59.1	**60.8**	24.7	40.8	**89.7**	31.1	36.3	**49.6**	30.3	38.0	**51.0**	**7.0**	6.4	**10.9**
	WT02		63.7	76.7		60.5	**73.6**		42.5	**74.5**		39.3	**47.0**		32.5	**40.1**		6.1	6.9
	WT05		74.9	**94.9**		42.9	**57.8**		20.7	**54.0**		18.3	**75.7**		19.5	**70.0**		6.2	5.7
	WT04		79.4	**94.9**		59.7	**76.5**		28.1	**72.3**		15.9	**75.8**		21.5	**78.9**		6.0	7.1

Appendix D: WT Overview

Table A5: Data specifications of the 7 WTs used in our work. The number of days refers to the range between the day of the first sample and the day of the last sample, therefore including days where no (valid) measurements were taken.

			Training & Validation Set (filtered, without scarcity)		Test Set (unfiltered, fixed)		
WT_ID	**Location**	**Rated Power [KW]**	**# days**	**# 12h-samples**	**# days**	**# 12h-samples**	**.. of which contain incidents**
WT01	Onshore	800	899	50574	387	48736	9049 (18.6%)
WT02	Onshore	3000	877	58403	387	45593	19630 (43.1%)
WT03	Onshore	2350	706	66407	323	43309	10102 (23.3%)
WT04	Onshore	2050	831	70191	355	35573	3275 (9.2%)
WT05	Onshore	2300	827	70922	354	37308	788 (2.1%)
WT06	Onshore	800	798	70197	341	21187	121 (0.6%)
WT07	Onshore	3050	830	66117	355	43107	7775 (18.0%)

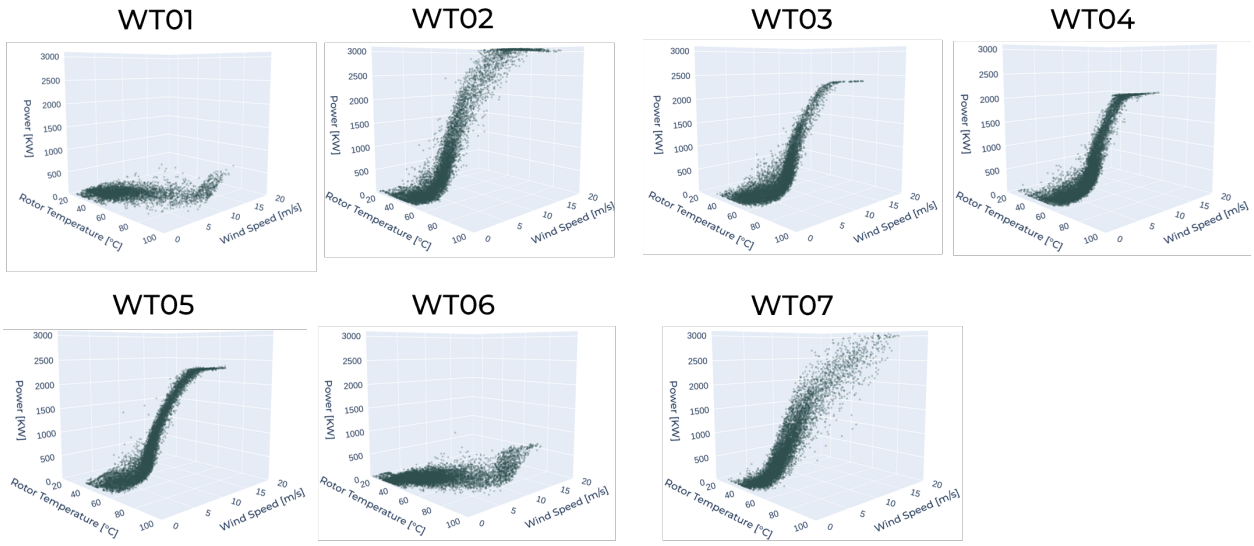


Figure A2: Scatter plots of the mean power, wind speed, and rotor temperature of a subset from filtered training samples illustrating the differences across all 7 WTs used in our experiments.

Appendix E: Representative Training Data

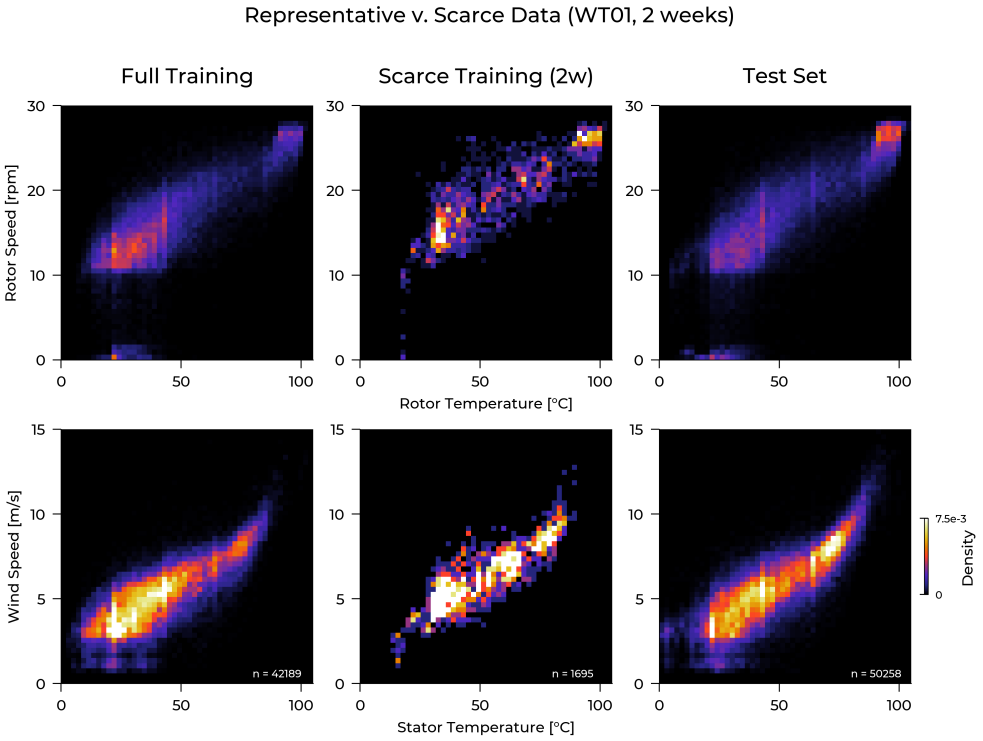


Figure A3: Density 2D-histograms with SCADA data of WT01. The top row shows the relationship between the rotor temperature and the rotor speed for the full training data (left column), for a scarcity scenario of 2 weeks of training data (middle column), and the filtered test set without incidents (right column). The bottom row visualizes the respective relationships between the wind speed and the stator temperature. We observe a visually very high similarity between the histograms of the full training data and the test set, i.e., the full training data appears to be *representative* of the WT's operational states. On the other hand, the histograms based on scarce training data show noticeable deviations to the test set data. For instance, a narrower range and less variability for low temperatures and low rotor and wind speeds.