Transfer Learning Approaches for Wind Turbine Fault Detection using Deep Learning

Jannik Zgraggen[1], Markus Ulmer[2], Eskil Jarlskog[3],Gianmarco Pizza[4], and Lilach Goren Huber[5]

[1,2,5] *Zurich University of Applied Sciences, Technikumstrasse 9, Winterthur, 8400 Switzerland*
jannik.zgraggen@zhaw.ch
markus.ulmer@zhaw.ch
lilach.gorenhuber@zhaw.ch

[3,4] *Nispera AG, Hornbachstrasse 50, CH-8008 Zurich, Switzerland*
eskil.jarlskog@nispera.com
gianmarco.pizza@nispera.com

ABSTRACT

Implementing machine learning and deep learning algorithms for wind turbine (WT) fault detection (FD) based on 10-minute SCADA data has become a relevant opportunity to reduce the operation and maintenance costs of wind farms. The development of practically implementable algorithms requires addressing the issue of their scalabililty to large wind farms. Two of the main challenges here are reducing the training times and enabling training with scarce or limited data. Both of these challenges can be addressed with the help of transfer learning (TL) methods, in which a base model is trained on a source WT and the learned knowledge is transferred to a target WT. In this paper we suggest three TL frameworks designed to transfer a semi-supervised FD task between turbines. As a base model we use a Convolutional Neural Network (CNN) which has been proven to perform well on the single turbine FD task. We test the three TL frameworks for transfer between WTs from the same farm and from different farms. We conclude that for the purpose of scaling up training for large farms, a simple TL based on linear regression transformation of the target predictions is an attractive high performance solution. For the challenging task of cross-farm TL based on scarce target data we show that a TL framework using combined linear regression and error-correction CNN outperforms the other methods. We demonstrate a scheme that enables the evaluation of different TL frameworks for FD without the need for labeled faults.

1. INTRODUCTION

Early fault detection (FD) in wind turbines (WT) is a first step towards implementing predictive maintenance for operational farms. In recent years there is an increasing recognition of the importance of FD methods based exclusively on 10-minute Supervisory Control and Data Acquisition (SCADA) data which is stored conventionally for all wind farms (Tautz-Weinert & Watson, 2016). Models based on this low resolution operational data are forced to rely on information from healthy functioning WTs only, rather than on labeled historical faults. The reason is that such faults are rare and each one of them is unique in character. The hope of deploying reliable classification methods for FD based on 10-minute SCADA data is therefore unrealistic. On the other hand various normal state methods have been developed and demonstrated the ability to detect faults by training the models using healthy data and detecting deviations from normality in the test data (referred to as "semi-supervised" models).

FD in multivariate time series data based on semi-supervised normal state modeling can be achieved using various machine learning (ML) techniques. Common approaches are based on clustering (Lapira et al., 2012), dimension reduction (Michau & Fink, 2021), reconstruction (Jiang et al., 2017) and regression (Zaher et al., 2009; Schlechtingen & Santos, 2014). The latter used various approaches for WT FD, including neural network models based on measured variables from the SCADA system. While many of these have been proven effective for the anomaly detection tasks, regression methods have a clear advantage when it comes to the possibility to localize the origin of the fault within the machine; whenever a large prediction error (PE) is detected in a certain regression target, it is assumed that this variable is related to the fault root-cause. This identification is not as straightforward and can be rather complex using other FD approaches.

In a previous paper (Ulmer, Jarlskog, Pizza, Manninen, & Goren Huber, 2020) we showed the advantages in training a Convolutional Neural Network (CNN) for regression-based

FD using WT SCADA data. The enhanced accuracy and robustness of this model was demonstrated. Moreover, we showed that this model can be easily extended into a multi-output version that allows simultaneous FD on multiple turbine components with the same accuracy and training time as the single output CNN. With this method, the common modeling approach of developing a separate model for each monitored turbine variable (Schlechtingen & Santos, 2014) can be spared and the number of trained models can be cut down considerably.

Training CNNs for FD requires enough representative data from healthy time periods and can get computationally expensive when scaled up to large commercial wind farms (WFs). Moreover, for new installations or replaced components historical data is often missing such that training a high performance CNN can be challenging. Similarly to other applications, within the field of Prognostics and Health Management (PHM) and beyond it, scarcity of training data can be overcome with the help of transfer learning (TL). The models are trained on a source unit with enough representative data and the knowledge of the trained model is transferred to other target units, possibly operated under different conditions, or suffering from limited training data.

TL has been applied in the past for WT data. Similarly to other PHM applications (see Zheng et al., 2019; Moradi & Groth, 2020 and references therein), most of the works focus on TL for classification tasks for fault diagnosis (Li et al., 2021; Chatterjee & Dethlefs, 2020; Yun et al., 2019; Zhang et al., 2018; Guo et al., 2020; Chen et al., 2021). However, classification methods for WT FD using 10-minute SCADA data are hard to implement practically. To the best of our knowledge there was no attempt to develop TL frameworks for FD tasks on WT which are based on healthy data alone with no fault labels available, both for the source turbine and for the target turbine. Moreover, the application of standard TL methods to time series data in general has been rather limited to classification or forecasting (Fawaz et al., 2018; Ye & Dai, 2021), including the use of TL with deep networks for wind farm short term power forecasting, as in Hu et al., 2016; Qureshi et al., 2017; Wang et al., 2020, and classification based anomaly detection (Vercruyssen et al., 2017).

A past application of TL for semi-supervised (i.e based on healthy data only) FD of WTs is not known to us. Moreover, semi-supervised TL for anomaly detection in any other PHM application except WTs has been demonstrated up to now in one paper (Michau & Fink, 2021). The anomaly detection task was solved using dimensional reduction for feature extraction followed by a one class classifier. The domain adaptation TL method presented there cannot be effectively applied to regression based FD in which the output time-series is most strongly affected by the faults, whereas domain shift is expected both for the input and for the output.

In this paper we suggest TL approaches to transfer a FD task based on a semi-supervised training (only healthy labels are available) of a regression CNN with 10-minute WT SCADA data as its input. One of the TL frameworks we test is fine-tuning, a well established method for classification task TL in various application fields. A model is trained on a large labeled source data set and then fine tuned partially or fully using the (small) available target data set, with possibly modified or missing classes. Fine tuning TL has been proven to outperform the alternative of training from scratch with the small data set for various applications, including for wind power forecasts (Qureshi et al., 2017). However, the effectiveness of fine tuning TL has not been demonstrated for semi-supervised anomaly detection tasks on multivariate time series data, and in particular not using deep CNNs for regression.

A key feature of our problem is that in case of scarce data from the target turbine, this data may be from a specific season and thus not representative for operating conditions outside the training set. In case the seasonal effects are strong, conventional fine tuning TL may fail to extrapolate into the test data domain.

To address this difficulty we suggest two additional TL frameworks. We compare the performance of all three TL frameworks and analyze the advantages and potential use-cases of each framework for the case of WT FD. All three frameworks are designed as extensions to a base CNN model for normal state modeling of various WT monitoring variables, such as component temperatures. The base model is pre-trained using one year of 10-minute resolution SCADA data from a source turbine. Next, a TL framework is applied in order to obtain predictions and use them for FD on a target turbine, either from the same wind farm or from a different farm.

The contributions of this paper are the following:

- We address the problem of TL for FD in WTs using only the readily available 10-minute SCADA data.
- We suggest simple approaches for TL for semi-supervised regression-based anomaly detection tasks rather than classification tasks that were previously addressed for wind turbine FD.
- We test the various TL frameworks for both within-farm and across-farm transfer and elucidate the practical benefit of TL in each case.
- We suggest new frameworks to quantify and compare the performance of TL methods for unlabeled data, a common situation for FD tasks in most industrial applications.
- We tackle the problem of transfer between units in the presence of seasonal domain shift, and deal with the challenge of limited and in particular season-specific target data.

2. TL FOR WIND FARM FAULT DETECTION

Implementing large scale FD for wind farms requires going beyond the single turbine modeling approach. Training a separate model for each turbine is not always feasible out of two reasons. The main reason is the lack of sufficient historical data for training ML models for some of the turbines. This could be because certain turbines are newly installed, and the SCADA system accumulated only little data. Alternatively, in some cases, all or some of the historical data is not informative for future predictions because of a recent component replacement or re-calibration. The second motivation to develop TL algorithms for FD of WTs is the prospect of training ML algorithms on one turbine instead of the entire farm (sometimes containing hundreds of turbines). This would amount to upscaling of the algorithm deployment to be at least an order of magnitude faster, offering an attractive reduction of the implementation costs.

Motivated by these two reasons we develop approaches of TL between WTs. These methods are aimed at training a base ML model on a source turbine and transferring the learned knowledge when using the model to predict on a different turbine.

The paper addresses the two scenarios separately. In Section 4.1 we focus on cross-turbine TL within a single wind farm. In this case the main objective of TL is to scale up training by transferring the FD task from one source turbine to all the rest in the farm. We evaluate the TL goodness of two different frameworks by comparing the FD performance to a baseline model trained on the target turbine with enough representative data.

A second scenario is demonstrated in Section 4.2. There we focus on TL across different wind farms, assuming a source turbine in an older farm, thus with abundant training data and a target turbine in a new farm, with only several months of data. The transfer in this case is particularly challenging, because the target data set is not only small in size but often also not representing all operating conditions, but only those of a single season. An additional challenge is a domain shift both in the input and in the output space between the source and the target.

2.1. Problem Definition

In this paper we address the problem of TL of a semi-supervised regression-based anomaly detection (or FD) task from the domain of a source turbine $\mathcal{D}_S = \{X_S, Y_S\}$ to a target turbine domain $\mathcal{D}_T = \{X_T, Y_T\}$. The real valued output variable $y_t \in Y$ (for example, the generator bearing temperature) is regressed on a multivariate time series input (in our case these are the power, wind speed, rotor speed and ambient temperature). The common situation in practice is that healthy training data of both inputs and output $\mathcal{D}^{(train)} = \{X^{(train)}, Y^{(train)}\}$ is abundant for the source turbine and is rather limited for the target turbine. The FD task is thus semi-supervised, because in both domains only healthy data is assumed to be available, with no fault labels at all.

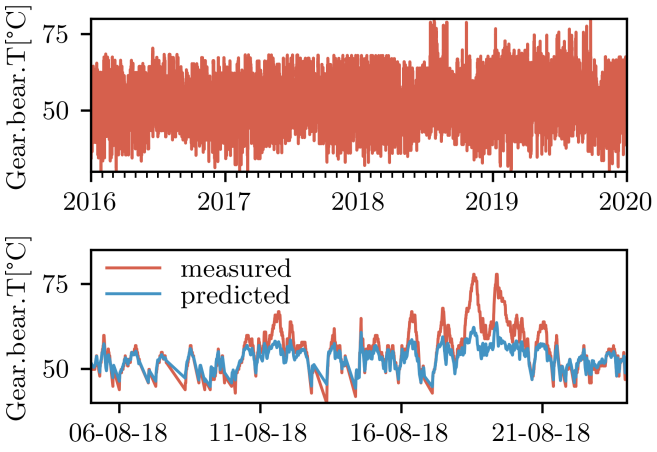

Figure 1. Raw gearbox bearing temperature of the turbine T_0. Lower panel: zoom in of the measured and predicted values when training the base CNN model with one year of data from this turbine (baseline training scheme).

The TL task can thus be formulated as learning a regression model $f_T(\cdot)$ in the target domain D_T by training a model $f_S(\cdot)$ in the source domain $\mathcal{D}_S^{(train)} = \{X_S^{(train)}, Y_S^{(train)}\}$ and exploiting the target domain data which we denote as $\mathcal{D}_T^{(tune)} = \{X_T^{(tune)}, Y_T^{(tune)}\}$ in order to tune or adapt the trained source model with the goal to achieve a high FD performance on the unseen test data in the target domain, $\mathcal{D}_T^{(test)} = \{X_T^{(test)}, Y_T^{(test)}\}$.

Our pre-trained source model $f_S(\cdot)$ which serves as the base model for TL is based on a CNN that has been previously developed and optimized for single-turbine FD (Ulmer, Jarlskog, Pizza, Manninen, & Goren Huber, 2020). In the following section we describe the base model and the TL approaches we tested on various examples of wind turbine FD.

3. METHODS

3.1. Base Model Description

Our base single-turbine fault detection pipeline includes as a first step a CNN for either single- or multi-target regression. In a single target setup, the target variable y_t is typically a temperature of a certain turbine component, e.g a generator bearing, the gearbox oil or the hub temperature at time t. The inputs are multivariate time series $X = \{x_i^j\}, i \in [t-m, t], j \in [1, N]$ where m is the size of the look-back window and $N = 4$ is the number of input variables. These are measured variables which were shown to serve as effective predictors independent of the fault type: output power, ambient temperature, wind speed and rotor speed. The base CNN model is trained with healthy data of a single turbine to minimize the mean squared error $\mathcal{L} = |y_t - \hat{y}_t|^2$ between the

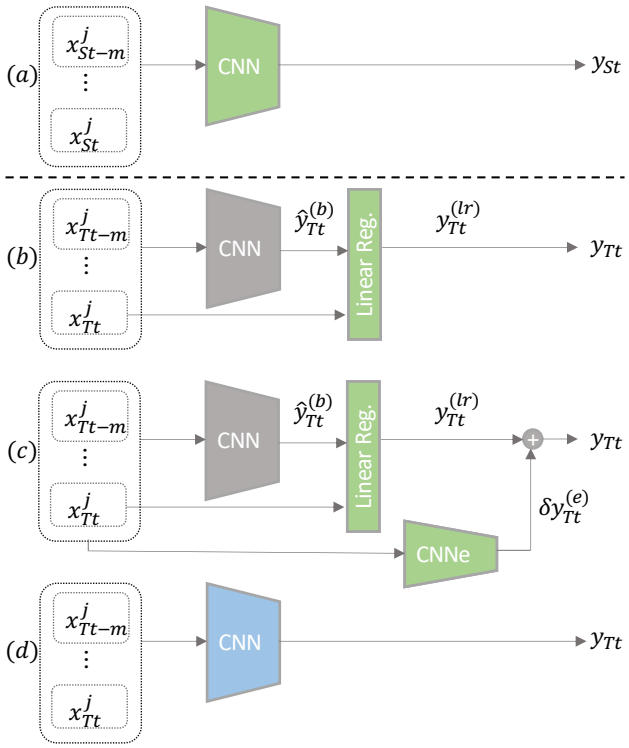

Figure 2. Training schemes for TL of CNN- based fault detection. (a) Pre-training of the base model with data from the source turbine. (b)-(d) the 3 TL schemes suggested in the paper for tuning the model to predict on target turbine data: (b) LRT (c) LRCNNT (d) Fine Tuning. Training from scratch, fine-tuning of pre-trained weights and no training (fixed weights) are colored green, blue and gray respectively.

predicted $\hat{y}_t$ and the measured y_t target variable. In a multi-output CNN configuration, the sum over the errors of all targets is minimized, such that the CNN is trained to predict a set of turbine variables, commonly temperatures of various components. The details of our architecture are described in (Ulmer, Jarlskog, Pizza, Manninen, & Goren Huber, 2020).

The base model is trained using enough representative 10-minute SCADA data. This is typically data from a full year, representing all potential seasonal variations, and including around $40,000$ data points. To test the base model we feed it with unseen data and measure the prediction errors (PEs). We expect large PEs whenever one of the target variables deviates from normal behavior. An example of the predicted and measured values of the gearbox bearing temperature is shown in Figure 1.

We note that for the sake of the analysis of TL frameworks we compare the different TL algorithms with a base model for a single output variable. The extension to TL for the multi-output CNN model is technically straightforward and its analysis will be discussed in a separate paper.

The next steps of the FD algorithm amounts to calculating the residuals (PEs) $r_t = y_t - \hat{y}_t$ and assigning a health index (HI) to each PE. To calculate the HI (also known as the anomaly score) we perform a Kernel Density Estimate of the PE distribution of our healthy training set and estimate it as a Gaussian probability distribution function (pdf) with mean μ and variance σ^2. We then assign to each new PE the following HI h_t:

$$h_t = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{r_t} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx \tag{1}$$

To detect faulty measurements we set a threshold at a desired significance level α and declare a point as faulty if the probability to obtain its PE (or higher) assuming a healthy state is smaller than α, or equivalently if the HI satisfies $h_t \geq 1 - \alpha$.

The post processing steps were optimized for FD tasks with the base (single turbine) model and comprise of a low-power filter and a moving median calculation of the PEs and their HIs.

3.2. TL Frameworks

We discuss three different TL frameworks, as depicted in Figure 2. The first step of all three frameworks is pre-training the base CNN model $f^{(b)}(\cdot)$ on a data set $\mathcal{D}_S^{(train)}$ from the source turbine to predict the output variable $y_{St}^{(b)}$,

$$y_{St}^{(b)} = f^{(b)}\left(\{x_{Si}^j\}_{i\in[t-m,t],j\in[1,4]}; \Theta_b\right). \tag{2}$$

The trained network is then used differently in each framework in order to achieve accurate predictions for the target turbine. Below we describe the details of each of the three TL frameworks. In the results section we present two different use cases and test the performance of the TL frameworks in each case. We then discuss the advantages and disadvantages of each approach for the different use cases.

3.2.1. Linear Regression Tuning

In a previous paper we introduced a cross-turbine training scheme (Ulmer, Jarlskog, Pizza, & Goren Huber, 2020). In this scheme we first use the pre-trained CNN to predict on the target turbine input data X_T,

$$\hat{y}_{Tt}^{(b)} = f^{(b)}\left(\{x_{Ti}^j\}_{i\in[t-m,t],j\in[1,4]}; \Theta_{bS}^*\right). \tag{3}$$

The optimal parameter set Θ_{bS}^* is determined using the source data. The next step is aimed at transferring the regression model to the target domain. To this end we use the target tuning data set $\mathcal{D}_T^{(tune)}$ to tune the predictions $\hat{y}_{Tt}^{(b)}$ with a linear regression model $f^{(lr)}(\cdot)$,

$$y_{Tt}^{(lr)} = f^{(lr)}\left(\{x_{Tt}^j\}, \hat{y}_{Tt}^{(b)}; \Theta_{lr}\right). \tag{4}$$

Note that unlike the CNN, the linear regression model is time-local, that is, uses only the inputs $\{x_{Ti}^j\}, j \in [1,4]$ at time $i = t$. The resulting corrected predictions $\hat{y}_{Tt}^{(lr)}$ are used for the calculation of the PEs of the Linear Regression Tuning

(LRT) framework,

$$r_t^{(LRT)} = y_{Tt} - \hat{y}_{Tt}^{(lr)} \tag{5}$$

The HIs for all future predictions on the target turbine are computed according to Eqn. 1 using the estimated pdf of the PEs based on the tuning data $\mathcal{D}_T^{(tune)}$.

Despite its simplicity and its low computational load on top of the source model training, the LRT framework was shown in (Ulmer, Jarlskog, Pizza, & Goren Huber, 2020) to be effective for TL in various cases. We demonstrated the ability of this approach to transfer the CNN-based FD task into a target domain with scarce data of a target turbine inside and outside the farm. In particular, even for a commonly encountered case in which the target turbine data is of a single season, the TL method performed well on test data from the other seasons which is clearly out of the training distribution. We are thus encouraged to test the LRT approach on various turbines and compare its performance to other methods.

3.2.2. Linear Regression+CNN for Error Component Tuning

As we show below, the ability of the LRT framework to transfer the FD task between turbines is quite good. However, under more severe domain shifts, such as the case of transfer between different farms, there is still potential to improve on the FD performance of the LRT. To this end, we introduce an additional tuning step subsequent to the linear regression part. This step aims at modeling the remaining error component which is not transferred well enough using the linear time-independent transformation of the LRT,

$$\delta y_{Tt}^{(e)} \equiv y_{Tt} - \hat{y}_{Tt}^{(lr)}. \tag{6}$$

To this end we train an additional CNN, denoted by CNNe, to predict $\delta y_{Tt}^{(e)}$ using the tuning set time series inputs from the target turbine,

$$\delta y_{Tt}^{(e)} = f^{(e)}\left(\{x_{Ti}^j\}_{i\in[t-m,t],j\in[1,4]};\Theta_e\right). \tag{7}$$

In the Linear Regression followed by CNNe Tuning (LRCNNT) framework, the final estimate of the target variable y_{Tt} is given by the sum of the LRT prediction and the CNNe error prediction,

$$\hat{y}_{Tt}^{(e)} = \hat{y}_{Tt}^{(lr)} + \hat{\delta y}_{Tt}^{(e)}. \tag{8}$$

The resulting PEs,

$$r_t^{(LRCNNT)} = y_{Tt} - \hat{y}_{Tt}^{(e)}, \tag{9}$$

are used for the calculation of the HI in this case, using the tuning set from the target turbine to estimate a reference pdf.

Note that training a CNNe to learn the residuals is not equivalent to training a CNN from scratch, nor to skipping the LRT step altogether. By including both the LRT and the CNNe, and training them subsequently and not simultaneously, we make sure that each step focuses on learning a different part of the transfer: the LRT corrects for the linear domain shifts whereas the CNNe corrects for more complex, non linear and time dependent shifts both in the data distributions and in the functional behavior of the two turbines.

3.2.3. Fine Tuning

Fine tuning is a common method for TL which has been demonstrated for various classification and forecasting tasks. Here we apply it for a regression-based anomaly detection task with a multivariate time series input. To this end we pretrain the base CNN model on the source training data (Eqn. 2) and subsequently fine-tune the entire CNN with a reduced learning rate, using the tuning data set $\mathcal{D}_T^{(tune)}$ from the target turbine,

$$y_{Tt}^{(ft)} = f^{(b)}\left(\{x_{Ti}^j\}_{i\in[t-m,t],j\in[1,4]};\Theta_{ft}\right). \tag{10}$$

In this case we focus on full fine tuning rather than partial fine tuning, which clearly showed worse TL performance.

The HIs for the target turbine are calculated again using the estimated pdf of the PEs of the tuning data set, $r_t^{(FineTune)} = y_{Tt} - \hat{y}_{Tt}^{(ft)}$.

3.3. Model Evaluation

Developing FD algorithms for a specific application using only field data can be a challenging task. One of the main difficulties lies in model evaluation. As in most practical applications, also here we lack true labels for almost all turbines, with very few exceptions of annotated faults. To circumvent the lack of true labels we suggest an evaluation methodology for TL models which analyzes the performance in comparison to a fixed reference model. In this case the natural candidate for a reference model is the base model, trained with enough representative data $X_T^{(ref)}$ on the *target turbine*. This is possible in our evaluation experiments since we can select a target turbine with enough training data, emulating the limited data scenario by using only part of it for tuning the TL models.

To set the baseline reference we assign "true labels" (healthy or faulty) to each measurement based on its HI as calculated using the base model trained on $X_T^{(ref)}$. We label as "faulty" only measurements that are above the 95% ($\alpha = 0.05$) detection threshold of the base model. We then use these "true" labels to evaluate all models in terms of recall and precision scores. While the threshold choice of 95% is arbitrary, setting it allows us to measure the similarity of all models to a baseline defined by training the base CNN model on the target turbine with enough healthy data. Such a comparison makes sense if we refer to the baseline as an accurate and robust fault detector, a task that we would like to transfer as well as possible using the various TL frameworks. We note that for the

sake of clarity of the evaluation, we select an evaluation period which contains both healthy and faulty data (according to the baseline scores).

4. Results and Discussion

We test several approaches for TL on two different use cases, as described in Sec.2. The first one is TL for the purpose of increasing the computational efficiency of training the FD algorithms for a large number of WTs. The second use case is transferring the FD task to turbines with too little training data. We will demonstrate the first use case using source and target turbine from the same wind farm. The second use case will be tested on a source and a target out of two different farms.

4.1. Cross-Turbine TL Within One Wind Farm

A wind farm often contains tens or even hundreds of turbines, usually of the same manufacturer and often of the same model. The prospect of training FD models on one turbine and using the trained models to predict on all other turbines in the park, with no (or very little) additional computational effort, is very attractive for practical implementations. Since the ambient conditions are similar for all turbines in the same farm, one could expect a good TL quality between a source and a target of the same farm. In our case the typical domain shift between WTs in the same farm is in the output variables, such as component temperatures, that may differ due to different calibration or component age.

To evaluate the potential of TL for an existing wind farm we assume that all turbines would potentially have enough healthy data for training, that is in our case representative data of one full year. However, instead of training the CNN from scratch on the individual data set of each turbine we train it on data $\mathcal{D}_S$ from one of the turbines, the source turbine S_0. We then use the trained source model as a starting point for two different TL approaches to transfer the FD task to a target turbine T_0. The entire target data set is split to a "tuning set" $\mathcal{D}_T^{(tune)}$, in this case one year of healthy data used for training the TL part of the algorithm, and "test set" $\mathcal{D}_T^{(test)}$ the rest of the data, used for testing the entire TL framework. The TL approaches we evaluate are:

1. Linear Regression Tuning (LRT), see Section 3.2.1.
2. Fine Tuning (FineTune), see Section 3.2.3.

Figure 3 displays the comparison of the two TL approaches for the transfer from turbine S_0 to turbine T_0 within the same farm for the task of FD on the gearbox bearing temperature. Fig 3(a) shows the PEs of the base CNN model when trained (in a standard single-turbine scheme) with one year of healthy data $\mathcal{D}_T^{(ref)}$ of the target turbine T_0 as a reference. Panels (b) and (c) show the results for the LRT and FineTune transfer frameworks respectively with S_0 as source and T_0 as target. We note that in this case we skip the comparison with the LRCNNT framework since the results of the LRT are already satisfactory to the extent that we avoid the additional computational step of the CNNe on top of the LRT step. This makes sense from a practical point of view, since the main goal behind the use case we describe is to allow for computational upscaling of the training.

The colored area in each plot marks the time period used for training (in the baseline case) or for tuning (in the TL frameworks). After assigning HIs we set the threshold for detection at $\alpha = 0.01$ for all three models. With this threshold, red colored PEs are detected as faulty, whereas blue ones are healthy. We observe only minor differences in the FD performance of both TL approaches and the baseline.

In order to quantify these differences and evaluate the performance of the different TL frameworks we use the baseline as reference as described above in Sec. 3.3. Figure 3(d) compares the performance of the models in terms of precision and recall scores using the true labels which were assigned in this way. The average precision (AP) scores, which correspond to the area under the curves, are given in the legend in brackets for each training method. The time period used for model evaluation ranges between June 2018 and September 2019, containing a high fraction of points that are labeled as faulty. Both TL frameworks perform quite similarly to the baseline, which was trained on one year data. In particular, there is only a slight advantage in performance to the FineTune approach (AP=0.76) over the LRT model (AP=0.74) whereas the latter requires close to zero additional computation on top of the source model training. Concretely, fine tuning allows for around 40% reduction of the training time compared to training from scratch. The LRT computation time is however negligible even compared to the data pre-processing time, and can offer a massive reduction of the computational time when deployed on a farm of tens or hundreds of turbines.

Since the goodness of transfer of both schemes is similar, we select the one TL framework which is more computationally efficient and demonstrate it on several other turbines from the same farm. The PEs and HIs are then compared with the baseline for each target turbine, achieved by training the base CNN model on $\mathcal{D}_T^{(tune)}$.

Figure 4 displays the transferred PEs for 5 different target turbines trained with the same source S_0 in the left column. The period used for tuning the LRT is colored blue. The TL results are contrasted with the result of the respective base model using the same data as an input for training or tuning, shown in the second column. The training period is marked in green. Note that data availability differs between the turbines, therefore some of the predictions start later than others. For the sake of the discussion we selected the most appropriate year for training the models for each turbine, and we plot the PEs also for time periods prior to the training period (differently from operational deployment).

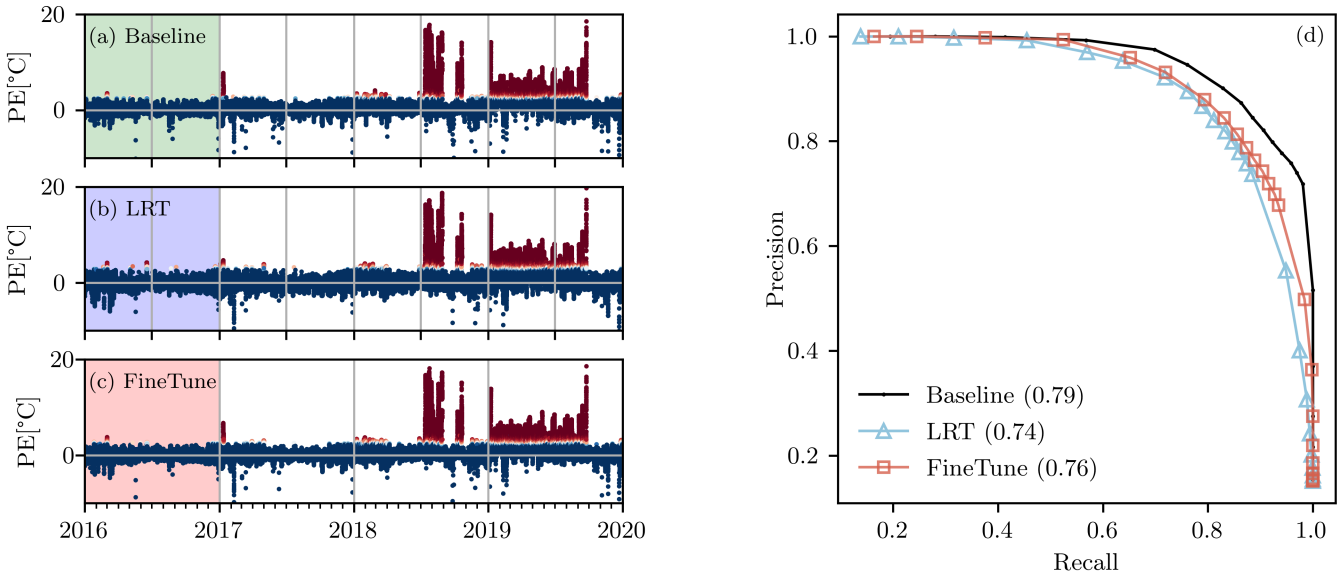

Figure 3. Comparison of approaches for TL between turbines within one farm. Prediction errors (PEs) are plotted vs. time for three training schemes: (a) baseline training of the base CNN model from scratch on target turbine data (b) LRT framework for TL (c) FineTune framework for TL. Both TL frameworks are between turbines from the same wind farm. PEs are colored red when the fault detection threshold using significance level $\alpha = 0.01$ is exceeded. The training or tuning period (1 year) is marked with colorful background. panel (d): Precision-Recall curves for the 3 training schemes of panels (a)-(c). Average precision values (area under curve) are stated on the legend in brackets.

The two right columns of Figure 4 show the PE distributions for the training (or tuning) set, contrasted with the same quantity for a test set of one year outside the training set. The test set was selected to be as healthy as possible, similarly to the training set. The distributions are contrasted for the LRT framework (third column) and the base model (fourth column). The purpose of this visualization of the results is to stress the importance of looking at error distributions, since they are used for the HI calculation and the threshold setting for the entire data. If there is a significant distribution shift between the healthy training and test PEs, we expect either missed detections or false positives to be frequent. The main result we would like to stress here is the similarity of the distribution plots between the TL and the base model. For most target turbines the shift between train and test PE distributions is small. In all cases we note that this shift is comparable in the TL and the base model, such that the FD performance of the TL framework is very similar to the baseline, explaining the similarity in the time-dependent HIs. We note that the analysis of the PE distributions allows for an evaluation of TL methods for semi-supervised FD even in the absence of labeled validation data.

For the FD task we observe almost no difference between training the base model and TL from the source turbine using the LRT framework. The difference in run times is however significant and scales linearly with the number of turbines in the farm. We conclude that for turbines within the same wind farm, the LRT TL framework provides a high quality alternative to training from scratch on the individual turbines for regression based FD.

4.2. Cross-Farm TL

In contrast to wind turbines within the same farm, turbines from different farms may have a much stronger domain shift, leading to a more challenging transfer of the FD task. In particular, the domain shift in this case is no longer only in the output variable but both in the inputs and the outputs. This can be due to different ambient conditions (such as temperature, wind speed and direction and their dynamic behavior over time) between the source and the target turbine in addition to differences in the operational modes of the turbines. We thus expect the task of transferring knowledge between wind farms for the purpose of early FD to be more complex than within the same farm. This includes also the challenge in selecting a good source for a given target, an important question that goes beyond the scope of the present paper.

The cross-farm TL is particularly useful in case of newly installed wind farms, with only very little data for FD model training. In this case the source turbine is older and has abundant healthy historical data that can be used to train the base CNN model. Similarly to the previous section we pre-train the base model on a source turbine using a one year training set $\mathcal{D}_S^{(train)}$. This time, however, the source turbine S_1

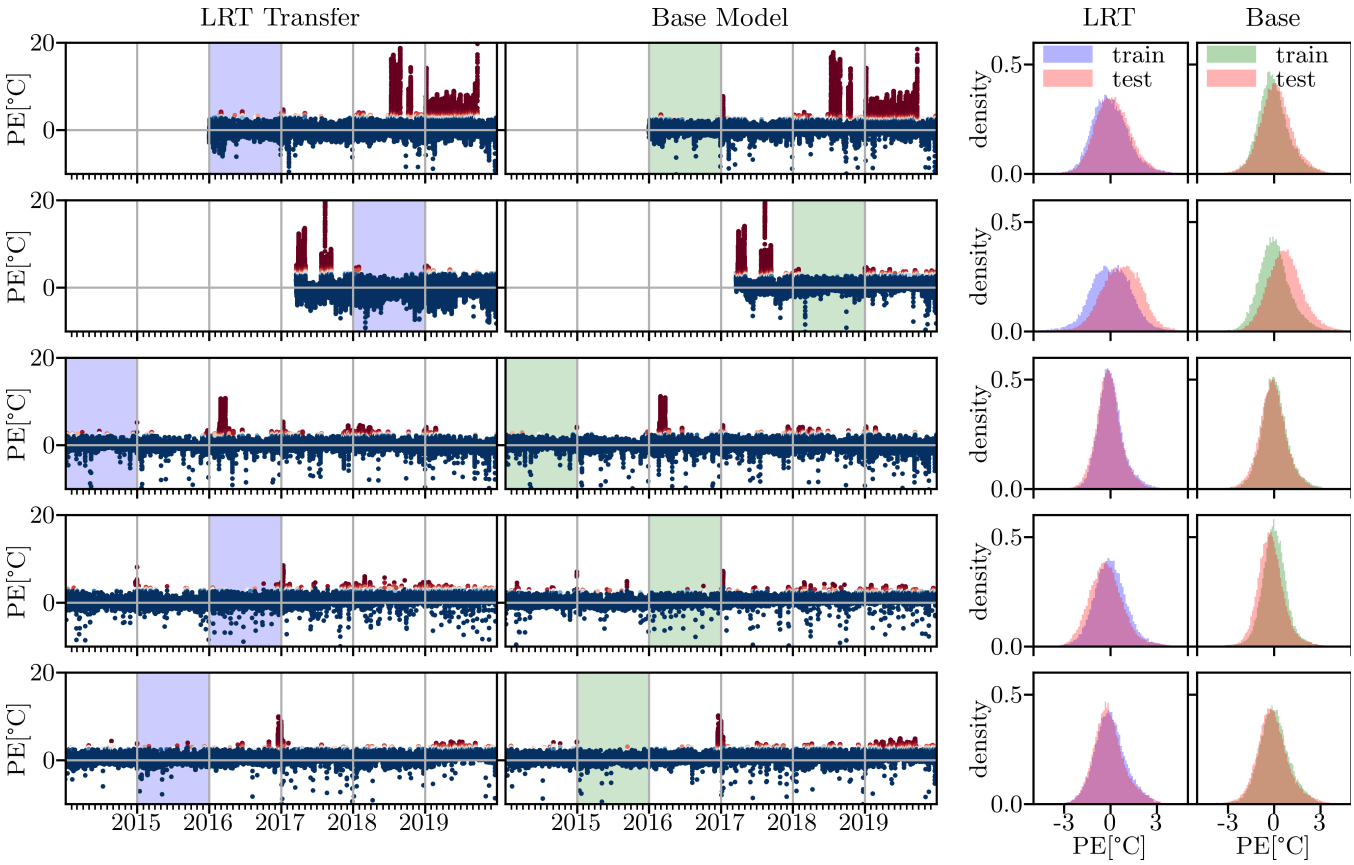

Figure 4. Scaling up FD training using TL within the farm. Each row displays the results of TL to another target turbine, all with the same source turbine. Two left columns: PEs for the LRT transfer framework, PEs using the base CNN model to train from scratch on the target turbine data. Two right columns: the corresponding PE distribution comparison between train and (healthy) test set for each training scheme. Color code: green for training from scratch, blue for TL, red for testing. The training or tuning periods are marked with the corresponding color in the left two columns.

is located in another wind farm. We then apply various TL methods to tune the model with the tuning set of the target turbine $\mathcal{D}_T^{(tune)}$ as input and generate predictions for the target turbine T_0. To emulate the data scarcity scenario we select only 3 months of winter data of the target turbine as the tuning set and use the rest of the data of this turbine $\mathcal{D}_T^{(test)}$ for testing the performance of the following TL frameworks:

1. Linear Regression Tuning (LRT), see Section 3.2.1.
2. Linear Regression + CNN Tuning (LRCNNT), see Section 3.2.2.
3. Fine Tuning (FineTune), see Section 3.2.3.

The results of the three TL frameworks are contrasted with those of the base model trained from scratch with a full year data from the target turbine, $\mathcal{D}_T^{(ref)}$, denoted by "baseline" training. In addition we compare the results to the "limited data" case by using only the tuning set $\mathcal{D}_T^{(tune)}$ to train the base model from scratch. Note that the same limited data set is used for training from scratch and for the three TL methods. This allows a comparison of using TL vs. training from scratch if we assume that only 3 months of healthy data are available from the target turbine, as would be the case if this turbine were newly installed.

The left column of Figure 5 displays the PEs colored according to their HI with a detection threshold of $\alpha = 0.01$ for all 5 training schemes; baseline, limited data and the three TL frameworks above. Measurements that would be detected as faulty with this threshold by each scheme are marked in red. Time periods used for training the base CNN model from scratch plotted on a green background, and periods used for tuning a TL algorithms using the limited target turbine data with a blue background. In addition we selected a 3 months healthy period outside the training and tuning set as a test set and marked it with a red background on all plots.

The right column of this figure shows a comparison of the PE distribution between the train (or tune) and the test set for each of the 5 training schemes. In all cases the training (green) or tuning (blue) set are the first 3 months of 2016. In the Baseline scheme these are the first 3 months of the training data and for the other 4 schemes it amounts to the entire tuning data, colored in the corresponding color in the

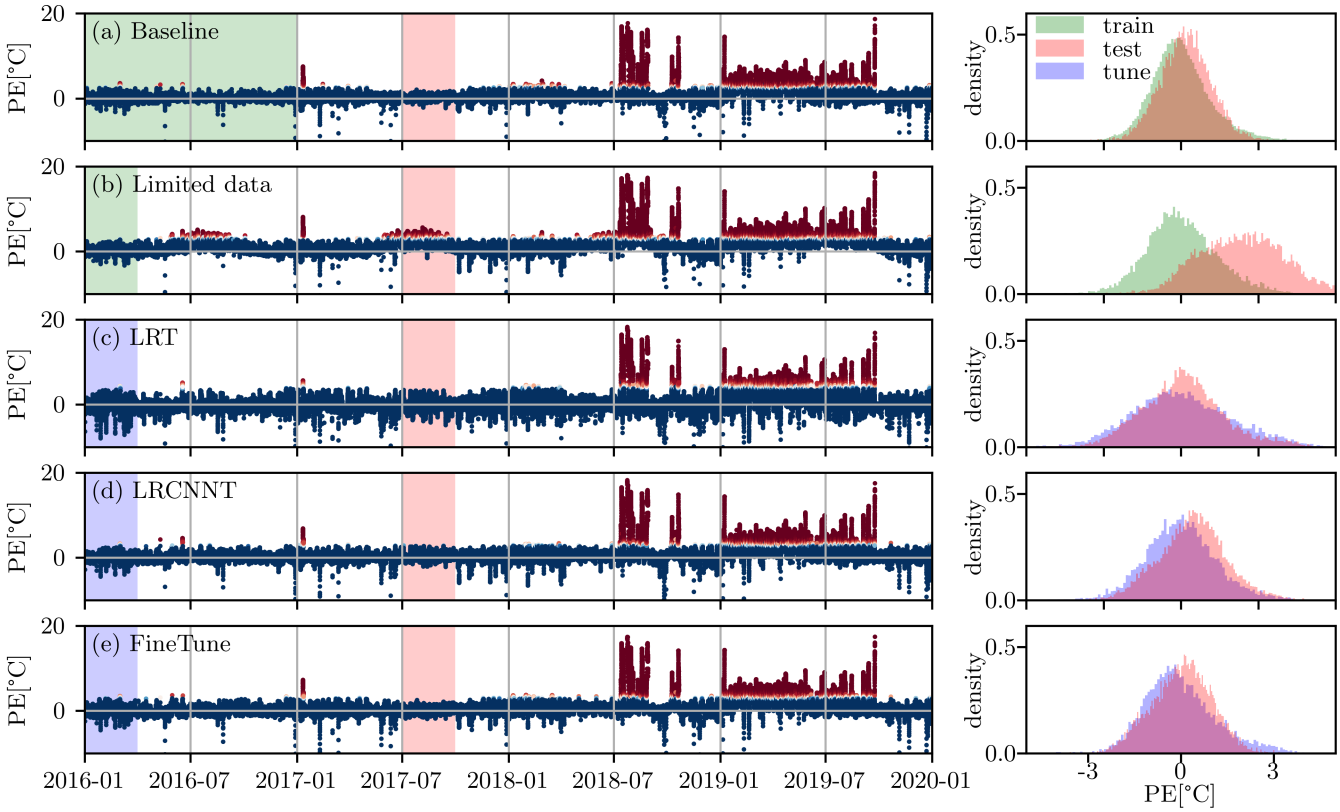

Figure 5. Comparison of different training schemes for cross-farm TL. Time dependent PEs (left column) and their distributions (right column) for 5 training schemes for FD on the same target turbine T_0. The baseline (a) and Limited data (b) schemes correspond to training from scratch of the base CNN model with 1 year and 3 months data respectively. In the TL frameworks LRT (c), LRCNNT (d) and FineTune (e) the base model is pre-trained with data from a source turbine S_1 of another wind farm and then tuned using the different TL frameworks to obtain PEs for the target turbine. Color code: green for training from scratch, blue for TL, red for testing. The training, tuning and testing periods are marked with the corresponding background color in the left column.

left column of this figure. The test sets (red) are of the same 3 months for all schemes.

Both the time dependent PEs and the distribution plots show clearly that training from scratch with 3 months data of the target turbine leads to a poor FD performance compared with the baseline, trained from scratch with 12 months of data. The main reason for this are seasonal domain shifts between the training months (winter 2016) and the test months (summer 2017): the functional dependence of the component (in this case gearbox bearing) temperature on the environmental variables (wind speed and ambient temperature) changes over the time scale of months. Therefore, training the CNN on winter data exclusively does not allow to extrapolate and lead to accurate enough predictions on summer data, which in turn causes a high false positive rate in summer. This seasonality of the PEs is largely corrected already by the simple LRT transformation of the target predictions. The resulting tuning and test distributions are only slightly shifted from each other and the false positive rate is rather low. However, the LRT yields a higher spread of the PE distribution than the baseline, leading to a high rate of missed detections of the true faults.

In order to preserve the good extrapolation provided by the LRT to the unseen summer domain, we use the LRT as a starting point for the LRCNNT framework. The additional CNNe is meant to allow for non-linear and time dependent corrections of the target domain predictions on top. We note that replacing the two-step training (LRT followed by CNNe) by a single step of a CNN only did not yield a similarly good transfer. The reason is that training a CNN from scratch on little and season-specific data has a poor performance, as we conclude from Figure 5(b). However, using a linear trans-

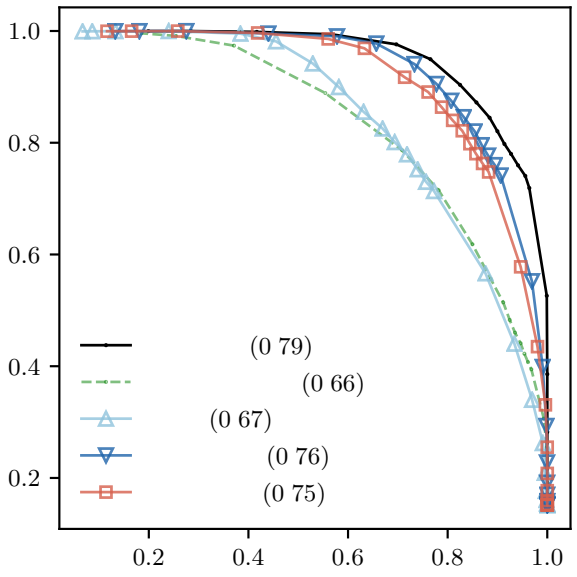

Figure 6. Precision-Recall curves for cross-farm TL. The source and the target turbines are from two different wind farms. The FD performance scores of the TL frameworks are compared with the baseline training scheme (solid black), and with the limited data scheme (dashed green), where the base CNN model is trained with 1 year and with 3 months of data from the target turbine respectively. Average precision values (area under curve) are stated in brackets for each training scheme.

formation to fix the linear component of the transfer, allows the CNNe to focus on learning only the non-linear and time dependent residuals instead of training it from scratch.

Both the LRCNNT and the FineTune transfer schemes are very effective in addressing both of the above difficulties: they both manage to correct for the winter-summer domain shift and avoid seasonal PEs. The PE distributions during tuning and testing periods are thus largely overlapping for both of these schemes. Moreover, the distributions are of a similar width as the baseline, thus leading to a FD performance which is similar in its FPR and TPR to the baseline reference.

The last statement can be quantified using the same approach as described in Section 3.3 to assign "true labels" to the data points. The resulting comparison of the recall-precision curves of all 5 schemes are displayed in Figure 6 together with their AP scores (see legend). Also here it is seen that the LRCNNT ($AP = 0.76$) and the FineTune ($AP = 0.75$) TL frameworks perform similarly to the baseline ($AP = 0.79$), with a slight advantage to the LRCNNT. The improvement achieved by supplementing the LRT with an additional CNNe for the prediction of the residual error component is seen clearly in this case: While the LRT ($AP = 0.67$) alone does not perform much better than training from scratch with only 3 months of data of the target turbine ($AP = 0.66$), the LRCNNT curve is close to the baseline, trained from scratch with a full year of data of this turbine. As opposed to the high performance of the LRT framework for transfer within the same farm, here the TL task is more complex and requires a more elaborate framework, involving either training of the additional CNNe or fine tuning of the original CNN using the 3 months of data of the target turbine.

Table 1. Seasonal Error Distribution Shifts

Model	μ shift	σ shift
Base Model	0.2 ± 0.03	-0.1 ± 0.02
Limited Data	1.46 ± 0.38	0.65 ± 0.32
LRT	-0.1 ± 0.11	-0.34 ± 0.05
LRCNNT	0.22 ± 0.09	-0.09 ± 0.04
FineTune	-0.24 ± 0.18	-0.15 ± 0.07

Table 1 summarizes the properties of the error distributions plotted in the right column of Figure 5 in terms of the distribution shift between the train/tune (winter) and test (summer). We quantify the distribution shift using the differences of the estimated mean μ and standard deviation σ between test and train. Note that these results are averages of 8 runs of the entire training scheme and are therefore given along with the standard deviation of the repeating experiments. From this quantification we confirm our finding that the LRCNNT framework provides the closest reproduction of the baseline results, both in terms of the distribution shift and in terms of the stochastic property of the training schemes (fluctuations between runs).

The opposite μ shift and the strong fluctuations for the FineTune imply that this TL framework is more sensitive to random effects in the training process. Therefore, despite its simplicity and relative efficiency, the FineTune framework suffers from drawbacks compared with the LRCNNT. Another clear conclusion from the table is that the limited data training suffers significantly more from random effects (due to the difficulty to regularize it properly), thus from fluctuations of the results between different runs. As such it is not only leading to low FD performance but also to high sensitivity of the outcomes to the choice of training data. We note that for Figures 5 and 6 we chose to display the "worst case scenario" of each of the training schemes.

It is worth discussing the common problem of goodness of transfer also in our context. Our transfer task comprises of two challenges. The first is overcoming the domain shift between turbines, both in the input data (due to variable ambient conditions, especially between farms) and in the output data (e.g due to different thermal insulation mechanisms, heating effects and other operating conditions). The second effect is the need to extrapolate from the tuning data set of the target turbine into an unseen season, with potentially different dependencies and dynamic properties of the data.

The goodness of transfer depends, as in other applications of

TL, on the selection of appropriate source for a given target. In particular we note that the LRT and LRCNNT algorithms are more effective than the fine tuning in extrapolating between seasons. This is to be taken with caution, especially if there are physical mechanisms that are only present in one of the seasons (such as active heating of the WT components in winter or cooling in summer).

Fine tuning methods are prone to forget important knowledge learned on the source turbine and be similar to training from scratch in case of a too high learning rate. Although this might require careful tuning of the learning rate, we found the a learning rate of $1/5$ of the original base model performs well for a large set of transfers between various farms. Since fine tuning ends up finding a compromise between the source and target domains and benefits from both, it usually requires an appropriate selection of the source turbine in order to extrapolate well out of the training distribution (Recht et al., 2019). An optimal selection of the source turbine, and the evaluation of seasonal effects on the TL task are both active research topics and will be pursued by us in the future.

5. Conclusions

In this paper we tested three algorithms for TL of a regression-based FD for WTs based on 10-minute SCADA data. After pre-training a base CNN model to predict the gearbox bearing temperature using healthy data from a source turbine, the different TL methods were used to obtain predictions for a target turbine, which were then used to extract HIs and set a threshold for fault detection. One of the three TL algorithms is the common fine tuning. The other two were developed by us to overcome the complex problem of TL with domain shifted inputs and outputs, having only single-season training data in the target domain. We showed that:

- For TL between WTs from the same wind farm a simple and computationally efficient TL method we developed based on linear regression achieves comparable FD performance to the base model trained on the target turbine. This framework can be therefore used to scale up the training of large wind farms by training on one source WT only instead of individually on each turbine.
- For TL across different farms, with a target turbine with limited healthy data, our LRCNNT framework outperformed other frameworks including a standard fine tuning approach.
- All three TL algorithms outperform training from scratch with limited data of the target turbine.
- The LRCNNT algorithms was shown to have a similar FD performance to the base model trained with abundant data from the target turbine.
- Evaluating and comparing TL approaches is possible even in the absence of true fault labels if one defines the base model as the reference for good FD performance.

We conclude that TL algorithms can enable scalable and reliable FD of WT based exclusively on readily available 10-minute SCADA data. The most adequate algorithm depends on the specific use-case (whether inside the same farm for upscaling purposes or between farms to overcome data scarcity). One of the most important open questions that will be pursued by us in the future is the challenge of optimizing the goodness of transfer by an appropriate selection of the source turbine.

Acknowledgment

This research was funded by Innosuisse - Swiss Innovation Agency under grant No. 32513.1 IP-ICT.

References

Chatterjee, J., & Dethlefs, N. (2020). Deep learning with knowledge transfer for explainable anomaly prediction in wind turbines. *Wind Energy*, *23*(8), 1693–1710.

Chen, W., Qiu, Y., Feng, Y., Li, Y., & Kusiak, A. (2021). Diagnosis of wind turbine faults with transfer learning algorithms. *Renewable Energy*, *163*, 2053–2067.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2018). Transfer learning for time series classification. In *2018 ieee international conference on big data (big data)* (pp. 1367–1376).

Guo, J., Wu, J., Zhang, S., Long, J., Chen, W., Cabrera, D., & Li, C. (2020). Generative transfer learning for intelligent fault diagnosis of the wind turbine gearbox. *Sensors*, *20*(5), 1361.

Hu, Q., Zhang, R., & Zhou, Y. (2016). Transfer learning for short-term wind speed prediction with deep neural networks. *Renewable Energy*, *85*, 83–95.

Jiang, G., Xie, P., He, H., & Yan, J. (2017). Wind turbine fault detection using a denoising autoencoder with temporal information. *IEEE/Asme transactions on mechatronics*, *23*(1), 89–100.

Lapira, E., Brisset, D., Ardakani, H. D., Siegel, D., & Lee, J. (2012). Wind turbine performance assessment using multi-regime modeling approach. *Renewable Energy*, *45*, 86–95.

Li, Y., Jiang, W., Zhang, G., & Shu, L. (2021). Wind turbine fault diagnosis based on transfer learning and convolutional autoencoder with small-scale data. *Renewable Energy*.

Michau, G., & Fink, O. (2021). Unsupervised transfer learning for anomaly detection: Application to complementary operating condition transfer. *Knowledge-Based Systems*, 106816.

Moradi, R., & Groth, K. M. (2020). On the application of transfer learning in prognostics and health management. *arXiv preprint arXiv:2007.01965*.

Qureshi, A. S., Khan, A., Zameer, A., & Usman, A. (2017). Wind power prediction using deep neural network based meta regression and transfer learning. *Applied*

Soft Computing, *58*, 742–755.

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do imagenet classifiers generalize to imagenet? In *International conference on machine learning* (pp. 5389–5400).

Schlechtingen, M., & Santos, I. F. (2014). Wind turbine condition monitoring based on scada data using normal behavior models. part 2: Application examples. *Applied Soft Computing*, *14*, 447–460.

Tautz-Weinert, J., & Watson, S. J. (2016). Using scada data for wind turbine condition monitoring–a review. *IET Renewable Power Generation*, *11*(4), 382–394.

Ulmer, M., Jarlskog, E., Pizza, G., & Goren Huber, L. (2020). Cross-turbine training of convolutional neural networks for scada-based fault detection in wind turbines. In *12th annual conference of the phm society, virtual, 9-13 november 2020* (Vol. 12).

Ulmer, M., Jarlskog, E., Pizza, G., Manninen, J., & Goren Huber, L. (2020). Early fault detection based on wind turbine scada data using convolutional neural networks. In *Proceedings of the european conference of the phm society* (Vol. 5).

Vercruyssen, V., Meert, W., & Davis, J. (2017). Transfer learning for time series anomaly detection. In *Proceedings of the workshop and tutorial on interactive adaptive learning@ ecmlpkdd 2017* (Vol. 1924, pp. 27–37).

Wang, Z., Zhang, J., Zhang, Y., Huang, C., & Wang, L. (2020). Short-term wind speed forecasting based on information of neighboring wind farms. *IEEE Access*, *8*, 16760–16770.

Ye, R., & Dai, Q. (2021). Implementing transfer learning across different datasets for time series forecasting. *Pattern Recognition*, *109*, 107617.

Yun, H., Zhang, C., Hou, C., & Liu, Z. (2019). An adaptive approach for ice detection in wind turbine with inductive transfer learning. *IEEE Access*, *7*, 122205–122213.

Zaher, A., McArthur, S., Infield, D., & Patel, Y. (2009). Online wind turbine fault detection through automated scada data analysis. *Wind Energy: An International Journal for Progress and Applications in Wind Power Conversion Technology*, *12*(6), 574–593.

Zhang, C., Bin, J., & Liu, Z. (2018). Wind turbine ice assessment through inductive transfer learning. In *2018 ieee international instrumentation and measurement technology conference (i2mtc)* (pp. 1–6).

Zheng, H., Wang, R., Yang, Y., Yin, J., Li, Y., Li, Y., & Xu, M. (2019). Cross-domain fault diagnosis using knowledge transfer strategy: a review. *IEEE Access*, *7*, 129260–129290.