# Task Embedding Temporal Convolution Networks for Transfer Learning Problems in Renewable Power Time Series Forecast

Jens Schreiber ✉[0000−0002−9979−8053], Stephan Vogt[0000−0002−3230−8822], and Bernhard Sick[0000−0001−9467−656X]

University of Kassel, Wilhelmshöher Allee 71, 34121 Kassel, Germany
{j.schreiber, stephan.vogt, bsick}@uni-kassel.de

**Abstract.** Task embeddings in multi-layer perceptrons (MLP) for multi-task learning and inductive transfer learning in renewable power forecasts is an exciting new technique. In many cases, this approach improves the forecast error and reduces the required training data. However, it does not take the periodic influences in power forecasts within a day into account, i.e., the diurnal cycle. Therefore, we extended this idea to temporal convolutional networks to consider those in tasks of day-ahead power forecasts for renewables. We propose transforming the embedding space, which contains the latent similarities between tasks, through convolution and providing these results to the network's residual block. The proposed architecture significantly improves the forecast accuracy up to 25% for multi-task learning for power forecasts on the open EuropeWindFarm and GermanSolarFarm datasets compared to the MLP approach. Based on the same data, we achieve a ten percent improvement for the wind datasets and more than 20% in most cases for the solar dataset for inductive transfer learning without catastrophic forgetting. Finally, we are the first to propose zero-shot learning for renewable power forecasts. The proposed architecture achieves an error as good as the task embedding MLP with a full year of training data in the respective experiments.

**Keywords:** Transfer Learning · Time Series · CNN · Renewables · TCN.

## 1 Introduction

The Paris commitment demands to limit human-induced global warming below 2°C above pre-industrial levels to reduce the impact of the climate crisis. To achieve the commitment, renewable energy resources need to increase their share from 14% in 2015 to 63% in 2050 in the worldwide energy production [1]. To assure grid stability, energy suppliers require reliable power forecasts based on numerical weather predictions (NWPs) as input due to the weather dependency. These predicted weather features, such as wind speed or radiation, are the input to models predicting the expected power generation in *day-ahead* forecasts between 24 and 48h into the future.

Another problem is that each of those parks typically has individual characteristics to learn by the forecast models. Therefore, a single forecast model is learned per park or even for a single wind turbine in practice. Assuming there are $30,000$ wind facilities with an average number of $10,000$ parameters per model solely in Germany, this makes a total of 300 million parameters to train. Additionally, the hyperparameters need to be optimized.

The training of those models contradicts the Paris commitment, as the training and even the inference themselves have an extensive energy demand and cause a considerable amount of carbon emmission [2]. To reduce the number of models that need to be optimized, while leveraging knowledge between parks, [3] proposed to utilize multi-task learning (MTL) approaches. In their task embedding multi-layer perceptron (MLP) the discrete task information, *which task is to be predicted*, is encoded through an embedding layer and concatenated with other input features. However, this approach still requires a reasonable amount of training data in the task of day-ahead power forecasts for renewables. This training data is often not available for new parks. Therefore, in [4], the task embedding MLP extracts knowledge from a set of source tasks, e.g., several known wind parks. This knowledge is adapted to unseen target tasks, during initial training, with limited training instances. In their approach, a Bayesian variant of the task embedding MLP leads to the best results for wind parks with limited training data.

However, there are periodic patterns, i.e., the diurnal cycle, within day-ahead forecasts affecting the forecast error [5]. Those are not considered within an MLP, even though the substantial benefits of time series forecasts through convolutional neural networks (CNNs) are known [6]. Further, ideally, we can forecast parks without any historical power measurements. This zero-shot learning paradigm is essential to assure reasonable forecasts from the beginning of a new park. Therefore, we are interested in answering the following research questions:

*Question 1.* Can an MTL CNN architecture improve the forecast error for wind and photovoltaic (PV) parks compared to a similar MLP MTL architecture?

*Question 2.* Are CNN based MTL architectures capable of providing good forecasts in zero-shot learning for renewable power forecasts?

*Question 3.* Are CNN based MTL architectures beneficial during inductive transfer learning (TL) compared to a similar MLP MTL approach?

To answer those questions, we propose task-*temporal convolution network (TCN)*, see Fig. 1. *Task-TCN* encodes task-specific information through an embedding layer. Thus, each park gets an *increasing* task ID ($m \in \mathbb{N}^+$) assigned as input to the embedding layer. We assure that the TCN learns relations between tasks during training by adding the encoded task information in a residual block. By encoding the task ID through an embedding layer, we can extend the network to new parks while avoiding catastrophic forgetting of previous parks. We utilize the idea of task embeddings in our proposed method as well in our baseline to assure a comparison of similar MTL architectures. Evaluation of the task-TCN on the

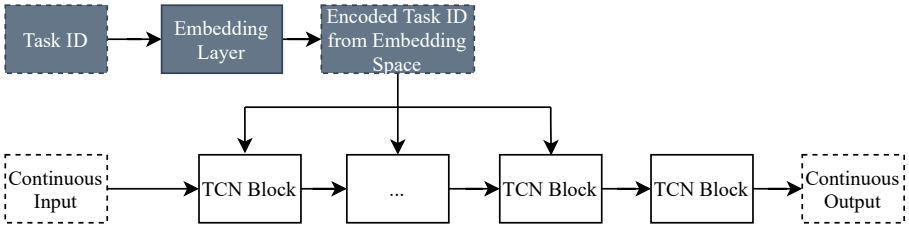


**Fig. 1.** Task-TCN encodes a task ID, for each task, through an embedding layer. The learned encoding from the embedding space is *added* in the residual block to provide task-specific forecasts.

open EuropeWindFarm and GermanSolarFarm[1], in comparison to the Bayesian task embedding MLP as the baseline, leads to the following contributions:

- The proposed task-TCN architecture, for MTL renewable power forecasts, leads to improvements of up to 25 percent.
- The proposed task-TCN leads to improvements of more than ten percent for wind and up to 40 percent for the solar dataset for inductive TL problems.
- We are the first to propose zero-shot learning for renewables and the task-TCN achieves an error as good as the task embedding MLP with a full year of training data in the respective experiments.

The source code and supplementary material of the experiments are openly accessible[2]. The remainder of this article is structured as follows. Section 2 describes related work. The following Section 3 introduces relevant definitions and details the proposed approach. We describe the datasets, discuss the experiment and most essential findings in Section 4. In the final section, we summarize our work and provide insides for future work.

## 2 Related Work

To answer our research question, we review related work in the field of MTL, (inductive) TL, and zero-shot learning focusing on power forecasts for wind and PV. Formal definitions of those topics are given in Sec. 3.1. For comprehensive surveys on TL and MTL, outside the domain of renewables, refer to [7, 8].

Most of the related work applying TL focuses on utilizing neural networks. However, the study of [9] proposes an MTL strategy for Gaussian processes to forecast PV targets. By clustering wind parks through their distribution, a weighting scheme provides predictions for a new park in [10]. The approach of [11] utilizes an auxiliary dataset, created through k-nearest neighbors, to improve the forecasts error in short-term wind power predictions. This data driven method is in principle adaptable to our approach but is outside the focus of our work.

[1] `https://www.uni-kassel.de/eecs/ies/downloads`, accessed 2021-06-30
[2] `https://github.com/scribbler00/task-TCN`, accessed 2021-06-30

The same argument holds for the instance-based TL for day-ahead solar power forecasts proposed in [12].

To find a suitable representation for (inductive) TL and MTL various articles utilize autoencoders, e.g, [13, 14] and more recently a self-attention based encoder-decoder structure [15]. One problem with autoencoders is that the information extraction is typically limited to the input space and neglects knowledge shared between tasks in the target space, which is, e.g., achieved through hard parameter sharing (HPS). Further, all of those articles consider (ultra) short-term forecasts and do not examine the more difficult day-ahead predictions. Most of the previously mentioned work utilizing neural networks focuses on feed forward networks neglecting the time series problem at hand. However, more recent work also considers recurrent networks and finetuning to achieve good results for ultra-short-term forecast horizon of PV [16]. Moreover, the article [17] provides a strategy for quantile regression for day-ahead solar power forecasts using CNNs and finetuning. But both articles are missing out on making use of multiple targets, which we consider in our approach.

The authors of [3] utilize a task embedding for encoding of task-specific information in a HPS network. This approach is similar to *word2vec* [18] and encoding of categorical features to replace one-hot encodings in [19]. The task embedding improves the forecast error with a minimal amount of parameters for day-ahead wind and solar power forecasts in a MTL setting. A thorough evaluation of inductive TL for day-ahead wind power forecast is given in [4]. Furthermore, a Bayesian task embedding assures that similar tasks are close to one another in case of limited data. The Bayesian task embedding is superior to models learned from scratch and a traditional HPS. The task embedding can be considered state of the art due to their excellent results with minimal parameters even when trained with limited data. Further, the approach of task embedding avoids catastrophic forgetting. However, both of the articles utilizing task embeddings neglect to extend this approach to CNNs that we address.

Overall, the related work shows limited research on TL and MTL for day-ahead power forecasts. Furthermore, none of those approaches provide a framework for wind as well as PV power forecasts. Moreover, there is limited research in addressing TL challenges through recent work in time series forecasts [20]. Note, due to the recent success and benefits of CNNs over recurrent networks [20, 21] we focus on the former. Further, to the best of our knowledge, there has been no attempt to provide zero-shot forecasts in the field of renewable energies. Besides, none of those proposals consider a unified framework that takes recent advances in time series forecast into account, such as TCN, while being applicable to MTL, inductive TL, and zero-shot learning.

## 3 Proposed Method

In the following, we detail essential definitions, relate the task embedding MLP to MTL, and describe our proposed architecture.

### 3.1 Definition of MTL, TL, and Zero-Shot Learning

The following definitions have been introduced in [7, 8]. We slightly modified them for a consistent formulation of multi-source TL and MTL.

**Definition 1 (Domain).** *A domain is defined by* $\mathcal{D} = \{\mathcal{X}, P(X)\}$*, where* $\mathcal{X}$ *is the feature space and* $P(X)$ *is the marginal distribution with* $X = \{\mathbf{x} \mid \mathbf{x}_i \in \mathcal{X}, i = 1, \ldots, N\}$*.*

**Definition 2 (Task).** *The task of a domain is defined with* $\mathcal{T} = \{\mathcal{Y},\ f(\cdot)\}$*, where the function is defined by* $f : \mathcal{X} \rightarrow \mathcal{Y}$*. The function* $f(\cdot)$ *is learned by training instances* $\{\mathbf{x}_i, y_i\}$ *with* $\mathbf{x}_i \in X$ *and* $y_i \in Y$*, where* $Y = \{y \mid y_i \in \mathcal{Y}, i = 1, \ldots, N\}$*. The function* $f(\cdot)$ *describes characteristics of the distribution* $P(Y \mid X)$*. In a Bayesian approach those are samples and the expectation in a frequentist view.*

Note, in these definitions, $\mathbf{x}_i$ are available weather predictions and $y_i$ are historical power measurements of a park. We assume that $X$ and $Y$ are ordered sets for the required time series forecasts for simplicity.

**Definition 3 (Inductive Transfer Learning).** *Inductive Transfer Learning has the goal to transfer knowledge from source (S) tasks* $\{\mathcal{T}_{S_m}\}_{m=1}^{m=M}$ *to a target (T) task* $\mathcal{T}_T$*. Therefore, we use* $M \in \mathbb{N}^+$ *source domains* $\{(\mathcal{D}_{S_m}, \mathcal{T}_{S_m}) \mid m = 1, \ldots, M\}$ *and (limited) training instances* $\{\mathbf{x}_{T_i}, y_{T_i}\}$ *with* $\mathbf{x}_{T_i} \in X_T$ *and* $y_{T_i} \in Y_T$ *to learn a function* $f_T(\cdot)$*.*

In contrast to transductive TL, labeled training data in the target domain is available in inductive TL.

**Definition 4 (Zero-shot learning).** *Zero-shot learning can be interpreted as unsupervised transductive TL [22]. In this setting, meta-information from source and target tasks is used to select an appropriate prediction function* $f_{S_m}(\cdot)$ *to predict the target task.*

In comparison to transductive TL, zero-shot learning is not using a domain adaption approach between the source(s) and the target. This unsupervised approach makes it an even more challenging problem as no assumptions are made of the source and target domain [22].

**Definition 5 (Multi-Task Learning).** *In MTL approaches, each task is accompanied by a domain* $\mathcal{D}_m$ *with* $i \in \{1, \ldots, N\}$ *training instances* $(\mathbf{x}_i^m, y_i^m)$*, where* $\mathbf{x}_i^m \in X^m$*,* $y_i^m \in Y^m$*,* $m \in 1, \ldots, M$*, and* $M \in \mathbb{N}^+$ *tasks.*

In contrast to inductive transfer and zero-shot learning, all tasks have a sufficient amount of training data in MTL problems. Furthermore, in MTL we are typically interested in improving all tasks' forecast errors simultaneously.

### 3.2 Proposed Method

Typically, in MTL, there are two approaches to share knowledge and improve the forecasts error: soft parameter sharing (SPS) and HPS. In HPS architectures, there are predominantly common layers that are the same for all tasks and few task-specific layers. In SPS, each task has an individual model and similarity is enforced through regularization [8]. As SPS requires additional parameters compared to HPS, due to the separate networks for each task, we focus on the latter to reduce the energy demand by the additional parameters.

Previously, we argued that the function $f(\cdot)$ describes the conditional distribution $P(Y \mid X)$, where $Y$ is a set of observed power generations and $X$ are observations of the numerical weather prediction. However, this description neglects the possibility to model individual forecasts for different parks in MTL settings. Therefore, we describe how to model this dependency through an embedding layer in an MLP as suggested in [3, 4]. By doing so, we are the *first to provide a mathematical description* between the task embedding and the MTL definition. Afterward, we explain the Bayesian embedding layer and detail how our approach extends the idea of task embeddings to TCNs.

**Task-Embedding for MLPs** encodes an increasing (discrete) task ID about *which task* $m \in \mathbb{N}^+$ *is to be predicted* through an embedding layer and concatenates results of this embedding space with other continuous input features [3, 4]. In the following we connect embedding layers to the MTL definition. Therefore, consider a common function $h(\cdot)$ for all $M$ tasks that approximates $P(y_m \mid X_m, g(m))$, where $m$ is the discrete information whose task is to be predicted with $m \in 1, \ldots, M$ and $M \in \mathbb{N}^+$. $g(m)$ is then a transformation into an arbitrary real valued dimension.

Assuming we have such a transformation, the conditional modeling allows us to develop a model without task-specific layers. The required information on the task is given through $g(m)$. Therefore, the function $h(\cdot)$ for MTL has the training instances $\{\mathbf{x}_i^m, y_i^m, g(m)\}$. Ideally, we want a mapping $g$ that is beneficial for the MTL problems, e.g., similar tasks are close to another in the embedding space. Respectively, an MTL approach needs to learn this mapping function during training. For such a problem, the authors of [19] propose the following equation of an embedding layer:

$$g(m) = \sum_{\alpha=1}^{M} \mathbf{w}_{\alpha\beta} \boldsymbol{\delta}_{m\alpha} = \mathbf{w}_{m\beta}, \tag{1}$$

where $\boldsymbol{\delta}_{m\alpha}$ is the Kronecker delta. Therefore, $\boldsymbol{\delta}_{m\alpha}$ is a vector of length $M$ ($M$ tasks), where only the entry with $\alpha = m$ is non-zero and $\mathbf{w}_{\alpha\beta} \in \mathbb{R}^D$ is the learnable vector at this position. Respectively, the function $g$ maps the discrete value $m$ of a task ID through a one-hot encoding to a trainable vector.

This transformation is then concatenated with other (continuous) features. Due to the joint training for multiple tasks, e.g., within one batch we have various tasks, it is beneficial for the network to learn a mapping where similar

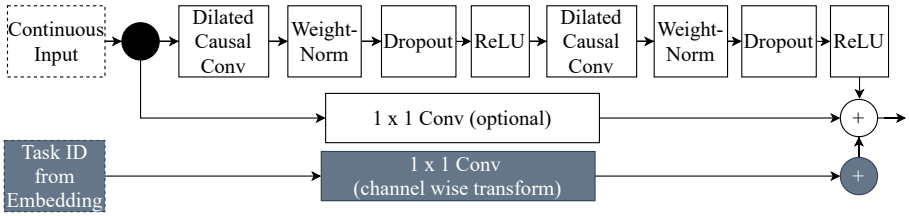


**Fig. 2.** Residual block of the task-TCN. The encoded task ID, from the embedding space, is transformed through a 1D convolution. The transformed task ID is then added to the results of the original TCN residual block.

tasks have a similar vector $\mathbf{w}_{\alpha\beta}$. This mapping allows the utilization of a similar transformation in later layers for similar domains and targets.

**Bayesian Task-Embeddings** are especially interesting in the scenario where limited data for a (single) task is available. If a limited amount of data is available, e.g., in the inductive transfer learning problem, a Bayesian approach allows placing an identical, independent prior on $\mathbf{w}_{m\beta}$. This prior ensures that indistinguishable tasks, due to limited data, are within a similar neighborhood. At the same time, tasks with sufficient data have an encoding that is different from other tasks. We apply *Bayes by backprop* solely on the embedding layer. By using a standard normal distribution and sampling weights solely from the embedding, to minimize the expected lower bound (ELBO)[23], we limit the training effort for the number of additional parameters and benefit from the embedding space's Bayesian approach.

**Task-Embedding for Temporal Convolution Neural Networks** are a way to take advantage of periodic patterns, e.g., the diurnal cycle, in *day-ahead* forecasts [5] in an MTL architecture. In MTL day-ahead power forecast problems, we forecast the expected power $y_{24}^m, \ldots, y_{48}^m$ for $m \in 1, \ldots, M$ parks. In contrast to intra-day forecasts between 0 and 23 hours into the future, day-ahead forecasts are more challenging as the forecast error increases with an increasing forecast horizon [24]. We have the numerical weather features $\mathbf{x}_{24}^m, \ldots, \mathbf{x}_{48}^m$ from a weather prediction originating from $t_0$ (e.g. originating from 0 o'clock UTC) as input. The fundamental building block of task-TCN is the *TCN* [21], a CNN architecture for time series. In this architecture, the residual blocks limit the problem of vanishing gradients, see Fig. 2.

In the previously discussed task embedding for MLPs, the embedding is concatenated with the continuous weather input features to provide task-specific forecasts. However, simply concatenating the embedding output along the time axis in CNNs leads to additional channels with the same information for each time step. Respectively, having redundant information along the time axis. We

found this not to work well in preliminary experiments due to the redundant information causing a plateau in the training error.

Instead, we propose to *add* the result from $g(m_i)$ at the end of the residual block, see Fig. 2. Therefore, we first transform the encoded task ID through a 1-D convolution layer with a kernel size of one. The transformation gives the network the possibility to learn an encoding that is different in each channel. Effectively, this allows the network to adjust each feature (channel) specifically for the task at hand. For instance consider, that the output of the first residual block has 100 channels of 24 time steps. For a single sample, the dimension will be $1 \times 100 \times 24$. Also, consider that each task has an embedding vector of dimension $1 \times 10$. The embedding tensor will then be $1 \times 10 \times 24$. After applying the transformation, through the 1D convolution, the transformed embedding tensor will also be of dimension $1 \times 100 \times 24$. By adding this tensor in the residual block, we provide distinct task related information in each channel, giving the network the possibility to learn task specific features in each channel while limiting redundant information in contrast to a concatenation. This principle applies to any residual block within the network. A study on the diverse learned similarities of tasks per channel is presented in the supplementary material.

## 4 Experimental Evaluation of the Task-Temporal Convolution Network

To answer our research questions, we conducted one experiment for each of the three research questions. We evaluate each of these experiments on the datasets explained in Sec. 4.1. Sec. 4.2 lists relevant evaluation measures. The design of experiments and the evaluations are detailed in Sec. 4.3, 4.4, and 4.5. The models detailed in Sec. 4.3 are the source models for the other experiments. The description also includes the Bayesian MLP that is the baseline in all experiments. All models are similar in the sense that they share the task embedding architecture and hyperparameters are identical. We refer to *TCN first* and *MLP first* when the encoded information is considered only in the first layer and *TCN all* when the task information is added in all, except the last, residual blocks.

### 4.1 GemanSolarFarm and EuropeWindFarm Dataset

In the GermanSolarFarm and the EuropeWindFarm dataset, the uncertainty of the NWP makes it challenging to predict the generated power. We refer to these as wind and solar datasets in the following. This mismatch between the weather forecast and power can be seen in Fig. 3 and Fig. 4. As weather forecasts are valid for a larger area, a mismatch between the forecast horizon and the placement of a park causes uncertainty. The uncertainty also increases with the increasing forecast horizon of the weather prediction [5].

The solar dataset consists of 21 parks, while the wind dataset consists of 45 parks. Both datasets include day-ahead weather forecasts, between 24 and 48h into the future, of the European center for medium-range weather forecast

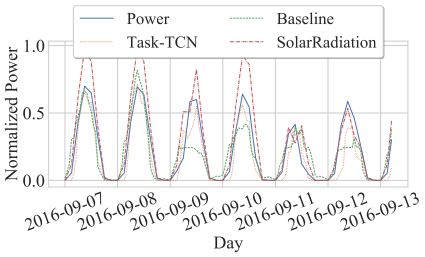


**Fig. 3.** Forecasts of a solar park with 30 days training data.

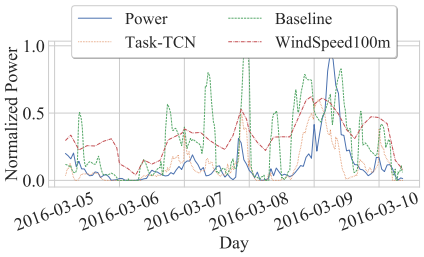


**Fig. 4.** Forecasts of a wind park with 30 days training data.

model [25]. The data also includes the normalized historical power measurements. The solar dataset has a three-hourly resolution for two years and two months. We linearly interpolate the data to have a resolution of one hour to increase the number of samples, especially for the inductive TL problem. The wind dataset has data from two years with an hourly resolution. The data is linearly interpolated to have a 15-minutes resolution. Respectively, the PV dataset has 24 timestamps per day and 96 for the wind dataset. Days not fulfilling these criteria are neglected. The first year is considered for training while the remaining data is used for testing. Weather features are standardized on the training data. This process results in about 8200 samples per park for training and 9400 for testing in the solar dataset. The wind dataset has about 28400 training and 27400 test samples. The solar dataset contains 38 features, such as sun position and solar radiation. The wind dataset contains seven features, such as wind speed and wind direction. To test the algorithm for inductive TL and zero-shot learning, we split the parks through five-fold cross-validation so that each park is a target task once while having different source tasks.

One disadvantage of these two datasets is that they do not contain any meta-information directly, not even the location. Typically, the meta-information in zero-shot learning is used as a proxy to find a similar source task. However, this information is also not available in other open datasets for renewable power forecasts. Therefore, we create meta-information through the input space, see Sec. 4.4. One advantage of the datasets is that they are distributed throughout Germany and even Europe. This distribution makes the MTL, inductive TL, and zero-shot learning more realistic as one cannot assume that parks are nearby in practice.

### 4.2 Evaluation Measures

In all experiments, we are considering the normalized root-mean-squared error (nRMSE). A significant difference between a (reference) model and the baseline is tested through the Wilcoxon test (with $\alpha = 0.05$), by comparing the nRMSE of a park from the baseline and a reference model. To calculate improvements

against the baseline, we consider the mean skill of all $m$ parks by

$$\text{skill}_m = 1 - \frac{\text{nRMSE}_{\text{reference}_m}}{\text{nRMSE}_{\text{baseline}_m}} \text{and skill} = \frac{1}{M} \sum_{m=1}^{m=M} \text{skill}_m, \tag{2}$$

where values larger than zero indicate an improvement upon the baseline.

### 4.3 MTL Experiment

In this section, we conduct an experiment to answer the research question:

*Question 1.* Can an MTL CNN architecture improve the forecast error for wind and PV parks compared to a similar MLP MTL architecture?

**Findings:** The proposed task-TCN improves the baseline up to 18% for the solar dataset and 13% for the wind dataset.

**Design of Experiment:** Ten percent of the training data is validation data for hyperparameter optimization. Details on the grid search for hyperparameter optimization are available in the supplementary material. Due to the five-fold cross-validation, each park is trained four times, see Sec. 4.1. We calculate the mean nRMSE of a park for these runs. In total, we train six models and evaluate them through the nRMSE and the skill. The first two are the task embedding MLP (*MLP first*) and the Bayesian variant of the embedding layer, where the latter is considered the baseline. Both MLP variants can be considered as state of the art, see Sec. 2. For the TCN we provide the encoded task ID from the embedding once only to the *first* residual block (*TCN first*) and in the other variant to *all* except the last layer (*TCN all*). We exclude the last layer as we assume that sufficient information from the embedding is available at this point in the network. Afterward, only a transformation to the power of a specific task is required. For each of those two models, we consider a Bayesian and a non-Bayesian variant. We do not consider an ablation study without the task embedding as the architecture would then no longer be extendable to new tasks without catastrophic forgetting for inductive TL. Further, the improvement through the embedding over single-task learning is also shown in [3, 4].

**Detailed Findings:** All variants of the TCN lead to a significant improvement compared to the Bayesian *MLP first* as the baseline, see Table 1. The non-Bayesian *MLP first* has significantly worse results than the baseline for both datasets. In the case of the PV dataset, the non-Bayesian *TCN all* has the smallest mean nRMSE (0.071), std, and best skill. It is essential to consider that probably due to the larger correlation between PV parks in this dataset, as described in [3], it is beneficial for the TCN to share information in *all* residual blocks. In contrast, the wind dataset tasks are less correlated compared to the PV dataset [3]. Therefore, the non-Bayesian *TCN first* has the best results with a mean nRMSE of 0.136.

**Table 1.** Evaluation results for the wind and PV dataset for the mean nRMSE and standard deviation (std) from all parks. The asterisk symbol marks significantly different nRMSE values of reference (Ref.) models than the baseline (BS). The significance is tested through the Wilcoxon signed-rank test with $\alpha = 0.05$. In case that the skill is larger than zero, this indicates a significant improvement upon the baseline.

| | | DataType | PV | | | Wind | | |
|---|---|---|---|---|---|---|---|---|
| | | | Skill | nRMSE | std | Skill | nRMSE | std |
| Model | Type | EmbPos | | | | | | |
| MLP | Bayes | First (BS) | 0.000 | 0.087 | 0.005 | 0.000 | 0.184 | 0.007 |
| | Normal | First (Ref.) | -0.087 | 0.095* | 0.015 | -0.239 | 0.229* | 0.047 |
| TCN | Bayes | All (Ref.) | 0.092 | 0.079* | 0.002 | 0.092 | 0.169* | 0.005 |
| | | First (Ref.) | 0.098 | 0.078* | 0.002 | 0.089 | 0.169* | 0.007 |
| | Normal | All (Ref.) | **0.181** | **0.071*** | **0.001** | 0.254 | 0.137* | 0.002 |
| | | First (Ref.) | 0.174 | 0.072* | 0.001 | **0.258** | **0.136*** | **0.002** |

Interestingly, for both datasets, the Bayesian TCNs perform substantially worse than their non-Bayesian variants. However, the Bayesian *MLP first* error is significantly better than the non-Bayesian approach. In the case of the Bayesian *MLP first*, the prior probably allows the model to better learn the commonalities and differences between tasks, especially in case of insufficient data [4]. However, in the case of the TCN, the additional channel wise transform allows the network to learn an encoding that is different in each channel. Due to this possibility of learning a vast amount of features that reflect different amounts of similarities between tasks, it makes the benefits of a Bayesian approach dispensable, especially since they are sensitive to the selection of the prior, which is constant in all experiments. Nonetheless, the results are promising for the following experiments, as the Bayesian *MLP first* baseline improves upon the non-Bayesian MLP similar to [4] and the proposed task-TCN improves upon the baseline.

### 4.4 Zero-Shot Learning Experiment

In this section, we conduct an experiment to answer the research question:

*Question 2.* Are CNN based MTL architectures capable of providing good forecasts in zero-shot learning for renewable power forecasts?

**Findings:** The best TCN models achieve excellent forecast errors for PV and wind as these models have a similar mean nRMSE compared to the baseline from the previous experiment with a full year of data. However, outliers for the wind dataset are present that need to be considered in future work.

**Design of Experiment:** In this experiment, we use the same model as described in the previous section. Initially, we need to find a suitable source task applicable to an unknown task. As stated earlier in Sec. 4.1, the two datasets do not have meta-information directly applicable to select an appropriate source task.

However, a common approach in TL for time series is to use dynamic time warping (DTW) as a similarity measure in the feature space between the source and the target [20]. We assume that a similar park type, e.g., the same rotor diameter, is placed in similar regions with identical weather features. Even though this is not optimal, it is still a reasonable assumption to make. For instance, wind parks near coasts typically have larger rotor diameters than parks placed in a forest. To determine the most similar wind or PV park, we calculate the mean squared difference through DTW between the source and the target. In the case of the wind dataset, we use the wind speed at 100 meter height. For the solar dataset, we utilize direct radiation, as those two are the most relevant features to forecast the expected power [24]. We calculate the similarity based on the first year's training data of the source and target task. Using the entire year is possible, as the input features are themselves forecasts and are extractable for the past. After finding the most prominent candidate through the training data, we forecast on the test dataset. Based on these forecasts, we calculate the nRMSE for each park and the skill to measure the improvement.

**Detailed Findings:** Table 2 summarizes the respective results. For the PV dataset, the non-Bayesian *TCN all* achieves the best mean nRMSE of 0.081. All other TCN models have a similar mean nRMSE of 0.082. This zero-shot learning result is outstanding since it is still better than the baseline from the previous experiment with a mean nRMSE of 0.087 with a full year of training data.

The Bayesian *TCN all* and *TCN first* have the best mean nRMSE with 0.182 and improvements above 10 percent compared to the zero-shot baseline of this experiment for the wind dataset. The non-Bayesian TCNs have a similar nRMSE of 0.188. Again, the result of the best TCNs is less than the mean nRMSE (0.184) of the baseline from the previous experiment.

Nonetheless, there are two interesting observations. First, for all models on the wind dataset, outliers above an nRMSE of 0.5 are present. As the wind dataset has parks spread throughout Europe with distinct topographies and weather situations, these outliers are not surprising. Furthermore, the source task selection is solely based on DTW, as a proxy for the missing meta-information. Respectively, these outliers need to be considered in future work, e.g., through additional meta-information. Second, for the wind dataset, the Bayesian TCN variants are better than their non-Bayesian counterparts. This effect is surprising, as their basis from the previous experiment behaves inversely. We can assume that due to the prior, acting as a regularizer, in Bayesian TCNs we have fewer outliers for indistinguishable tasks. However, this should be considered in future work.

### 4.5 Inductive TL Experiment

In this section, we conduct an experiment to answer the research question:

*Question 3.* Are CNN based MTL architectures beneficial during inductive TL compared to a similar MLP MTL approach?

**Table 2.** Result for zero-shot learning, cf. with Table 1.

| | | DataType | PV | | | Wind | | |
|---|---|---|---|---|---|---|---|---|
| | | | Skill | nRMSE | std | Skill | nRMSE | std |
| Model | Type | EmbPos | | | | | | |
| MLP | Bayes | First (BS) | 0.000 | 0.093 | **0.018** | 0.000 | 0.202 | **0.071** |
| | Normal | First (Ref.) | -0.055 | 0.098 | 0.028 | -0.159 | 0.226* | 0.082 |
| TCN | Bayes | All (Ref.) | 0.123 | 0.082* | 0.018 | **0.109** | **0.182*** | 0.076 |
| | | First (Ref.) | 0.123 | 0.082* | 0.019 | 0.104 | 0.182* | 0.073 |
| | Normal | All (Ref.) | **0.129** | **0.081*** | 0.020 | 0.058 | 0.188 | 0.078 |
| | | First (Ref.) | 0.127 | 0.082* | 0.019 | 0.059 | 0.188 | 0.077 |

**Findings:** With only 90 days of training data, the nRMSE of the TCN architecture is similar to the TCN with an entire year of training data. This result shows that the proposed model is extendable to new tasks having a similar error while avoiding catastrophic forgetting with improvements above 25% compared to the baseline.

**Design of Experiment:** As seasonal influences affect the forecast error [24], we test how the different seasons as training data result in other target errors. Further, to test the influence of the amount of available training data, we train with data from $7, 14, 30, 60, 90$ and additionally 365 days. To avoid catastrophic forgetting, we only finetune the embedding layer. In this way, the network needs to learn a transformation of the task ID similar to previous source parks and use this encoding. Finally, we tested two methods to initialize the embedding of the new task. The first one uses the *default* initialization strategy of pytorch. The second one *copies* the task embedding based on the most similar task through the smallest mean squared error on 10 percent of the training data.

**Detailed Findings:** In the following, we show exemplary results of the spring season for both datasets. Those lead to the best results compared to other seasons for training. When comparing models with another, observations are similar to those presented in the following also for other seasons. The results with a year of training data and other seasons are available in the supplementary material.

The results for the wind dataset are summarized in Table 3. For all except one case, copying the task ID leads to an improved mean nRMSE compared to the default initialization strategy for all models and available training amounts. All TCN variants achieve a significant improvement upon the baseline for all different numbers of training data. The non-Bayesian *TCN all*, where we *copy* the most similar embedding vector, has the best mean nRMSE for most training amounts. The non-Bayesian *MLP first* is either significant worse or has a similar nRMSE than the Bayesian MLP.

Table 4 summarizes the same results for the solar dataset. Again, copying the task ID leads to an improved mean nRMSE for all models, compared to

**Table 3.** Mean nRMSE for spring of wind data set. Significant differences of the reference (Ref.) compared to the baseline (BS) is tested through the Wilcoxon signed-rank test with $\alpha = 0.05$ and marked with *.

| | | | | nRMSE | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | DaysTraining | 7 | 14 | 30 | 60 | 90 |
| Model | Embedding Type | Embedding Position | Embedding Initialization | | | | | |
| MLP | Bayes | First | Copy (Ref.) | 0.233 | 0.229 | 0.213 | 0.206 | 0.186 |
| | | | Default (BS) | 0.234 | 0.229 | 0.213 | 0.205 | 0.186 |
| | Normal | First | Copy (Ref.) | 0.237 | 0.233 | 0.212 | 0.200 | 0.181 |
| | | | Default (Ref.) | 0.255 | 0.247* | 0.229* | 0.217* | 0.200* |
| TCN | Bayes | All | Copy (Ref.) | 0.178* | 0.192* | 0.171* | 0.170* | 0.162* |
| | | | Default (Ref.) | 0.186* | 0.200* | 0.177* | 0.173* | 0.163* |
| | | First | Copy (Ref.) | 0.177* | 0.195* | 0.176* | 0.172* | 0.165* |
| | | | Default (Ref.) | 0.184* | 0.202* | 0.180* | 0.175* | 0.165* |
| | Normal | All | Copy (Ref.) | **0.161*** | **0.173*** | 0.152* | **0.141*** | **0.137*** |
| | | | Default (Ref.) | 0.179* | 0.188* | 0.155* | 0.146* | 0.139* |
| | | First | Copy (Ref.) | 0.163* | 0.181* | **0.148*** | 0.142* | 0.137* |
| | | | Default (Ref.) | 0.187* | 0.199* | 0.162* | 0.152* | 0.142* |

the default initialization strategy, except in one case. A significant improvement is achieved through the non-Bayesian *MLP first* and *copying* the encoded task ID. All TCN models achieve a significant improvement upon the baseline. In most cases, the *TCN first* that *copies* the embedding vector has the best or at least a similar result. Interestingly, the non-Bayesian *MLP first* is significantly better than the Bayesian baseline, which contrasts with the results from the MTL experiment. Potentially, this contradiction is caused by an insufficient amount of training epochs during inductive TL for the Bayesian model that is required to reduce the ELBO.

Due to these results, we can summarize that the TCN models achieve significant improvements compared to the MLP baselines. This result is not surprising, as, also in other research domains [20], the hierarchical learning structure of CNNs is beneficial for TL in time series. In our cases, the best PV results are when considering the embedding vector solely at the first residual block. It is probably sufficient since the network can bypass information to later layers, without losing information, due to the residual block. However, for the more non-linear problem of wind power forecasts, it is also beneficial to consider the encoded task ID in all residual blocks.

## 5 Conclusion and Future Work

We successfully showed the applicability of the proposed task TCN for wind and PV day-ahead power forecasts. The proposed architecture provides the possibility to solve critical real-world challenges in renewable power forecasts. It provides a framework for MTL, inductive TL, and zero-shot learning and

**Table 4.** Mean nRMSE for spring of pv data set, cf. with Table 3.

| Model | Embedding Type | Embedding Position | DaysTraining Embedding Initialization | nRMSE 7 | 14 | 30 | 60 | 90 |
|---|---|---|---|---|---|---|---|---|
| MLP | Bayes | First | Copy (Ref.) | 0.305 | 0.204 | 0.167 | 0.110 | 0.116 |
| | | | Default (BS) | 0.305 | 0.204 | 0.167 | 0.110 | 0.116 |
| | Normal | First | Copy (Ref.) | 0.170* | 0.192 | 0.096* | 0.086* | 0.089* |
| | | | Default (Ref.) | 0.263 | 0.199 | 0.132 | 0.093* | 0.097* |
| TCN | Bayes | All | Copy (Ref.) | **0.094*** | 0.100* | 0.097* | 0.083* | 0.084* |
| | | | Default (Ref.) | 0.095* | 0.098* | 0.097* | 0.083* | 0.084* |
| | | First | Copy (Ref.) | 0.106* | 0.104* | 0.091* | 0.084* | 0.084* |
| | | | Default (Ref.) | 0.106* | 0.104* | 0.092* | 0.084* | 0.084* |
| | Normal | All | Copy (Ref.) | 0.096* | 0.083* | 0.078* | 0.076* | **0.077*** |
| | | | Default (Ref.) | 0.106* | 0.105* | 0.117 | 0.079* | 0.078* |
| | | First | Copy (Ref.) | 0.106* | **0.081*** | **0.076*** | **0.076*** | 0.078* |
| | | | Default (Ref.) | 0.117* | 0.090* | 0.099* | 0.078* | 0.080* |

improves the forecast error significantly compared to the Bayesian MLP task embedding as the baseline in all three domains. However, some results of the Bayesian architecture for the inductive TL are contradictory. Those contradictions are probably caused by insufficient training epochs to reduce the ELBO that should be investigated in future work. The most intriguing result for future work concerns the zero-shot learning experiment as the proposed TCN architecture achieves at least a similar low forecast error, without training data, as the Bayesian MLP with a full year of training data. Even though we achieve excellent results for the PV and wind dataset, the wind dataset led to some outliers that can be considered by taking additional meta-information into account in future work.

**Acknowledgments** This work results from the project TRANSFER (01IS20020B) funded by BMBF (German Federal Ministry of Education and Research).

## References

1. D. Gielen, F. Boshell, D. Saygin, et al. The role of renewable energy in the global energy transformation. *Energy Strategy Reviews*, 24:38–50, 2019.
2. R. Schwartz, J. Dodge, N. A. Smith, et al. Green AI. *CoRR*, pages 1–12, 2019. arXiv: 1907.10597.
3. J. Schreiber and B. Sick. Emerging Relation Network and Task Embedding for Multi-Task Regression Problems. In *ICPR*, 2020.
4. S. Vogt, A. Braun, J. Dobschinski, et al. Wind Power Forecasting Based on Deep Neural Networks and Transfer Learning. In *18th Wind Integration Workshop*, page 8, 2019.
5. J. Schreiber, A. Buschin, and B. Sick. Influences in Forecast Errors for Wind and Photovoltaic Power: A Study on Machine Learning Models. In *INFORMATIK 2019*, pages 585–598. Gesellschaft für Informatik e.V., 2019.

6. M. Solas, N. Cepeda, and J. L. Viegas. Convolutional Neural Network for Short-term Wind Power Forecasting. In *Proc. of the ISGT-Europe 2019*, 2019.
7. Z. Fuzhen, Q. Zhiyuan, D. Keyu, et al. A comprehensive survey on transfer learning. *Proc. of the IEEE*, 109(1):43–76, 2021.
8. Yu Zhang and Qiang Yang. A Survey on Multi-Task Learning. *IEEE TKDE (Early Access)*, pages 1–20, 2021.
9. T. Shireen, C. Shao, H. Wang, et al. Iterative multi-task learning for time-series modeling of solar panel PV outputs. *Applied Energy*, 212:654–662, 2018.
10. S. Tasnim, A. Rahman, A. M. T. Oo, et al. Wind power prediction in new stations based on knowledge of existing Stations: A cluster based multi source domain adaptation approach. *Knowledge-Based Systems*, 145:15–24, 2018.
11. L. Cao, L. Wang, C. Huang, et al. A Transfer Learning Strategy for Short-term Wind Power Forecasting. In *Chinese Automation Congress*, pages 3070–3075. IEEE, 2018.
12. L. Cai, J. Gu, J. Ma, et al. Probabilistic wind power forecasting approach via instance-based transfer learning embedded gradient boosting decision trees. *Energies*, 12(1):159, 2019.
13. A. S. Qureshi and A. Khan. Adaptive transfer learning in deep neural networks: Wind power prediction using knowledge transfer from region to region and between different task domains. *Computational Intelligence*, 35(4):1088–1112, 2019.
14. X. Liu, Z. Cao, and Z. Zhang. Short-term predictions of multiple wind turbine power outputs based on deep neural networks with transfer learning. *Energy*, 217:119356, 2021.
15. Y. Ju, J. Li, and G. Sun. Ultra-Short-Term Photovoltaic Power Prediction Based on Self-Attention Mechanism and Multi-Task Learning. *IEEE Access*, 8:44821–44829, 2020.
16. S. Zhou, L. Zhou, M. Mao, et al. Transfer learning for photovoltaic power forecasting with long short-term memory neural network. In *Proc. of the BigComp 2020*, pages 125–132, 2020.
17. H. Zang, L. Cheng, T. Ding, et al. Day-ahead photovoltaic power forecasting approach based on deep convolutional neural networks and meta learning. *Int. JEPE*, 118:105790, 2020.
18. Tomas Mikolov, Kai Chen, Greg Corrado, et al. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*, pages 3111–3119, 2013.
19. C. Guo and F. Berkhahn. Entity Embeddings of Categorical Variables. *CoRR*, pages 1–9, 2016. arXiv: 1604.06737.
20. H. I. Fawaz, G. Forestier, J. Weber, et al. Transfer learning for time series classification. In *2018 IEEE BigData*, pages 1367–1376, 2019.
21. Jining Yan, Lin Mu, Lizhe Wang, Rajiv Ranjan, and Albert Y. Zomaya. Temporal Convolutional Networks for the Advance Prediction of ENSO. *Scientific Reports*, 10(1), 2020.
22. J. Reis and G. Gonçalves. Hyper-Process Model: A Zero-Shot Learning algorithm for Regression Problems based on Shape Analysis. *JMLR*, 1:1–36, oct 2018.
23. C. Blundell, J. Cornebise, K. Kavukcuoglu, et al. Weight Uncertainty in Neural Networks. In *32nd ICML 2015*, volume 37, pages 1613–1622, 2015.
24. J. Schreiber, M. Siefert, K. Winter, et al. Prophesy: Prognoseunsicherheiten von Windenergie und Photovoltaik in zukünftigen Stromversorgungssystemen. *German National Library of Science and Technology*, page 159, 2020.
25. European centre for medium-range weather forecasts. `http://www.ecmwf.int/`, 2020. [Online; accessed 2021-03-30].