Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Continuous Learning of Deep Neural Networks to Improve Forecasts for Regional Energy Markets

Yujiang He * **Janosch Henze** * **Bernhard Sick** *

* *University of Kassel, Kassel, Germany (e-mail: {yujiang.he,jhenze,bsick}@uni-kassel.de).*

Abstract: Germany generated 54.5% of electricity from renewable energy in March 2019, according to the data collected by the Fraunhofer Institute for Solar Energy Systems. Forecasting power generation and consumption play an essential role in establishing a regional smart energy market. Numerous researches contributed to the field of power forecasting using machine learning and deep learning technologies. However, developing and perfecting energy markets lead to an unavoidable problem of adjusting the architectures of neural networks to adapt to new situations, e.g., new consumers or producers in the power grid. Another critical challenge is to learn new knowledge from the sequentially collected measurements efficiently and how to integrate the new information into the current neural network model. When retrained for a new task with a regular training process, neural network models could perform poorly on the previously learned tasks, which is referred to as the catastrophic forgetting problem. In this article, we design two real-world continuous learning scenarios for those challenges. The scenarios are based on the historical power data, which are obtained from a regional power grid in Germany. The results show that well-known continuous learning algorithms can be used to improve power forecasts with a sequential data stream in such scenarios. We believe that the work is the first step towards establishing an adaptively updating forecast system to assist the highly dynamic intelligent energy markets.

Keywords: Smart grids; Intelligent control of power systems; Modeling and simulation of power systems

1. INTRODUCTION

Germany, as well as other countries, integrates more and more renewable energy sources in their energy mix. Such energy sources are more decentralized than common power plants. This decentralization allows for more regional energy usage. Regional energy usage can put more strain on the local power grids, hence causing faulty power grid states. A mechanism to overcome these problems is regional energy markets, as introduced in Alanne and Saari [2006], which means the regional re-allocation of personal responsibility, ownership, expertise, and decision in relation to energy supply. In these regional energy markets, power consumption and generation forecasts are used to obtain information about the future state of the power grid. With knowledge about emerging problems, regional energy markets can employ market mechanisms to control the power grid before problems occur. Hence, forecasting power generation and consumption is a crucial step in establishing intelligent regional energy markets.

Deep learning refers to a method of machine learning, which applies artificial neural networks (ANNs) with several hidden layers between the input layer and output layer, thus forming an extensive internal structure, see in LeCun et al. [2015]. Deep learning structures, such as deep neural network (DNN), auto-encoder (AE), and recurrent neural networks (RNNs), have been proven to be good at forecasting power consumption and generation, see Gensler et al. [2017], Henze et al. [2020], and Schreiber et al. [2019]. Current applications often use models that are only trained on historical data of, e.g., a wind power plant, a specific household, or a small company. Hence, they learn behavior that might already be outdated. Even updates to the training data sets often range several years back, are collected every year for the upcoming billing cycle, or might not even be available anymore due to data protection laws. Another driver of regional energy markets is smart meters. These smart meters collect information about power consumption or generation of, e.g., a household or a solar plant. The data is collected from a remote location over the internet. The collected data contains more accurate and up-to-date information about the power grid. The continuous flow of the smart meter data, in combination with continuous learning paradigms for neural networks, can overcome the previously stated problems of outdated training data of new consumers, producers, or varying grid configurations. Well-known continuous learning algorithms, such as Learning without Forgetting (LWF), Elastic Weight Consolidation (EWC), Online-EWC, or Synaptic Intelligence (SI), are ideal for training neural networks with a continuous and changing data stream. By applying these algorithms, the neural network can adapt to new situations that are only sparsely or not even at all available in the historical non-data-stream based training data sets.

In this work, we design two scenarios in regional smart energy markets where catastrophic forgetting can occur in real-world applications. Catastrophic forgetting happens if a neural network is trained with new information. In such cases, neural networks can abruptly forget previously learned information, as explained by McCloskey and Cohen [1989]. In the first scenario, the *Task-Domain Incremental* (Task-DI) scenario, due to developing and perfecting the energy market, new and previously unmeasured energy units (consumers or generators) have to be integrated into the energy market as new forecasting tasks. These tasks require to adjust the architecture and the weights of the current neural network model to the new forecast targets. The second scenario, the *Data-Domain Incremental* (Data-DI) scenario, helps to improve already trained models to adapt to new situations, previously not occurring in the data. In the Data-DI scenario, changes only occur in the distributions of the data and not in the architecture of the power grid. In such a case, the model architecture is maintained, but the model input and output distributions are changing.

Based on these scenarios, we design the corresponding experiments with continuous learning algorithms and show how we can improve the ability of the neural network models to learn and solve sequential tasks in both scenarios. With the help of our methods, the updated neural network models can integrate new data more adaptively, seamlessly, and flexibly into the continuously changing and developing world of smart grids. Hence, it allows for regional energy markets to continuously learn and adapt to new power plants and integrate data from new smart meters more seamlessly. The experiments are run on data from a small power grid, with the goal in mind to allow for local redistribution of energy. The findings of this article can be applied to any level of the power grid, where catastrophic forgetting can occur.

The main contributions of this article are briefly summarized as follows:

- We point out two continuous learning scenarios, Task-DI and Data-DI, in the context of smart energy markets.
- We review some popular continuous learning approaches (LWF, EWC, Online-EWC, and SI), and implement them in the two scenarios based on a real-world power dataset.
- We compare their performance with baseline experiments evaluated on the average MSE, the training time, and the forgetting ratio.

The remainder of this article starts with a literature review in Section 2 and gives an overview of previous work in this area, with a focus on smart meter applications of continuous learning in smart grids. Afterward, in Section 3 we introduce several continuous learning algorithms. The algorithm introduction is followed by a presentation of the continual learning scenarios we address in our work in Section 4. In Section 5 we describe our dataset and how we set up our experiments. The experimental results with different continuous learning approaches follow in Section 6. The article closes by providing a conclusion of our findings in Section 7, giving implications for future application of continuous learning in the field of smart grids and regional energy markets.

2. LITERATURE REVIEW

In this section, we review relevant literature on artificial neural networks, smart grids, and continuous learning, setting the context of our work compared to other research in the field.

The review by Raza and Khosravi [2015] gives a good overview of the state of load demand forecasting. They discuss artificial neural networks with a focus on integration into smart grids, and strategies to improve the whole data mining process that leads to a load demand forecast. According to the authors, new and improved learning paradigms and training algorithms have to be created to deal with the structure and availability of data in smart grids.

Ziekow et al. [2013] highlights the importance of using smart meter data in household load forecasting to obtain disaggregated load data over high time resolutions. Those data allowed them to improve demand forecasting. With their experiments, they have identified that additional in-home sensors are more useful to forecast demand than higher time resolutions.

In Singh and Yassine [2018], a database is used to maintain patterns of smart meter data to identify appliance usage of households. The patterns are mined every 24 hours from smart meter data using frequent pattern mining and clustering algorithms.

In Cárdenas et al. [2012], a modeling approach is proposed, which helps to model energy consumption in intelligent energy management systems. The authors use an Adaptive Network Fuzzy Interference System (ANFIS) according to Jang [1993] and a five-step modeling approach to continuously update their machine learning model. The five steps are data selection, data preprocessing, ANFIS configuration, training, and model update. A new model replaces the current model if the new model is better than a reference model, e.g., the currently used one. Newly arriving data from measurement infrastructure is stored in a database until a new training cycle starts.

Huang and Liu [2013] shows another approach to model residential energy use. They employ single critic neural networks with a continuous self-learning approach. Their approach mainly uses adaptive dynamic programming and multi-layer perceptrons to schedule battery usage based on the modeled residential load.

Another approach to continuous learning is presented in Bernecker et al. [2014]. The authors use sky images to predict short term cloud movements. Their model uses a Kalman filter to integrate previous forecasts of cloud movements into the current forecast. Therefore, the authors are able to forecast cloud movements in sub 3 minutes intervals and up to 10-minute intervals.

In Vrablecová et al. [2018], the authors analyze different approaches for online load forecasting in smart grids to their approach, which uses an online support vector regression. Furthermore, they highlight the problem of batch learning methods, which often fail to learn from a continuous flow of data.

Other approaches to continuous learning in the energy domain include Peng et al. [1990] or Haupt and Kosovic [2015]. Peng et al. [1990] used very simple retraining after one day of data collection. In Haupt and Kosovic [2015] also a continuous retraining approach is chosen to update their forecasting models.

Parisi et al. [2019] provides a more general view on continuous learning, with a focus on neural networks. The authors summarize the challenges for artificial neural networks and compare approaches to alleviate these challenges. Different learning paradigms, such as lifelong learning, transfer learning, or multisensory learning, are discussed.

Another more general approach to continuous learning, called incremental learning is discussed in Gepperth and Hammer [2016]. They introduce incremental learning methods and several challenges that occur during incremental learning. They end their overview by providing several general application fields, e.g., big data, robotics, or anomaly detection.

Our work tries to use paradigms mentioned in Parisi et al. [2019] to overcome the problem of catastrophic forgetting described by McCloskey and Cohen [1989], e.g., due to complete retraining of the neural network with the data. To do this, we design the two different scenarios, where either the available data change or the number of forecasting targets change. With the applied algorithms, we aim to show that no complete retraining of the neural network models is needed but that the neural networks can be adapted to newly available data without losing the information of previous training.

3. ALGORITHMS

Various continuous learning algorithms, especially for the deep neural network architectures, were proposed in the past years. However, currently, there are no common terminology and clear objectives in the research community. Maltoni and Lomonaco [2019] described a three-way fuzzy categorization of the most common continuous learning strategies:

- architectrual strategies,
- regularization strategies, and
- rehearsal strategies.

In this article, we focus on the regularization strategies, in which the loss function is extended with loss terms of the MSE for the current task and the weighted regularization loss for previous tasks. Four algorithms are compared in the experiments for the two scenarios:

Elastic Weight Consolidation (EWC): EWC was proposed by Kirkpatrick et al. [2017]. The main idea is to control forgetting by penalizing moving weights which are important for previous tasks. The total loss function in EWC is:

$$\mathcal{L}_{ewc}(\mathbf{\Theta}) = \mathcal{L}_{mse}\left(y(\mathbf{\Theta})^K - \hat{y}^K\right) + \sum_{k=1}^{K-1}\left(\sum_{i=1}^{N_{params}} \frac{\lambda}{2} F_{ii}^k \left(\Theta_i - \hat{\Theta}_i^{(k)}\right)^2\right), \quad (1)$$

where $K-1$ is the number of previous tasks and the hyperparameter $\frac{\lambda}{2}$ is the regularization strength. $y(\mathbf{\Theta})^K$ denotes the forecast of the current task K and the $\hat{y}^K$ represent the true measurement. Θ_i is the i_{th} parameter optimized after training on one batch of current task K. $\hat{\Theta}_i^{(k)}$ is the i_{th} parameter after training on the k_{th} previous task. The i_{th} diagonal element of a Fisher information matrix $F_{ii}^{(k)}$ can be calculated as the variance of $\frac{\partial \mathcal{L}_{mse}\left(y(\mathbf{\Theta})^k - \hat{y}^k\right)}{\partial \Theta_i}$ over all training samples of k_{th} task .

Online-EWC: Online-EWC is a variant of EWC that was proposed by Schwarz et al. [2018]. It overcomes the disadvantage of EWC in which the number of quadratic terms in regularization terms grows linearly with the number of tasks. The total loss function of Online-EWC is given by:

$$\mathcal{L}_{o-ewc}(\mathbf{\Theta}) = \mathcal{L}_{mse}\left(y(\mathbf{\Theta})^K - \hat{y}^K\right) + \frac{\lambda}{2}\sum_{i=1}^{N_{params}} \tilde{F}_{ii}^{(K-1)} \left(\Theta_i - \hat{\Theta}_i^{(K-1)}\right)^2, \quad (2)$$

where $\tilde{F}_{ii}^{(K)} = \gamma \tilde{F}_{ii}^{(K-1)} + F_{ii}^{(K)}$, with $\tilde{F}_{ii}^{(1)} = F_{ii}^{(1)}$. The meanings of the other symbols are the same as described above. The hyperparameter $\gamma \leq 1$ governs a gradual decay of each previous task's contribution. If $\gamma < 1$, there is an interesting side effect, the effects of the previous tasks will be explicitly forgotten in a graceful and controlled manner. It could be useful if the learning has not converged on an older task. In our work, the γ is set to 1, which denotes the effects of all previous tasks contribute equally to learn newer tasks.

Compared to EWC, the storage of F and $\hat{\Theta}$ requires $2 \times N_{params}$ values, where N_{params} is the number of model parameters.

Synaptic Intelligence (SI): SI was proposed by Zenke et al. [2017] to calculate the importance of parameters online during training. Compared to EWC, the computational and storing overhead of SI is lower. The loss function of SI is given as follows:

$$\mathcal{L}_{si}(\mathbf{\Theta}) = \mathcal{L}_{mse}\left(y(\mathbf{\Theta})^K - \hat{y}^K\right) + c\sum_{i=1}^{N_{params}} \Omega_i^{K-1} \left(\Theta_i - \hat{\Theta}_i^{(K-1)}\right)^2, \quad (3)$$

where the regularization strength c is a hyperparameter, which is typically set equal to or smaller than 1 to correspond to a weighting of old and new memories. The Ω_i^{K-1} estimates the importance of the i_{th} parameter for the first $K-1$ tasks:

$$\Omega_i^{K-1} = \sum_{k=1}^{K-1} \frac{\omega_i^k}{\left(\Delta_i^{(k)}\right)^2 + \xi}, \quad (4)$$

with $\Delta_i^{(k)} = \Theta_i\left[N_{iters}^{(k)}\right] - \Theta_i\left[0^{(k)}\right]$, where $\Theta_i\left[0^{(k)}\right]$ indicates the initialized value of parameter i before training on task k starts. The importance of a parameter Θ_i for a specific task depends on two quantities:

(1) $\omega_i^{(k)}$ denotes how much an individual parameter contributed to a drop in the loss, and
(2) $\Delta_i^{(k)}$ denotes how far it moved.

ω_i^k can be further interpreted as the per-parameter contribution to changes in the total loss for i_{th} parameter in every new task k. ξ is a dampening term and usually set to 0.1.

$$\omega_i^{(k)} = \sum_{t=1}^{N_{iters}} \left(\Theta_i \left[t^{(k)}\right] - \Theta_i \left[(t-1)^{(k)}\right]\right) \frac{-\partial \mathcal{L}_{total}[t^{(k)}]}{\partial \Theta_i}, \quad (5)$$

with N_{iters} being the total number of iterations per task. The $\omega_i^{(k)}$ are updated continuously during training, whereas the cumulative importance measures, Ω_i^{K-1}, and the reference weights, $\hat{\Theta}_i^{K-1}$, are only updated at the end of each task.

Learning Without Forgetting (LWF): This is a regularization approach proposed by Li and Hoiem [2017]. The input data of the current task is used not only to train the neural network for solving the current task but also to replay the forecasts from the outputs of the previous tasks. The original loss function declared by Li and Hoiem [2017] is intended to be used for classification problems:

$$\mathcal{L}_{lwf}(\mathbf{\Theta}) = (1-\lambda)\mathcal{L}_{cross}\left(y(\mathbf{\Theta})^K - \hat{y}^K\right) + \frac{\lambda}{K-1}\sum_{k=1}^{K-1}\mathcal{L}_{kdl}\left(y(\mathbf{\Theta})^k - \hat{y}(\mathbf{\Theta}_{old})^k\right). \quad (6)$$

The first term $\mathcal{L}_{cross}\left(y(\mathbf{\Theta})^K - \hat{y}^K\right)$ is a regular cross-entropy loss function to adjust the model to the current task K. The second term $\mathcal{L}_{kdl}\left(y(\mathbf{\Theta})^k - \hat{y}(\mathbf{\Theta}_{old})^k\right)$ is a Knowledge Distillation Loss proposed by Hinton et al. [2015]. During learning the current task K, before starting training one batch, the batch is used to generate forecasts from the previous $K-1$ outputs. Each output is symbolized as $y(\mathbf{\Theta}_{old})^k$ (also called soft target). The range of k is from 1 to $K-1$, the same below. After training the batch in current task K, the forecasts from the previous $K-1$ outputs are generated again on the batch, and symbolized as $y(\mathbf{\Theta})^k$. The $\mathcal{L}_{kdl}$ aims to calculate the loss between $y(\mathbf{\Theta})^k$ and $y(\mathbf{\Theta}_{old})^k$. Minimizing the $\mathcal{L}_{kdl}$ attempts to preserve the stability of outputs on old tasks. Li and Hoiem [2017] mentions that Knowledge Distillation Loss can be replaced by other reasonable losses.

In our work, we focus on solving regression problems. Thus, we replace the cross-entropy loss and the distillation loss with Mean-Square-Loss (MSE). Hence, the LWF total loss function is implemented as follows:

$$\mathcal{L}_{lwf}(\mathbf{\Theta}) = (1-\lambda)\mathcal{L}_{mse}\left(y\left(\mathbf{\Theta}\right)^K - \hat{y}^K\right) + \frac{\lambda}{K-1}\sum_{k=1}^{K-1}\mathcal{L}_{mse}\left(y\left(\mathbf{\Theta}\right)^k - \hat{y}\left(\mathbf{\Theta}_{old}\right)^k\right). \quad (7)$$

Baseline: Additionally, there will be a set of baseline experiments, in which the models are trained on all tasks consecutively. The baseline experiments will be using fine-tuning only in a general supervised-learning setting. It means the model, which is trained on the previous $K-1$ tasks, will be directly trained on the current K task. The loss function of the baseline experiments is:

$$\mathcal{L}_{baseline}(\mathbf{\Theta}) = \mathcal{L}_{mse}\left(y\left(\mathbf{\Theta}\right)^K - \hat{y}^K\right). \quad (8)$$

4. CONTINUAL LEARNING SCENARIOS

We consider models that can learn several tasks sequentially and perform well on previous and current tasks without storing and recalling the observed data. In van de Ven and Tolias [2018], different distinct scenarios are designed to quickly compare the performances of numerous methods for alleviating catastrophic forgetting. Two distinct scenarios for continual learning occur in real-world applications, i.e., task-domain and data-domain incremental scenarios. These two different scenarios help to make comparisons among different algorithms easier and more interpretable, by looking at two different continuous learning paradigms.

In the first scenario, which we refer to as *task-domain incremental* (Task-DI), new energy generators or consumers will join the forecasting list, thus adding a new forecast task, i.e., the neural network gets extended. The multi-output neural network model retains its current network structure to predict the current targets, and the new output units are added to the model to account for the new targets. In a real-world application, the number of devices will increase as the energy market extends. In this context, we gradually expand the structure of the neural network to increase the performance of the old and new tasks. The decremental case is not considered in this work. However, it can be handled similarly, for example, by deactivating the outputs where power units need to be removed.

In the second scenario, which we refer to as *data-domain incremental* (Data-DI), the number of the targets is fixed, and only the distribution of the data is changing. The neural network model does not have to adapt the architecture. Instead, the model is required to incrementally learn new knowledge from a continuous input data stream by adjusting to the change in the data distribution of particular forecasting tasks. In this context, the neural network aims to continuously learn to forecast unexpected and unexperienced relationships between weather features and energy generation or consumption without reusing historical data. Such scenarios occur if the residents of a household change, or a business change from being a restaurant to being a supermarket.

5. EXPERIMENT

This section will go into detail about our performed experiments, detailing our data, experiment parameters, and experimental results.

5.1 Data

All experiments in this article are conducted on a dataset that consists of one-year power information for a regional power grid in central Germany. The dataset includes the following information:

- numerical weather prediction (NWP) data ,
- power data (generation and consumption).

The raw NWP dataset contains the historic regional NWP data for one year in a six-hour temporal resolution. 27 weather features are available at each time step and the corresponding 180-hour forecast values starting from the measurement time point in a 3-hour resolution. Through shifting and interpolating, we obtain NWP data in a 15-minute resolution, which is the same temporal resolution as of the power data. The data is normalized between 0 and 1 using min-max normalization.

The power generation of 10 renewable energy generators and the power consumption of 10 renewable local consumer plants for one year in a 15-minute resolution are selected from the raw power data set as the targets for load regression forecasts. The 10 renewable energy generators consist of 3 wind parks, 6 photovoltaic generators, and 1

Table 1. Architecture parameters of the auto-encoder and the multi-layer perceptron. Dropout layers are used in the auto-encoder (from Encoder 1 to Encoder 4) with a dropout rate 0.1 and in multi-layer perceptron (from Layer 1 to Layer 3) with a dropout rate 0.15.

Auto Encoder		
Layer	Size	Activation
Input	1×27	None
Encoder 1	27×19	Leaky ReLu
Encoder 2	19×15	Leaky ReLu
Encoder 3	15×12	Leaky ReLu
Encoder 4	12×8	Leaky ReLu
Decoder 1	8×12	Leaky ReLu
Decoder 2	12×15	Leaky ReLu
Decoder 3	15×19	Leaky ReLu
Output	19×27	None
Forecasting layer		
Layer	Size	Activation
Input	1×18	None
Layer 1	18×50	ReLu
Layer 2	50×50	ReLu
Layer 3 (Output)	50×20	None

Table 2. Hyperparameters for the training and continuous learning algorithms in the experiments.

Training			
Batch Size	Iteration	Optimzier	Learning Rate
64	2500	Adam	0.001
LWF		**SI**	
Target	Loss	Parameter	Value
Previous	MSE	c	0.1
Current	MSE	ξ	0.1
EWC		**Online-EWC**	
Parameter	Value	Parameter	Value
λ	1000	λ	1000
sample size	5000	*sample size*	5000
		γ	1.0

combined heat and power plant. The produced power of each generator is normalized using the rated capacity of the corresponding facility. The power consumption measurements are normalized between 0 and 1 using min-max normalization.

5.2 Method

The same neural network structure and pre-processing methods are used for comparing the performance among the different algorithms. A multi-layer auto-encoder is used to reduce the dimensionality of the NWP data. An auto-encoder (AE) is a multi-layer symmetrical neural network with a small bottleneck layer at the center. The high-dimensional input features are compressed in the first part of the AE, the encoder, and can be reconstructed with the second part of the AE, the decoder. More details of how we applied the auto-encoders can be found in Henze et al. [2020]. An auto-encoder encodes the NWP data at each time step with a 4-layer encoder using a LeakyReLU with a slope parameter of 0.3 in all hidden layers. There is a dropout layer with a dropout rate of 0.15 between a pair of encoder layers to avoid over-fitting. Afterward, the encoded features are concatenated with 5 temporal features, i.e., month of the year, day of the year, week of the year, day of the week, and hour of the day. The temporal features are normalized between -1 and 1 using sine/cosine coding, resulting in 10 encoded temporal features.

In the Task-DI scenario, we use a model with multiple outputs to run all experiments. Every energy generator or consumer corresponds to an output unit, which is active only when the respective task is under consideration during training, either the current task or the replayed task in LWF experiments.

In the Data-DI scenario, the architecture of the model is similar to the model in the Task-DI scenario. Multiple output units are corresponding to each energy generator or consumer. The one-year dataset is split into n sections with the same number of samples. n sections, namely n tasks, share the same outputs.

The neural network parameters, training parameters, and the parameters of the continuous learning algorithms are listed in Table 1 and Table 2. The main focus of this article is on comparing the performance of the different continuous learning algorithms. Therefore, the neural network parameters are selected empirically and kept the same in all experiments. More details about the parameter selections can be found in our previous work, Henze et al. [2020], and Gensler et al. [2017]. The reduced dimension of the NWP data is 8. The encoded NWP is concatenated with 10 encoded temporal features to form the input of the forecasting layer.

6. RESULTS

The 10 energy generators and 10 energy consumers are the targets. In the Task-DI scenario, 20 targets are seen as training tasks. In the Data-DI scenario, the dataset is split into 4 sections to create 4 tasks that are trained consecutively. A task is a section including 3 months of historical data for 20 targets. In both scenarios, each task is evaluated on test datasets twice. The first time is after training the current task. The evaluation is based only on the test dataset of the current task and named *During* in the tables below, which shows how well it learns from the current task. The second is after training all tasks. The evaluation is conducted on the test dataset for all tasks and is named *After* in the tables below. Additionally, the tables show how much information for each experienced task the model retains.

6.1 Fixed task number

Each experiment is run 20 times with different random seeds and evaluated using MSE. The average test errors are listed in Table 3. Compared to the baseline experiment, the test errors of the continuous learning algorithms after training all tasks are lower than of the baseline experiments. It indicates that the four continuous learning algorithms ease the forgetting of previous tasks in both scenarios. It should be noted that the test errors of Online-EWC in both scenarios are the lowest. EWC is outperformed only by Online-EWC. More importantly, test errors of continuous learning methods after training all tasks are

Table 3. Average MSE loss and standard deviation on the test datasets over all tasks in two scenarios.

Algorithm	Task-DI	
	During	After
Baseline	0.01923 (±0.01891)	0.04466 (±0.04215)
LWF	0.01836 (±0.01647)	0.03160 (±0.02278)
EWC	0.02100 (±0.01688)	0.02137 (±0.01704)
O-EWC	0.02052 (±0.01664)	**0.02066 (±0.01655)**
SI	0.01828 (±0.01694)	0.02446 (±0.01909)

Algorithm	Data-DI	
	During	After
Baseline	0.02712 (±0.01095)	0.02964 (±0.01656)
LWF	0.02982 (±0.00988)	0.02479 (±0.00538)
EWC	0.02513 (±0.01175)	0.02053 (±0.00382)
O-EWC	0.02513 (±0.01176)	**0.02049 (±0.00377)**
SI	0.02591 (±0.01144)	0.02555 (±0.01027)

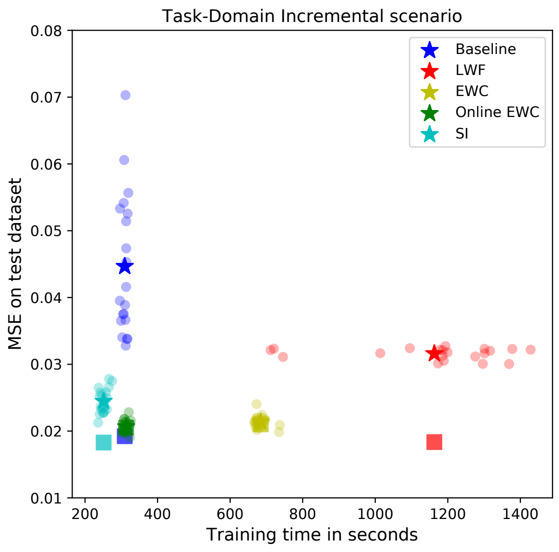

Fig. 1. Comparsion of training time and the average MSE on test datasets over 20 targets among algorithms in the task-domain scenario. The dots indicate test errors after training all tasks at individual runs. Squares indicate the average test error for each task during training, and stars indicate the average test error for each task after training all tasks.

lower than during training in Data-DI scenarios, which denotes the model performs well on new tasks without forgetting the previous tasks, and at the same time, the model obtains new knowledge from the new tasks, further optimizing its weights. That is, the forecast performance is improved on all seen tasks through the proposed methods in this case. LWF outperforms SI in the Data-DI scenario but is outperformed by SI in the Task-DI scenario.

In a real-world application, not only the forecast accuracy but also the computational efficiency of algorithms should be considered. A comparison of training times of all methods in both scenarios are shown in Figure 1 and 2 (also see the articles of van de Ven and Tolias [2018] and Farquhar and Gal [2018]). In the figures, dots indicate individual runs. The squares indicate the average test error

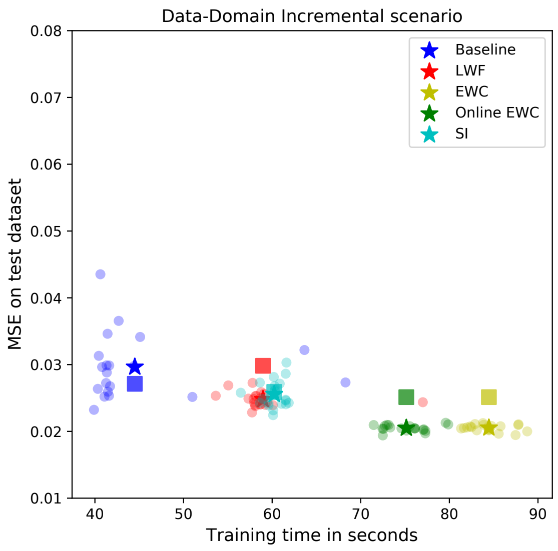

Fig. 2. Similiar to Figure 1, except in the Data-DI scenario, the dataset is split into 4 sections. The dots indicate test errors after training all tasks at individual runs. Squares indicate the average test error for each task during training, and stars indicate the average test error for each task after training all tasks.

Table 4. An example of the difference between average test error and forgetting ratio.

		During	After
Expriment 1	Task 1	0.45	0.46
	Task 2	0.01	0.02
	Average test error		0.24
	Forgetting ratio		0.261
Expriment 2	Task 1	0.45	0.48
	Task 2	0.01	0.03
	Task 3	0.001	0.003
	Average test error		0.173
	Forgetting ratio		1.356

for each task during training, and the stars indicate the average test error for each task after training all tasks. We observe that, compared to EWC, Online-EWC does speed up training in both scenarios. Finally, from the aspect of computational overhead, although LWF fails in the Task-DI scenario. In the Data-DI scenario, it outperforms the other algorithms.

6.2 Increasing task number

Moreover, we also explore the performance of the algorithms as the number of tasks increases, which is hard to be evaluated using the average test error due to the diverse tasks. Let us take the problem shown in Table 4 as an example. In the example, the tasks are tested during training one task and after training all tasks separately, as reported in Table 3. The forgetting on Task 1 and Task 2 is increasing as the number of tasks increases. However, the average test error is dropping, which makes us mistakingly think that the forgetting problem vanishes. To make up for this shortcoming, the forgetting ratio, $Ratio_f$, is proposed with the equation:

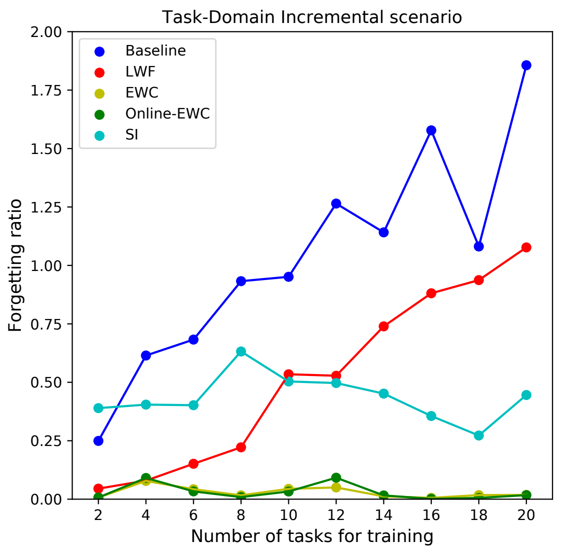

Fig. 3. Performance of algorithms with increasing number of tasks by 2 in each step in the Task-DI scenario (Evaluated for forgetting ratio).

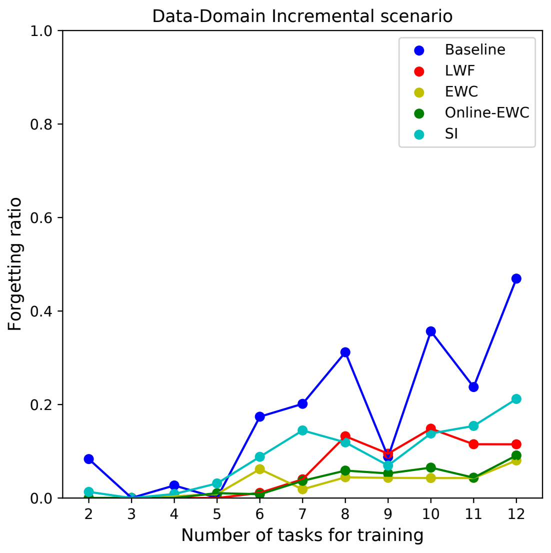

Fig. 4. Same as the Figure 3, except increasing the number of tasks by 1 in the Data-DI scenario.

$$Ratio_f = \frac{1}{K}\sum_{k=1}^{K} \frac{max\left(0, \mathcal{L}_{after}^{k} - \mathcal{L}_{during}^{k}\right)}{\mathcal{L}_{during}^{k}}, \qquad (9)$$

where K is the number of tasks and $max\,(x_1, x_2)$ returns the larger value of either x_1 or x_2. The main idea behind the forgetting ratio is to calculate the percentage of the incremental test error after training all tasks over the test error during training.

From Figures 3 and 4, it can be seen that forgetting is more problematic with increasing tasks in both scenarios. EWC and Online-EWC keep the model performance more stable, as they have a lower forgetting ratio. SI outperforms LWF in the Task-DI scenario.

7. CONCLUSION

We wanted to show how catastrophic forgetting can influence forecasting results in smart grids. To do so we presented two different scenarios where catastrophic forgetting can happen, task domain incremental, and data domain incremental, and showed several methods to overcome these problems. In our experiments, the EWC and Online-EWC algorithm kept a low forgetting ratio, yet depending on the scenario increased the training time compared to other algorithms.

Our work can be seen as a first step to address the ever occuring changes in smart grids due to growing cities, e.g., new houses, or businesses getting introduced to the smart grid, or ownership changes of already occupied buildings ith subsequent behavioral changes. Such changes need to be reflected in the forecasting algorithms, as they are the major driving force behind a functioning intelligent regional energy market. We made a first step to address such changes by providing methods and scenarios which helps to further study catastrophic forgetting in a small controlled environment. Hence, this work could be seen as the starting point for designing a benchmark scenario in the field.

We plan to propose a more general framework with more real-world requirements, which helps others to benchmark their algorithms easily. This allows to establish continuous learning in power forecasting for regional energy markets, enabling forecasting algorithms to adapt to new situations. Furthermore, we need to address decremental scenarios, as we currently only handle addition of new forecasting tasks. Deletion of forecasting task can occur, e.g., due to demolition of buildings, or non-stationary power plants.

ACKNOWLEDGEMENTS

This work was supported within the c/sells RegioFlex-Markt Nordhessen (03SIN119) project, funded by BMWi: Deutsches Bundesministerium für Wirtschaft und Energie/German Federal Ministry for Economic Affairs and Energy.

REFERENCES

Alanne, K. and Saari, A. (2006). Distributed energy generation and sustainable development. *Renewable and sustainable energy reviews*, 10(6), 539–558.

Bernecker, D., Riess, C., Angelopoulou, E., and Hornegger, J. (2014). Continuous short-term irradiance forecasts using sky images. *Solar Energy*, 110, 303–315. doi: 10.1016/J.SOLENER.2014.09.005.

Cárdenas, J.J., Romeral, L., Garcia, A., and Andrade, F. (2012). Load forecasting framework of electricity consumptions for an Intelligent Energy Management System in the user-side. *Expert Systems with Applications*, 39(5), 5557–5565. doi:10.1016/J.ESWA.2011.11.062.

Farquhar, S. and Gal, Y. (2018). Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*.

Gensler, A., Henze, J., Sick, B., and Raabe, N. (2017). Deep Learning for solar power forecasting - An approach using AutoEncoder and LSTM Neural Networks. In *2016 IEEE International Conference on Systems, Man,* *and Cybernetics, SMC 2016 - Conference Proceedings*. doi:10.1109/SMC.2016.7844673.

Gepperth, A. and Hammer, B. (2016). Incremental learning algorithms and applications. In *European Symposium on Artificial Neural Networks (ESANN)*. Bruges, Belgium.

Haupt, S.E. and Kosovic, B. (2015). Big Data and Machine Learning for Applied Weather Forecasts: Forecasting Solar Power for Utility Operations. In *2015 IEEE Symposium Series on Computational Intelligence*, 496–501. IEEE. doi:10.1109/SSCI.2015.79.

Henze, J., Schreiber, J., and Sick, B. (2020). Representation Learning in Power Time Series Forecasting. 67–101. doi:10.1007/978-3-030-31760-7_3.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Huang, T. and Liu, D. (2013). A self-learning scheme for residential energy system control and management. *Neural Computing and Applications*, 22(2), 259–269. doi:10.1007/s00521-011-0711-6.

Jang, J.S. (1993). Anfis: adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics*, 23(3), 665–685.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13), 3521–3526.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436–444.

Li, Z. and Hoiem, D. (2017). Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12), 2935–2947.

Maltoni, D. and Lomonaco, V. (2019). Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116, 56–73.

McCloskey, M. and Cohen, N.J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, 109–165. Elsevier.

Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113, 54–71. doi:10.1016/J.NEUNET.2019.01.012.

Peng, T.M., Hubele, N.F., and Karady, G.G. (1990). Conceptual approach to the application of neural network for short-term load forecasting. In *1990 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2942–2945 vol.4. doi:10.1109/ISCAS.1990.112627.

Raza, M.Q. and Khosravi, A. (2015). A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings. *Renewable and Sustainable Energy Reviews*, 50, 1352–1372. doi: 10.1016/J.RSER.2015.04.065.

Schreiber, J., Jessulat, M., and Sick, B. (2019). Generative adversarial networks for operational scenario planning of renewable energy farms: A study on wind and photovoltaic. In I.V. Tetko, V. Kůrková, P. Karpov, and F. Theis (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2019: Image Processing*, 550–564. Springer International Publishing, Cham.

Schwarz, J., Luketina, J., Czarnecki, W.M., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R., and Hadsell, R. (2018). Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*.

Singh, S. and Yassine, A. (2018). Big Data Mining of Energy Time Series for Behavioral Analytics and Energy Consumption Forecasting. *Energies*, 11(2), 452. doi: 10.3390/en11020452.

van de Ven, G.M. and Tolias, A.S. (2018). Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*.

Vrablecová, P., Ezzeddine, A.B., Rozinajová, V., Šárik, S., and Sangaiah, A.K. (2018). Smart grid load forecasting using online support vector regression. *Computers & Electrical Engineering*, 65, 102–117.

Zenke, F., Poole, B., and Ganguli, S. (2017). Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3987–3995. JMLR. org.

Ziekow, H., Goebel, C., Strüker, J., and Jacobsen, H. (2013). The potential of smart home sensors in forecasting household electricity demand. In *2013 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 229–234. doi: 10.1109/SmartGridComm.2013.6687962.