Dynamic Incremental Learning for real-time disturbance event classification

Veerakumar, Nidarshan; Cremer, Jochen; Popov, M.

DOI
10.1016/j.ijepes.2023.108988

Publication date
2023

Document Version
Final published version

Published in
International Journal of Electrical Power & Energy Systems

Electrical Power and Energy Systems 148 (2023) 108988

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Dynamic Incremental Learning for real-time disturbance event classification

Nidarshan Veerakumar, Jochen L. Cremer, Marjan Popov *

Delft University of Technology, Faculty of EEMCS, Mekelweg 4, Delft, 2628CD, The Netherlands

ARTICLE INFO

Keywords:
Incremental Learning
Rehearsal IL strategies
Replay strategy
Parametric buffer
Reservoir sampling
Real-time model training
Experiences
Event classification
Hyper-parameter tuning
Catastrophic forgetting

ABSTRACT

With recent telemetric advancements, the real-time availability of power grid measurements has opened challenging opportunities for the design of advanced protection and control schemes. Artificial neural networks (ANN) are promising approaches for detecting and classifying disturbance events from measurement data. Numerous offline ANN-based classification algorithms were proposed in the past, which increased the interest for their real-world deployment. However, these algorithms are inadequate due to their conventional offline training procedures, model updating, and large backend computing requirements. Besides, most ANN-based algorithms require disturbance event samples to be collectively available during training. This availability may be uncommon in practice as disturbance events are rare, non-deterministic, and uncertain. Hence, an online training procedure where the model processes the events on-the-fly is required. However, ANNs may also suffer from catastrophic forgetting where the model may unintentionally unlearn an occurred disturbance under the learning of new event types; this means ANN may not detect very similar disturbances of the same type in the future. In this paper, we propose Dynamic Incremental Learning (IL) method for ANN models, which is updated in real-time when a new disturbance is detected. Our proposed method adopts a Replay-based IL strategy for designing long-term IL, balancing the accuracy with catastrophic forgetting of disturbance events. The method is designed in a way to learn efficiently for incoming disturbance data with minimized training time and the highest classification accuracy eliminating catastrophic forgetting. The results describe the methodology's performance regarding classification accuracy, training time, and storage memory. The findings demonstrate that the Dynamic IL method is promising for efficient learning and event classification.

1. Introduction

Decarbonization of the energy systems requires expanding and modernizing the grid infrastructure and other energy sectors by using more renewable energy resources. Therefore, novel technologies like High Voltage Direct Current (HVDC) grids, distributed energy generation, microgrids, offshore energy hubs, and bidirectional grid operation are more and more utilized. There, power electronic devices will become the backbone of the future electrical power systems. Whilst the individual failure rates are low as the power electronics are developed at high standards, the overall occurrence of failures will increase due to the increased complexity of the power systems [1]. The highest impact on power systems is caused by the cascading failures as system operators currently cannot fully predict the consequences upon the occurrence of a contingency. Hence an early (real-time) identification and classification of equipment and system failures causing specific disturbance events are essential for the security of power systems [2].

Most electrical system failures, such as cable joint arcing, capacitor switching restrikes, faults, lightning, and tree contact by a power line, shall produce unique electric signatures. These signatures can be observed from the voltage and current measurement waveforms associated with the equipment/device. The waveform-type data containing voltage and current signatures representing equipment failure or power quality disturbance is called a *disturbance event sample* dataset. Since some of these technologies are not massively deployed, the produced disturbance events' signatures will be yet rather unknown, and developing disturbance identification algorithms through classical, manual methods [3] will be time-consuming and expensive. Classically, collecting event signatures with high resolutions through relays and power quality meters has always existed. These signatures were used for manual root cause analysis only in the case of mal-operation or when a large disturbance (failure) has occurred. Continual data retrieval from remote monitoring sites into a central processing unit for real-time processing was not an option as the telemetric systems were still naive. However, it is desired for every incoming waveform to be automatically identified and tabulated into specific disturbance categories [4]. This automated classification can further facilitate quick corrective measures to prevent significant collateral grid damages.

* Corresponding author.
E-mail address: m.popov@tudelft.nl (M. Popov).

https://doi.org/10.1016/j.ijepes.2023.108988
Received 29 September 2022; Received in revised form 12 December 2022; Accepted 19 January 2023

Nomenclature

Indices

b	mini-batch
d	device
e	epoch
i	sample
j	experience
k	class
t	time

Sets

Ω^B	set of mini-batches occurring in an experience
Ω^D	set of event samples collected from d number of devices
Ω^E	events experience set
Ω^O	evaluation dataset of each event experience set
Ω^S	set of all event samples in a scenario set
Ω^T	training dataset of each Event experience set

Parameters

γ	learning rate reduction factor
ι	number of epochs passed after critical epoch
κ	Memory buffer limit
Ψ	generic performance metric used in incremental learning
τ	length of the recorded time-series of each event sample x_i
ξ	number of epochs to wait before updating learning rate
K	total number of classes in a particular scenario Ω^S
υ	number of measurement variables in each event sample x_i
x_i	disturbance event sample dataset of index i
y_i	unique label of an disturbance event sample i

Variables

Θ	deep learning model parameters (weights and biases)
L	learning rate
p_{max}	maximum number of epochs to train each experience Ω_j^E

Many ANN approaches were proposed for real-time event classification algorithms. With the ongoing advancements in telemetric infrastructure, a large amount of grid data could be seamlessly transferred almost instantly [5]. This has opened many opportunities to explore the design of advanced real-time disturbance event classification algorithms using machine learning (ML) approaches [6–9]. There, ANNs is one type of ML model that is often preferred as ANNs can handle complex disturbance signatures that are sensitive for disturbance classification tasks [10,11]. Within the scope of disturbance classifications, several different ANN-based models can be suitable such as convolutional neural networks (CNN), recurrent neural networks (RNN), and the way they are trained varies, for example, using Autoencoders [12]. In [13], the authors made use of threaded ensembles of Auto-encoders for learning from data streams, whilst in [14] an RNN model is presented that deals with online anomaly detection under the influence of concept drift. In [15], a classical incremental method using ML-models is explained and demonstrated. However, these ML-based methods are not well suited when analyzing complex highly-sampled electrical signals. The pre-requisite for all these ML-based methods to perform real-time classification of disturbance events is to determine a-priori which types of events the system will have to identify. Based on this a-priori information, an underlying model will be designed and further trained using a disturbance event training dataset. In these methods, at the end of the training session, it is expected that the model is trained well enough to yield the desired accuracy concerning the test dataset. Some methods can adapt themselves when the test data has new events which are not part of the training. These methods adapt the number of neurons, and layers, or modify iteratively the training strategy to achieve the desired accuracy. In [16], case studies are summarized comparing ensemble learning with IL methods.

However, these ANN methods have shortcomings in model scalability, manual re- engineering requirements, lack of standardization, and catastrophic forgetting. Due to the lack of model scalability, the model will incorrectly identify the event class when the incoming waveform of the disturbance event does not fall under the existing trained classes. This shortcoming implies that a new event class augmentation over an already trained model of fixed class size is difficult to consider without retraining the model from scratch. Some work has approached this shortcoming by "freezing" the trained model and training a new model for the new event class. In [17–19], the classification section involves the training of a new autoencoder model under the detection of unforeseen/new classes. Once an ensemble of auto-encoders is formed, the output layer is combined from one single event classification problem. Even though these methods pose an interesting argument, all of them suffer constant (manual) re-engineering requirements of the expert system due to multiple models and cannot be made entirely automated. Furthermore, when a new model for a waveform of the existing class needs to be trained, the entire dataset of that particular class and all other classes stored in a database needs to be available hinting toward an expensive (inefficient) memory utilization. Additionally, all the aforementioned event classification algorithms lack standardization, hindering easy real-time implementation, and may suffer from catastrophic forgetting of once observed disturbances [16]. More precisely, catastrophic forgetting is defined as a complete erasure of previously acquired model parameter values that occur when the model is trained incrementally with a new dataset [20]. Therefore, IL is promising to be used with a deep learning (DL) model to address the aforementioned challenges.

This paper investigates the class IL method for event classification which we found particularly promising to address the issue of catastrophic forgetting when new classes (event types) are encountered and need to be trained on-the-fly [21–24].

Class IL can adapt incrementally to new event types, in the literature related to IL called 'experiences', expanding the model's event classification capabilities in an online fashion, without the need to be completely retrained. Class IL strategies can be classified into three groups, namely rehearsal, regularization, and architectural-based strategies [25]. Architectural and regularization-based strategies alter the model layers to adapt to new event types. Rehearsal strategies intelligently replay some data samples from the previous experience during the new experience's training. This rehearsal method retrains the model with selective datasets from all experiences, thus retaining previous knowledge along with training for new event types. However, for real-time applications of IL, the requirements are to balance training duration, computational burden, memory requirements, and strategy complexity [26].

The paper has three primary novel contributions. Firstly, it introduces Incremental Learning (IL) to power systems that train the model "on-the-fly" without requiring complete retraining when a new disturbance event type occurs. Secondly, we compared and analyzed four IL strategies namely, Replay [27], GDumb [28], LWF [29] and, EWC [30] as these strategies have advantages and are successful with class IL. Then, by identifying Replay strategy as the best strategy for our application, the paper proposes a modified Replay strategy that consists of a dynamic memory buffer that can adapt itself based on the incoming event streams. A dynamic memory buffer is a combination of the parametric memory buffer and reservoir sampling that intelligently stores important event sample representatives from previously seen classes. These event sample representatives will be replayed along with newly received streams for model training. This *Replay combined with a dynamic memory buffer* strategy is practical and straightforward to address catastrophic forgetting. However, it has computational training times that are too high, making it unsuitable for real-time studies in the long run. Thirdly, we propose, for the first time, a novel Dynamic IL method to balance the training duration, buffer memory requirements, and classification accuracy for the task of near-real-time event learning. This Dynamic IL method adds an efficient experience training method to the Automated workflow for real-time event classification. Additionally, this paper contributes to an IL-based real-time disturbance event classification system consisting out of

1. an IL strategy to train a model that considers new types of disturbance events in real-time,
2. an algorithm that generates for this training strategy non-repetitive experiences of event types and
3. performance metrics for evaluating the real-time training of these non-repetitive experiences.

The paper is organized as follows: Section 2 deals with an automated workflow of real-time event classification description. Further, the steps required to achieve the classification description through IL are discussed. Section 3 describes the new Dynamic IL method which is built upon a generic IL methodology. In Section 4, we provide a stage-wise analysis of Dynamic IL using various performance metrics like loss, accuracy, training time, and memory utilization. Various formulations are devised to describe our findings in detail. Finally, in Section 5, we provide conclusions and discuss future research goals for an IL-based real-time event classification system.

2. Learning for real-time event classification

2.1. Automated workflow for real-time event classification

The proposed real-time workflow towards automating the event detection and classification in transmission control rooms is shown in Fig. 1. The novel Dynamic Incremental Learner is the green part which requires as a "pre-step", an event detector that records time-series disturbance events from multiple locations of the power system. An event identifier is used to identify if the detected event is known or unknown to the "representative" model. Then, this representative model can be used to classify the event class in real-time. If the event identifier identifies the detected event as a new event type, then the proposed Dynamic IL method is used to expand the model "on-the-fly" to learn this new event. The new event datasets are named as experience dataset Ω^E formed at time t. This updated model forms the new representative model that in the future, can classify this new event data class in real-time. The model M training is supervised, since the classification labels are provided to the model. An IL strategy governs the model adaptation for this new event type without forgetting the knowledge of the previously seen events and addresses in parallel the subject of catastrophic forgetting aiming for high accuracy of previously seen events as well.

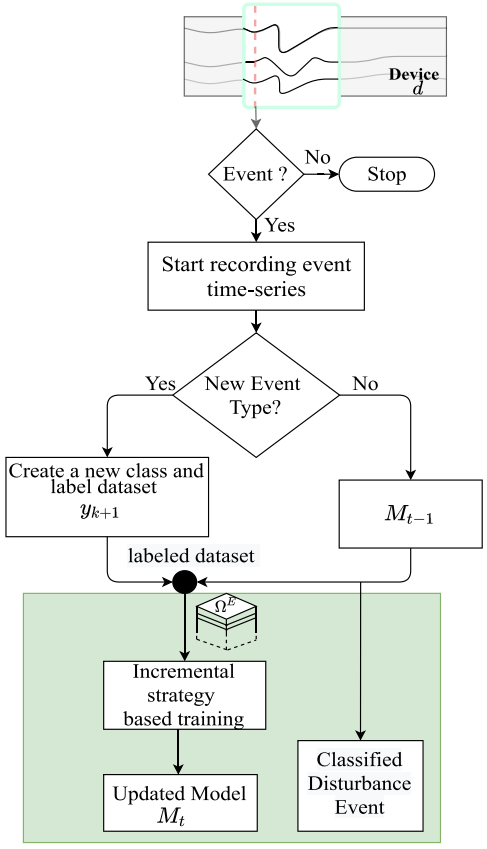

Fig. 1. Real-time workflow for incrementally learning a classifier for new events when they arise.

2.2. Near-real-time incremental learning

The proposed workflow includes three generic parts

- Generating experiences (introduced in Section 2.2.1)
- Incremental model learning strategies (introduced in Section 2.2.2)
- Model evaluation metrics (introduced in Section 2.2.3)

These operations will be explained in detail in the subsequent sections.

2.2.1. Experience generation

We consider the future power grids being fully observable and equipped with a number of devices Ω_D where each device $d \in \Omega_D$ can record data that complies with IEC 61850 protocols and/or IEEE C37-118.1 synchrophasors. If a disturbance event i occurs, the device's $\Omega_i^D \subset \Omega_D$ that are in the area of the event i are detecting the event using an event detecting scheme. Detecting this event i triggers the recording of event's voltage and current signatures at the corresponding devices Ω_i^D. The record of the event i is

$$x_{i,d} \in \mathbb{R}^{\upsilon \times \tau} \quad \forall d \in \Omega_i^D, \tag{1}$$

where τ is the length of the recorded time-series and υ is the number of measurement variables. Then, this disturbance event i with the recorded time-series x_i from affected devices Ω_i^D are transmitted to the control station in real-time. Next, as shown in Fig. 1, an algorithm determines whether the event is of a new type (unknown) or not (known). If the event i is not of a new type (known before), then, a

pre-trained ML-model M assigns an existing class label $\{0, 1, 2, \dots K\}$ to the detected event i by

$$\hat{y}_i \leftarrow M(x_{i,d}), \tag{2}$$

for a randomly selected $d \in \Omega_i^D$. If the assigned class label $\hat{y}_i$ matches the actual class label y_i, then the model prediction of event i is correct and classification accuracy remains high. All recorded time-series data $x_{i,d}$ and their assigned label $\hat{y}_i$ are stored and their unique combinations $\{i, d\}$ of event i and time-series at the devices d are added to the existing *experience event dataset*

$$\Omega^E \leftarrow \{i, d\} \qquad \forall d \in \Omega_i^D. \tag{3}$$

There, the cardinality of the set $|\Omega^E|$ is increasing by $|\Omega_i^D|$ that can differ for any event i.

Challenging is to consider a new event type in the existing event classification model M. If the event is of a new type, then, a new class label $'K+1'$ has to be generated and the number of possible class labels has to be incremented $K \leftarrow K+1$. Then, the new label for this new event is manually assigned by

$$y_i = K + 1. \tag{4}$$

It is not straightforward to change the model M so that in the future events of this new event type are classified correctly. For example, a conventional DL model would misclassify and assigning inaccurately one of the existing event types/classes.

This paper proposes the class incremental learning approach to address this challenge. If there is a new class, then, we start by introducing a new experience j, a new experience event dataset $\Omega_j^E = \{\}$, where we add

$$\Omega_j^E \leftarrow \{i, d\} \qquad \forall d \in \Omega_i^D, \tag{5}$$

and increment the number of classes by

$$K_j = K_{j-1} + 1, \tag{6}$$

where we introduced the index j to denote the maximal number of classes K varies from experience to experience. Eq. (5) replaces Eq. (3). Then, this new experience j is added to the set of experiences $\Omega^S \leftarrow j$. This set Ω^S is also called *class incremental scenario set*.

During real-time model training, the experience dataset from Eq. (5) is split into training $\Omega_j^{E,T}$ and evaluation data $\Omega_j^{E,V}$ with a ratio of r (e.g., typically set at $r = 0.8$) so that

$$\Omega_j^{E,T} \cup \Omega_j^{E,V} = \Omega_j^E. \tag{7}$$

The training set is formed by randomly selecting a temporary subset of event samples $\Omega_j^{E,T} \subset \Omega_j^E$ with cardinality $|\Omega_j^{E,T}| = r|\Omega_j^E|$. The residual is the evaluation data $\Omega_j^{E,V} = \Omega_j^E \backslash \Omega_j^{E,T}$.

2.2.2. Incremental learning strategies

IL aims at efficiently learning a new model M_j predicting K_j classes based on the new experience dataset Ω_j^E and based on the previous model M_{j-1} predicting K_{j-1} classes. Various training strategies exist to avoid forgetting of previous classes (catastrophic forgetting). IL strategies differ in their approaches that are either based on architecture, regularization, or rehearsal [25].

Architecture-based IL strategies: Inspired by the hippo-campus-cortex duality nature of the human brain, dual memory models are developed, where catastrophic forgetting is addressed by modifying layers, freezing certain model parameters θ^{M_j} which can be weights or biases in activation functions. Hence, some model parameters a are frozen and selected from the previous model M_{j-1} for the new trained model M_j so that

$$\theta_a^{M_j} = \theta_a^{M_{j-1}}. \tag{8}$$

As a result, these dynamic architectures can address catastrophic forgetting, however, they quickly result in considerable complex models [31].

Regularization based IL strategies: Regularization techniques can also address catastrophic forgetting by penalizing changes in model parameters (weights and biases) that are strong representatives of past experiences. There, the IL training strategies have the objective to minimize the balance

$$\min \varepsilon + \lambda|\theta^{M_j} - \theta^{M_{j-1}}|, \tag{9}$$

where $\lambda|\theta^{M_j} - \theta^{M_{j-1}}|$ represents the model regularization and ε the supervised training error. There, $|\cdot|$ denotes the absolute value of the deviation of the model parameters and λ is the regularization parameter. Various strategies are devised under this technique that quantify this penalty regularization parameter λ, including synaptic relevance [32], fisher information [30], and uncertainty estimates [33]. These regularization techniques are efficient in training time and memory usage. However, they suffer from brittleness from representation drift [34].

Rehearsal based IL strategies: Rehearsal-based IL strategies train the model $M_j(x_{i,d})$ by 'replaying' some selected event samples $\{i, d\}$ from previous experiences Ω^S in addition to the new experience dataset Ω_j^E. When designing a rehearsal-based IL strategy, the task is to define the function f that selects previous experiences $\hat{j}$

$$\hat{j} \leftarrow f(\Omega^S) \tag{10}$$

so that the model $M_j(x_{i,d})$ is trained considering the training data $\Omega_{\hat{j}}^{E,T}$ of previous experiences $\hat{j}$ in the training loss function, e.g., as the cross-entropy multi-classification loss

$$\min \underbrace{\sum_{\{i,d\} \in \Omega_{\hat{j}}^{E,T}} \sum_{k=0,1,\dots K_{\hat{j}}} Y_{i,k} \log \hat{Y}_{i,k}}_{\varepsilon}. \tag{11}$$

There, $Y_i \in \{0, 1\}^{K_{\hat{j}}}$ is the one-hot encoded actual class label y_i. $Y_{i,k}$ denotes the kth entry of the vector Y_i where exactly one entry is 1, hence, the norm of the vector is $|Y_i| = 1$. Hence, when the class label is $y_i = k$, then the kth entry is $Y_{i,k} = 1$, and all other entries are zero $Y_{i,k} = 0 \quad \forall k \in \{0, \dots K_{\hat{j}}\} \backslash k$. Correspondingly, the one-hot encoded predicted class label from Eq. (1) is $\hat{Y}_i$. The rehearsal-based IL training strategy alternates between Eqs. (10) and (11), where in each alternation the function f selects an experience $\hat{j}$ in Eq. (10) and 'rehearses' (trains the model with) the data from this experience using the supervised loss function defined in Eq. (11). This paper explores rehearsal-based IL strategies in detail and further develops a variant suitable for real-time event learning and classification tasks. There are several methods to design function f that selects the right event samples $\{i, d\}$ from the experiences. These methods can be based on memory reservoirs and pseudo-rehearsal with generative models. The paper then proposed to use replay with dynamic memory buffer strategy as this training strategy is simple and often results in high classification accuracies for class IL scenarios as shown by [25].

2.2.3. Evaluation metrics

A model M_j trained for classification is typically evaluated by the classification accuracy

$$\Psi^{acc} = 1 - \frac{1}{|\Omega_j^{E,V}|} \sum_{\{i,d\} \in \Omega_{\hat{j}}^{E,V}} \sum_{k=0,1,\dots K_j} \frac{|Y_{i,k} - \hat{Y}_{i,k}|}{2} \tag{12}$$

that signifies the model's generalization capabilities. There, $|\cdot|$ in $|\Omega_{\hat{j}}^{E,V}|$ denotes the cardinality of the evaluation dataset $\Omega_{\hat{j}}^{E,V}$ and $|\cdot|$ denotes the absolute difference of actual and predicted labels $|Y_{i,k} - \hat{Y}_{i,k}|$. Similarly, also the evaluation loss

$$\Psi^{loss} = \sum_{\{i,d\} \in \Omega_{\hat{j}}^{E,V}} \sum_{k=0,1,\dots K_j} Y_{i,k} \log \hat{Y}_{i,k} \tag{13}$$

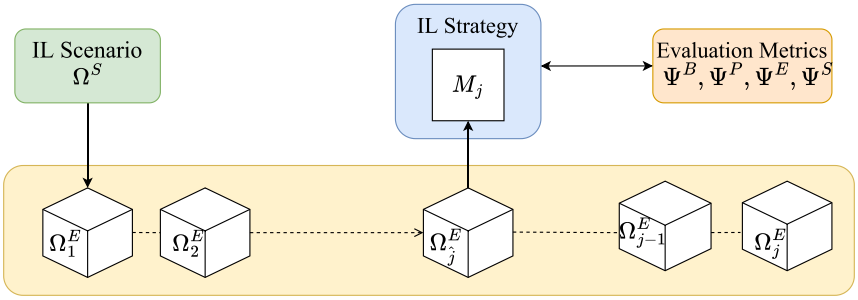

Fig. 2. Workflow description of incremental learning experiments.

can be computed which is similar to calculating ϵ but here the evaluation data $\Omega^{E,V}_{\hat{j}}$ is used. However, scenario level classification accuracy using $\Psi^{S,acc}$ or $\Psi^{S,loss}$ alone does not suffice as IL has additional (multi-) objectives such as efficient training in addition to the accuracy.

In response, this paper approaches the multi-objective nature of the problem by adding calculations of internal stage-wise metrics. These stage-wise metrics make the training efficient and calibrate the model hyper-parameters. Then, these performance metrics track and support the performance analysis of an IL strategy over a set of experiences in a scenario.

2.3. Workflow of incremental learning

The generic workflow of a class incremental learner, is shown in Fig. 2 that connects the three steps discussed in Section 2.2, generating experiences, the IL strategies and the evaluation metrics. The IL strategy selects sequentially the experiences $\hat{j}$ from the set of all experiences Ω^S. In each sequence, the IL strategy learns (incrementally) the model parameters Θ^{M_j} for the model M_j; then, updates the evaluation metrics $\Psi^{B,acc}$, $\Psi^{B,loss}$, $\Psi^{P,acc}$, $\Psi^{P,loss}$, $\Psi^{E,acc}$, $\Psi^{E,loss}$, $\Psi^{S,acc}$, $\Psi^{S,loss}$ that are based on Eqs. (12) and (13), respectively. In the remainder of this section, we simplify the notation to Ψ^B, Ψ^P, Ψ^E and Ψ^S by dropping the superscripts for *loss* and *acc*.

A naive class-IL strategy is shown in more detail in Algorithm 1. Initially, the experience training datasets $\Omega^{E,T}_{\hat{j}}$ are randomly split into B mini-batches $\Omega^{E,T,B}_{\hat{j},b}$ of equal size

$$|\Omega^{E,T,B}_{\hat{j},b}| = \frac{|\Omega^{E,T}_{\hat{j}}|}{B}, \qquad \forall b \in \Omega^B_{\hat{j}},\ \forall \hat{j} \in \Omega^S \tag{14}$$

where $\Omega^B_{\hat{j}}$ is the set of mini-batches for the experience $\hat{j}$. In the same way, the evaluation dataset $\Omega^{E,V}_{\hat{j}}$ is split into B mini-batches $\Omega^{E,V,B}_{\hat{j},b}$ of equal size. For each experience $\hat{j} \in \Omega^S$, the algorithm considers the training data $\Omega^{E,T}_{\hat{j}}$ in total $p_{\max}$ times when training the model M_j. There, in each epoch p the model is trained not on the entire training data at once, but in B mini-batches. When training with each mini-batch b, the model M_j predicts the labels $\hat{y}_i$ using Eq. (1), then the loss is computed with Eq. (11), respectively with mini-batch data $\forall\{i,d\} \in \Omega^{E,T,B}_{\hat{j},b}$. Subsequently, the model parameters Θ^{M_j} are updated by back-propagating the loss gradients. Subsequently, the classification accuracy Ψ^B_b as defined in Eq. (12) is computed using the evaluation data $\Omega^{E,V,B}_{\hat{j},b}$ of the batch b. After the model has been updated on all mini-batches, hence if, $b = p_{\max}$, then, the epoch p is completed. Subsequently, the average metric for the epoch p is

Algorithm 1 to run naive class-IL strategy

Input: Class incremental scenarios Ω^S with $|\Omega^S|$ experiences. Each experience $\hat{j} \in \Omega^S$ has $K_{\hat{j}}$ classes with training data $\Omega^{E,T}_{\hat{j}}$ and evaluation data $\Omega^{E,V}_{\hat{j}}$. The training data of each experience $\hat{j}$ is $\{i,d\} \in \Omega^{E,T,B}_{\hat{j},b}$ and the evaluation data is $\{i,d\} \in \Omega^{E,V,B}_{\hat{j},b}$, respectively for all batches $\forall\Omega^B_{\hat{j}}$. The training and evaluation data $\{i,d\}$ corresponds to the time-series $x_{i,d}$ and actual labels y_i. Learning rate L and stopping criterion $p_{\max}$.
Output: M_j with Θ^{M_j}, Ψ^B, Ψ^P, Ψ^J, Ψ^S

```
for ĵ = 1 to |Ω^S| do
  for p = 1 to p_max do
    for b = 1 to B do
      Predict ŷ_i with model M_j(x_{i,d}) ∀{i,d} ∈ Ω^{E,T,B}_{ĵ,b} as in Eq. (1)
      Compute loss Eq. (11) on (y_i, ŷ_i) ∀{i,d} ∈ Ω^{E,T,B}_{ĵ,b}
      Update Θ^{M_j} with L and back-propagation
      Update mini-batch metrics Ψ^B_b with Eq. (12) and Eq. (13) but
      using (y_i, ŷ_i) ∀{i,d} ∈ Ω^{E,V,B}_{ĵ,b}
    end for
    Update epoch metrics Ψ^P_p with Eq. (15)
  end for
  Update experience metrics Ψ^E_ĵ with Eq. (16)
end for
Update scenario metrics Ψ^S_s with Eq. (17)
```

$$\Psi^P_p = \frac{1}{B} \sum_{b \in \Omega^B_{\hat{j}}} \Psi^B_b. \tag{15}$$

The next epoch starts, so $p \leftarrow p+1$ is incremented, and all mini-batches are used again to further update the model parameters Θ^{M_j}. This sequence is repeated $p_{\max}$ times. The metric Ψ^P_p is the performance of the IL strategy when "showing" the data p times to the model. After the model M_j was trained $p_{\max}$ times on the data from experience $\hat{j}$, the average performance metric is computed across all epochs

$$\Psi^E_{\hat{j}} = \Psi^P_{p_{\max}}. \tag{16}$$

Then, the next experience is selected using Eq. (10).

In this naive baseline the experiences are selected one after each other $\hat{j} = 1 \ldots |\Omega^S|$. The model M_j hence is trained on that new selected experience as described above. Finally, after the model was trained on all experiences Ω^S, the overall average performance across all experiences is

$$\Psi^S_s = \frac{1}{|\Omega^S|} \sum_{\hat{j}=1,\ldots|\Omega^S|} \Psi^E_{\hat{j}}. \tag{17}$$

The IL strategies generally force additional control on updating model parameters (architectural and regularization-based strategies)

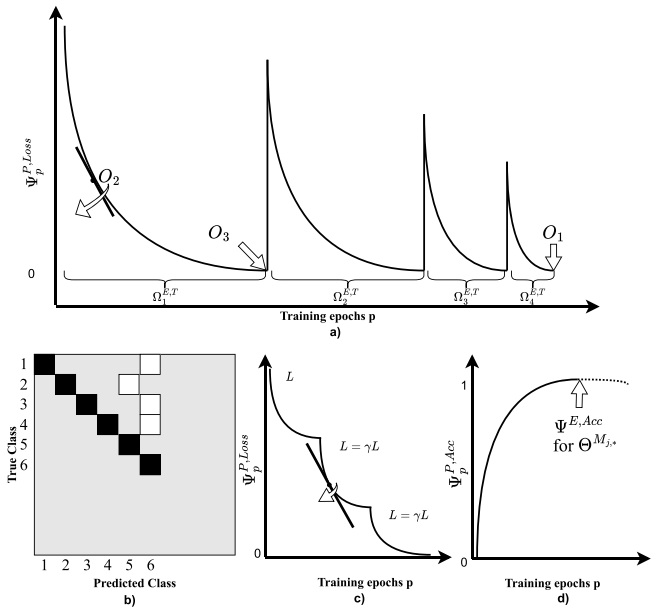

Fig. 3. Graphical representation of Dynamic IL objectives.

or modify the experience dataset(rehearsal-based strategies) to ensure high classification accuracy over all the incrementally considered experiences with the least catastrophic forgetting.

3. Dynamic incremental learning in real-time

This section introduces our proposed Dynamic IL method based on a rehearsal strategy to address the problem of catastrophic forgetting and, at the same time, achieve near-real-time training of new experiences. We target this by defining three objectives $O1$, $O2$, and $O3$ which are the following.

- $O1$ is avoiding catastrophic forgetting. This objective corresponds to maintaining classification accuracy for previous classes during training of new classes,
- $O2$ is a high learning efficiency. This objective corresponds to accuracy improvement per time and per number of training data, and
- $O3$ is a high classification accuracy at the end of each experience training phase.

Fig. 3.a summarizes this multi-objective problem using an ideal loss curve. If $O3$ determines the lowest $\Psi_p^{P,loss}$, ideally approaching 0 at an experience level, then $O1$ determines the lowest $\Psi_p^{P,loss}$, ideally approaching 0 at the scenario level considering all the experiences (or event types) seen so far. $O2$ can be assessed by the steepest slope approached in the loss curve, signifying fast training. To achieve these objectives, we propose Algorithm 2 by applying three modifications to Algorithm 1.

3.1. Modification I: Replay with dynamic buffer

The objective of the proposed *modification I* is $O1$ to maintain accuracy for previous classes, *modification I* controls the rehearsal to 'remember' the previously trained classes avoiding catastrophic forgetting and ensuring high classification accuracies. Also, this modification limits the training time ($O2$) for replay using reservoir sampling and introducing the limit κ on

$$\sum_{\hat{j}\in\Omega^S\setminus j} |\Omega_{\hat{j}}^{E,T}| \leq \kappa \tag{18}$$

Algorithm 2 to run dynamic IL for real-time training

Input: as in **Algorithm 1** but modified $\Omega_{\hat{j}}^{E,T}$ ⟵ Modification I

Output: as in **Algorithm 1**

1: **for** $\hat{j} = 1$ **to** $|\Omega^S|$ **do**
2: **for** $p = 1$ **to** $p_{\max}$ **do**
3: run lines 3:9 from **Algorithm 1**
4: **if** $|\Psi_p^{P,loss} - \Psi_{p-1}^{P,loss}| \leqslant \delta$ and $p > 1$ **then** } Modification II
5: $\xi = \xi + 1$
6: **else**
7: $\xi = 0$
8: **end if**
9: **if** $\xi \geq \xi_{\max}$ **then**
10: Update learning rate $L = \gamma L$
11: $\xi = 0$
12: **end if**
13: **if** $\Psi_p^{P,acc} > \Psi_{\hat{j}}^{E,acc}$ and $p > 1$ **then** } Modification III
14: Update best accuracy $\Psi_{\hat{j}}^{E,acc} = \Psi_p^{P,acc}$
15: Store best model $\Theta^{M_j,*} = \Theta^{M_j}$
16: $\iota = 0$
17: **else**
18: $\iota = \iota + 1$
19: **end if**
20: **if** $\iota \geq \iota_{\max}$ or $p \geq p_{\max}$ **then**
21: Take best model $\Theta^{M_j} = \Theta^{M_j,*}$
22: **break**
23: **end if**
24: **end for**
25: **end for**
26: Update scenario metric Ψ_s^S with Eq. (17)

the total amount of training data from previous experiences $\hat{j} \in \Omega^S\setminus j$. To comply with this limit, the training data per experience $\hat{j}$ is reduced by randomly removing the data

$$\{i, d\} \leftarrow \Omega_{\hat{j}}^{E,T}, \tag{19}$$

that is repeated until the experience $\hat{j}$ has the reduced training data size (training buffer size)

$$|\Omega_{\hat{j}}^{E,T}| = \frac{\kappa}{|\Omega^S| - 1} \qquad \forall \hat{j} \in \Omega^S\setminus j \tag{20}$$

and then, also constraint Eq. (18) holds. In this way, we prepare a training dataset, which is a representative collection of all datasets from previous experiences. The impact of this modification is illustrated in Fig. 3.b using a confusion matrix. A confusion matrix is a table that allows visualization of the classification accuracy considering multiple classes. Each row and column of the matrix represents the instances of the true class and the predicted class respectively. The 100% classification accuracy (true class = predicted class) is represented by a diagonal matrix (shown in black), however, when influenced by catastrophic forgetting, a conventional DL model inaccurately predicts/misclassifies previous class datasets into new classes forming a vertical strip (shown in white). Thereby reducing the classification accuracies of the previous classes. In IL, when a new class is introduced, a new class label/s is introduced and the evaluation of the proposed strategy is done after each experience training.

3.2. Modification II: Dynamic updating of learning rates

The objective of the proposed *modification II* is $O2$ to improve learning efficiency. As we propose to use rehearsal strategy in near real-time, for the first time, we consider the learning efficiency in rehearsal strategies as typically the training time is not an objective. This modification controls the learning rate L. As shown in Algorithm

2, in each epoch p the epoch metric $\Psi_p^{P,loss}$ is computed with Eqs. (13) and (15). Subsequently, the metric from epoch p is compared to the previous epoch $p-1$, and, if the reduction $|\Psi_p^{P,loss} - \Psi_{p-1}^{P,loss}| \leqslant \delta$ is lower than a threshold δ for $\xi_{\max}$ epochs, then, the learning rate is reduced by the factor $0 \leq \gamma \leq 1$. Therefore, the user has to specify the tuple $(\delta, \xi_{\max}, \gamma)$ based on the requirement and objectives of the application, e.g., to meet real-time requirements. The impact of this modification is illustrated in Fig. 3.c where, the L is reduced in steps of γ from the initial user-defined value L until $\xi_{\max}$ is reached.

3.3. Modification III: Dynamic termination of training

The objective of the proposed *modification III* is $O3$, maximizing the classification accuracy. This modification controls the number of training epochs p to avoid overfitting. The validation accuracy $\Psi_p^{P,acc}$ is computed with Eqs. (12), (15) after each epoch p. Then, only after the first epoch $p > 1$, this validation accuracy is compared to the best accuracy $\Psi_{\hat{j}}^{E,acc}$, and if greater $\Psi_p^{P,acc} > \Psi_{\hat{j}}^{E,acc}$, then, this updates the best validation accuracy $\Psi_{\hat{j}}^{E,acc} = \Psi_p^{P,acc}$, and the best model parameters $\Theta^{M_j,*} = \Theta^{M_j}$ are stored. However, the training terminates when the $\Psi_p^{P,acc}$ is not greater than the best accuracy for $\iota_{\max}$ times in a row. The impact of this modification is illustrated in Fig. 3.d where the validation accuracy after each epoch is recorded and compared with the previous value until no improvement is observed in a row.

4. Results and analysis

In this Section, we provide results and performance analysis of the Dynamic IL method that targets to incrementally train a model M_j in near real-time for unforeseen events without catastrophic forgetting. We achieve this by applying *modifications* to the naive class IL strategy to improve training time, experience data size, and classification accuracy. Classification accuracy is generally prioritized over training time and experience data size as it directly signifies the event recognition capability of the trained model. Due to multiple objectives, we have divided dynamic IL's performance analysis in terms of *formulations* (forms. in short). Table 1 provides a summary of various formulations highlighting the control parameters, evaluation metrics, assumptions, and goals. In form. A, we compare three promising IL strategies described in [27–29] with the baseline naive strategy and demonstrate the advantages of replay strategy in meeting $O1$ and $O3$ related to classification accuracy. However, due to near real-time training requirements, it is desirable to achieve this maximum classification accuracy in the minimum training time to describe as efficient learning $O2$. This minimum training time is achieved by simultaneously optimizing two different local hyper-parameters the learning rate L and the number of training epochs p_{max} described in form. B and C. The form. ABC describes the overall performance of the Dynamic IL method in achieving $O1$, $O2$, and $O3$. Furthermore, Section ABC^* and section ABC^{**} are special cases that extend our analysis further to check the robustness of our Dynamic IL method.

4.1. Dataset preparation

To demonstrate the applicability of our proposed Dynamic IL method, time-series signatures of nineteen disturbance event types ($K_j = 19$) are generated using a Real-Time Digital Simulator (RTDS). The method was applied on disturbance events that were generated on the MV ring network owned by Stedin [35]. The event signatures generated by RTDS corroborate the closest resemblance to the real-world signatures. However, the proposed method only depends on the provided measurements regardless of the network and the simulator type. Table 2 enlists the disturbance events along with their corresponding labels. The listed disturbance events are few of the many events that can be trained and classified. Our method can learn and further classify any disturbance event that produces a distinctive signature. However, some pseudo-events, like cyber-attacks that are non-distinctive and mimic other disturbance events, are hard to classify [36]. It requires system logs such as relay status log and network event monitor logs as input along with input measurements and also a model M that can process such heterogeneous inputs. One should note that, the class-IL workflow can accommodate multiple new classes witnessed at the same time. In this regard, three combination sets of scenarios are prepared in which each experience contains a different number of classes as shown in Table 2. Experiments performed in form. A, B, C, and ABC follow the scenario set A. The scenario set A has classes divided into five experiences $\Omega^E = [4, 4, 4, 4, 3]$ with randomly chosen classes. Every experience in scenario set A introduces four new classes for the model to learn. We use scenario set B to execute the form. ABC^* experiments where one class per experience is considered. This is the most practical scenario that is witnessed by the Dynamic IL method, as multiple new event types at the same time are rare. The first experience has two classes and hence there are eighteen sets of experience. The scenario set C is used for form. ABC^{**} experiments which have two classes per experience. Classes that belong to each experience in a scenario will not undergo shuffling to maintain the experiment's repeatability. However, the datasets inside each experience are shuffled for better generalization during the training procedure. The number of event samples belonging to each class has not been massively varied, since a nearly balanced dataset is assumed. The event's sampling rates are carefully selected to prioritize the clarity of the event's dynamics however, for each event i, the record of the event $x_{i,d}$ has the dimensions $\upsilon \times \tau = 6 \times 500$ as shown in Eq. (2). The measurement variables are the time-series signatures of voltages and currents. These dimensions are maintained consistently throughout our experiments since the CNN model is designed to accommodate these input dimensions. However, users can design models based on their event measurement dimensions and still use our Dynamic IL method.

4.2. Form. A — Comparative analysis between various rehearsal IL strategies

This study first demonstrates the phenomenon of catastrophic forgetting, and thereafter, how to address the phenomenon with class IL learning. Fig. 4 shows the confusion matrices after training two consecutive experiences using naive strategy (Algorithm 1). It can be observed that when the second experience set Ω_j^E is trained on the model M_j, the model completely forgets the previously trained classes by Ω_{j-1}^E and miss-classify all the previously trained events to the classes corresponding to new sets of experience. This appears as a vertical strip in an evolving confusion matrix. It should be pointed out that the confusion matrix also shows the classes from future sets of experience. However, this is not considered during the computation of classification accuracy.

Fig. 5 shows the results of the comparative analysis for various class IL strategies that address catastrophic forgetting using evolving confusion matrices. The comparison along *x-axis* denotes the IL strategy's performance on experiences, while the comparison along *y-axis* denotes a comparative analysis of rehearsal IL strategies for remembering previous experience's classes.

The naive strategy excessively forgets the previously trained classes which can be observed in row 1. The model is unsuccessful in remembering the old classes, and only the classes corresponding to the current dataset Ω_j^E are classified correctly.

The Learning Without Forgetting (LWF) strategy [29] retains by transferring the learned "knowledge" from a previously trained model to the model trained on the new data. As shown in row 2, as the model is highly dependent on data, the model gradually builds up errors. From columns 1 and 2, we can observe that the model aims at "remembering" classes from the first experience on the diagonal (since the events are concentrated on the diagonal). However, as the classification error

Table 1
Formulations derived to monitor and measure the key performance indicators of Dynamic IL method.

Form.	Designed for	Strategy	Metric	Optimizing Parameter	Assumptions	Parameters	Goal
A (Section 3.1)	Choosing best IL Strategy	Rehearsal Strategies	Evolving Confusion Matrix	Classification Accuracy	Fixed L and Fixed p_{max}	Memory buffer size κ	Training without Catastrophic Forgetting
B (Section 3.2)	Optimizing L for individual experiences	Naive	$\Psi_p^{P,loss}$	Learning Rate (L)	Fixed p_{max}	ξ_{max}, γ, δ	Controlling L s.t. $\Psi_p^{P,loss}$ reaches 0 at the earliest for Ω_j^E
C (Section 3.3)	Dynamic termination of training	Naive	$\Psi_p^{P,acc}$	$\Theta^{M_j,*}$ and p	Fixed L	ι_{max}	Choosing optimal $\Theta^{M_j,*}$ specific to Ω_j^E avoiding over-training
ABC	Evaluating overall performance on scenario level acc. and training time	Replay with dynamic memory buffer	$\Psi^{S,acc}$ and overall training time	Events classification accuracy and training time	–	Parameters of A, B and C	Efficient near real-time learning without catastrophic forgetting
ABC^*	Investigation on one class per experience case which is highly probable in power grids	Replay with dynamic memory buffer	Eval. Acc and Confusion Matrix	Adaptation of weights and biases due to constant new event types	One or max. two event classes per experience	Parameters of A, B and C	To analyze performance metrics when experience contain fewer event samples
ABC^{**}	Evaluating influence of mem. buffer size on classification acc.	Replay with dynamic memory buffer	Memory buffer size and Eval. accuracy	Memory buffer size (κ)	–	Memory buffer size κ	To achieve max. classification acc. with minimal mem. buffer size

Table 2
Description of disturbance event classes and their corresponding scenario sets. Forms. A, B, C, ABC follow scenario set A, form. ABC^* follow scenario set B and form. ABC^{**} follow scenario set C.

Disturbance event class	Event ID	Scenario set A	Scenario set B	Scenario set C
3 phase fault	0	Ω_5^E	Ω_1^E	Ω_1^E
F + 2nd harmonics	1	Ω_1^E	Ω_1^E	Ω_1^E
F + 3rd harmonics	2	Ω_5^E	Ω_2^E	Ω_1^E
F + 4th harmonics	3	Ω_4^E	Ω_3^E	Ω_2^E
F + 5th harmonics	4	Ω_1^E	Ω_4^E	Ω_2^E
F + 7th harmonics	5	Ω_3^E	Ω_5^E	Ω_3^E
F + 9th harmonics	6	Ω_4^E	Ω_6^E	Ω_3^E
F + 11th harmonics	7	Ω_1^E	Ω_7^E	Ω_4^E
F + 3rd + 5th harmonics	8	Ω_1^E	Ω_8^E	Ω_4^E
F + 3rd + 7th harmonics	9	Ω_4^E	Ω_9^E	Ω_5^E
F + 3rd + 11th harmonics	10	Ω_4^E	Ω_{10}^E	Ω_5^E
AG fault	11	Ω_5^E	Ω_{11}^E	Ω_6^E
BG fault	12	Ω_3^E	Ω_{12}^E	Ω_6^E
CG fault	13	Ω_2^E	Ω_{13}^E	Ω_7^E
ABG fault	14	Ω_3^E	Ω_{14}^E	Ω_7^E
BCG fault	15	Ω_2^E	Ω_{15}^E	Ω_8^E
CAG fault	16	Ω_2^E	Ω_{16}^E	Ω_8^E
Healthy data	17	Ω_2^E	Ω_{17}^E	Ω_9^E
Lightening data	18	Ω_3^E	Ω_{18}^E	Ω_9^E

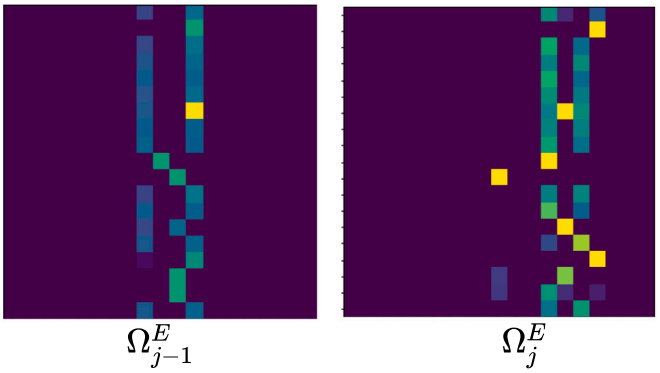

Fig. 4. Effects of catastrophic forgetting observed between two consecutive experiences in the case of naive strategy. Forgetting is clearly highlighted by a vertical band moving from Ω_{j-1}^E to Ω_j^E, without remembering the knowledge acquired from previous experience training.

builds up over time, the learning capabilities drastically decrease after the second experience (third column and following).

The Elastic Weight Consolidation (EWC) strategy [30] is shown in row 3. EWC strategy is best known for Domain Incremental Learning, where the "ground truth" of the disturbance events drift over time. However, it is investigated in class incremental learning due to its translational property. The strategy slows down the learning of certain weights based on how important they are to previously seen classes. From the figure, it can be seen that the strategy performs poorly for the initial few experiences, however, the classification accuracy improved at the later stages (row 3, column 5), signifying the inter-dependency of model size and the number of classes in the input dataset.

The Greedy sampler, Dumb selector (GDumb) strategy [28] is shown in row 4. This strategy replays old experiences and has the potential to addressing catastrophic forgetting. However, still, we can see some miss-classifications at some individual class classification-accuracy. Therefore, the training is needed for a higher number of epochs than the prefixed value ($p_{max} = 100$) that is not aligned with objective $O2$.

The Replay strategy [27] with additional reservoir sampling and parametric buffer see row 4, shows the best performance as the trained events are stacked on the diagonal of a confusion matrix, and at each experience stage, the model remembers classes from the previous experiences without catastrophic forgetting. Hence, the replay strategy with parametric buffer and reservoir sampling will be further used for the proposed dynamic IL implementation in form. ABC, ABC^*, ABC^{**}. This strategy shows a high performance as the strategy organizes the way of sampling and storing the experience datasets. The parametric

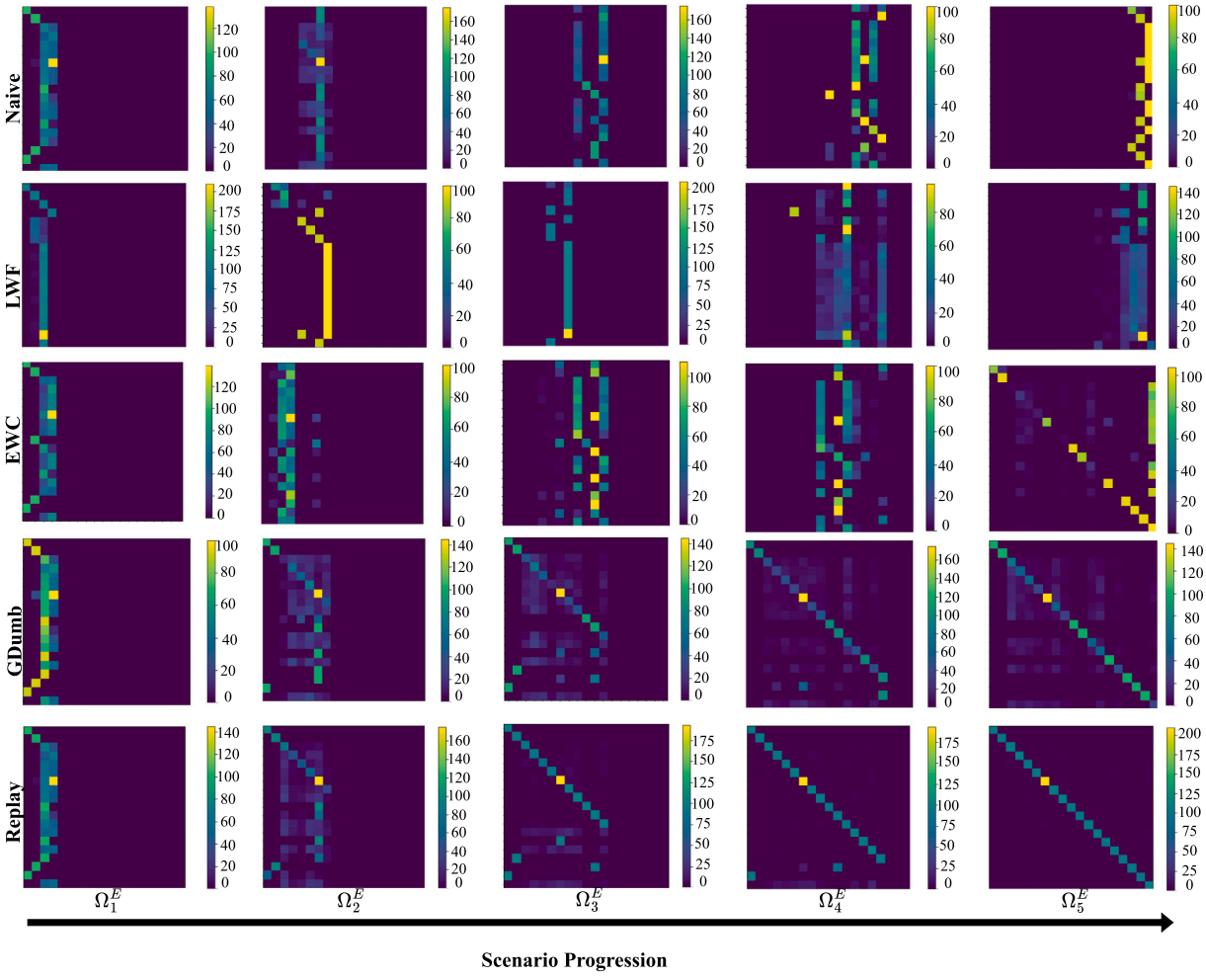

Fig. 5. Comparison between various class IL strategies suffering from catastrophic forgetting phenomenon. Each sub-figure corresponds to one confusion matrix with individual value scales ranging from 0 to 200. Each row represents one IL strategy. Each column represents a "snapshot" in time of the strategies' performances, evolving column-wise from left to right. Hence, the strategies can be compared across the columns. The performance is high when only values are on the diagonal, and all others are zero. Then, the disturbances are detected well, with a few classification errors.

buffer updates every time the algorithm receives a new experience. The buffer has a limit of $\kappa = 0.7$, signifying that 70% of the datasets from the previous experiences are stored and replayed. The parametric buffer memory fills with data samples from previous experiences until the maximum buffer capacity of $|\Omega_{\hat{j}}^{E,T}| = 300$ samples. However, on further experiences' arrival, the buffer data samples are randomly replaced to accommodate new experience datasets. Reservoir sampling ensures that the data samples are replaced from the parametric buffer such that the parametric buffer contains an equal number of event datasets to avoid imbalance in the dataset from all classes seen so far. The limit κ and maximum buffer capacity $|\Omega_{\hat{j}}^{E,T}|$ can be chosen based on user requirements after ensuring computational capability.

4.3. Form. B — Dynamic updating of learning rates

This study investigates the improvement in minimizing the loss with *modification II* that proposes a dynamic learning rate L. We start by investigating the importance of L in training. Four different experiments are performed on Algorithm 1 with $L = [0.1, 0.001, 0.0006, 0.0001]$. Conventional DL uses a static dataset where the full dataset is available all at once. The value of L is determined based on a parametric sweep to achieve the best results on the classification accuracy. For class IL, when the model is presented with a new unknown set of event classes with varying and unknown transient dynamics, a single pre-determined L may lead to overshooting or slow learning effects that contemplate real-time model adaptation capabilities. For instance, $L = 0.1$ performs poorly, and the loss does not converge to 0 for all Ω_j^E. $L = 0.001$ performs poorly on Ω_2^E while other experiences adapt well. For $L = 0.0001$, Ω_1^E and Ω_2^E does not converge well. Caused by this specific L requirement for each experience dataset, the user cannot pre-define the L value, which highlights the importance of dynamic updating of learning rates by tracking loss $\Psi_p^{P,loss}$ "on-the-fly". As discussed in Section 3.2, we apply *modification II* to the Algorithm 1 whilst setting aside other modifications. The results from these four experiments are documented in Fig. 6 demonstrate the importance of L on training loss $\Psi_p^{P,loss}$. Fig. 7 presents a comparison between *modification II* and the best suitable $L = 0.0006$ for all experience training's in this scenario. Each experience dataset Ω_j^E starts learning with a conservative $L = 0.001$. Once the loss stops reducing for $\xi_{max} = 5$ epochs, L is reduced by a factor of $\gamma = 0.9$ as shown in Algorithm 2, line 10. We observe that *modification II* outperforms (or is at least equal to) all training settings with a static, fixed L. For each experience training the value of L updates approaching objectives $O2$ and $O3$. However, during the training of Ω_2^E, we observe that the steepness of the loss curve for *modification II* is lower than that when using a fixed, static $L = 0.0006$. Therefore, using this static L is sub-optimal and not aligned with the objective $O2$. This accounts for an unnecessary update of L at an early stage causing a delay. However, the loss value is lower when using *modification II*. Table 3 shows the final value of L attained at the end of each experience's training. This is the closest value to optimally

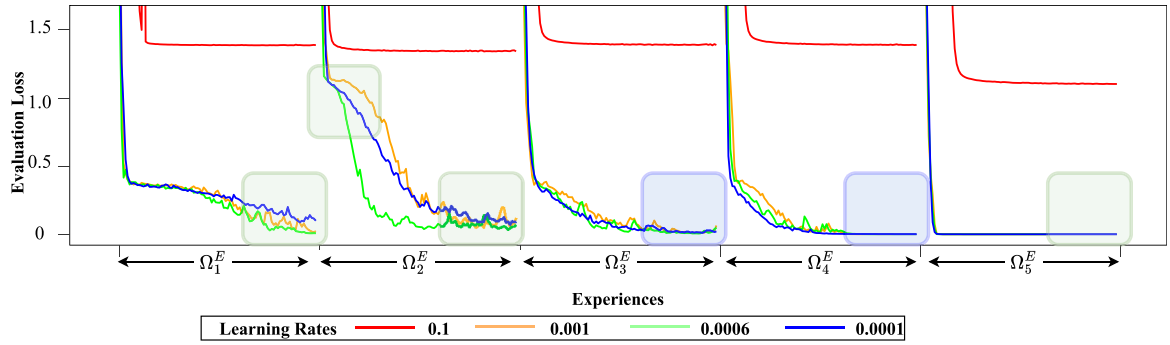

Fig. 6. Evaluation loss $\Psi_p^{P,loss}$ is computed for experiences Ω_j^E while adopting different L. It is preferred that $\Psi_p^{P,loss}$ curves should reduce to 0 at the earliest. $\Omega_{1,2,5}^E$ performs best with $L = 0.0006$, whereas $\Omega_{3,4}^E$ performs best with $L = 0.0001$. This signifies the need for dynamic updates of L unique to each experience.

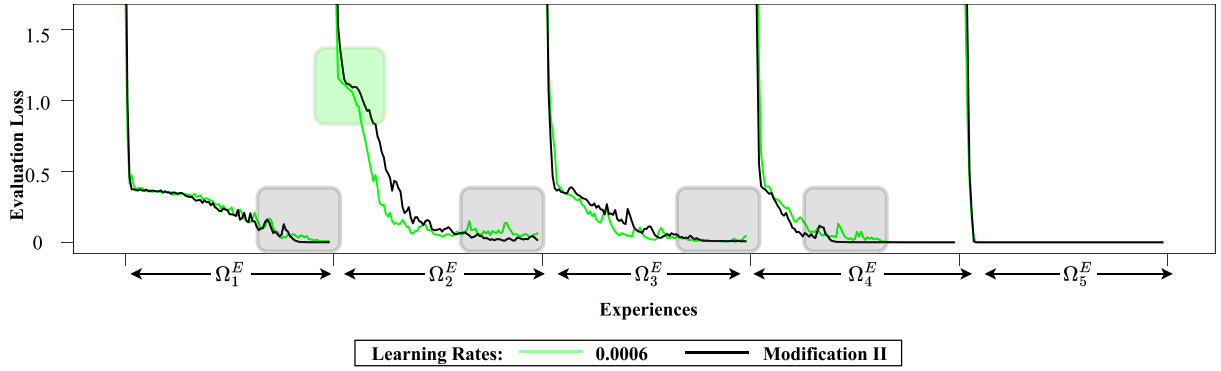

Fig. 7. Comparison between evaluation loss $\Psi_p^{P,loss}$ computed for experiences Ω_j^E using fixed learning rate $L = 0.0006$ and proposed *modification II* case. It is preferred that $\Psi_p^{P,loss}$ curves should reduce to 0 at the earliest. The *modification II* case as highlighted recorded a lower $\Psi_p^{P,loss}$ than best performing case of fixed $L = 0.0006$.

Table 3
Learning rate L settled at the end of each experience training due to *modification II*.

Settled hyperparameter	Ω_1^E	Ω_2^E	Ω_3^E	Ω_4^E	Ω_5^E
L	0.0007	0.00059	0.00065	0.0009	0.001

approach $O3$ with the aforementioned static, baseline naive approach using the static, fixed L. The higher L for a particular experience, the lower are the temporal complexities of event signatures, and the selected L is optimal, requiring no step-wise reduction of L. However, a lower L for a particular experience causes a step-wise reduction in L from the conservative value of $L = 0.001$. The final settled L value will be the most suitable learning rate for event signatures that are complex over time. Thus, *modification II* makes the Dynamic IL method adapting when the complexities of event signatures change over time. This feature of *modification II* adapting to these complexities will be useful during the transitional phase of the electric grids. For example, it will be useful when the grid simultaneously uses the two old and new devices with low and high sampling rates.

4.4. Form. C: Dynamic termination of training

The goal of this study is to investigate the importance of experience training duration and the impact of *modification III* on approaching $O2$ and $O3$. The experiments show the impact of *modification III* on Algorithm 1 whilst setting aside the other modifications. Typically, in conventional DL, the total number of training epochs p_{max} is selected based on user expertise. Then, the model is trained on a static dataset until the user observes the maximum improvement in evaluation accuracy.

This case study trains a model with class IL where the model is incrementally presented with a new unknown set of event classes. The new event classes have different transient dynamics that are unknown to the model (the model has never seen these before). As the pre-determined p_{max} value may result in under-fitting or over-fitting (e.g., as in conventional DL), we constrain the training for each Ω_j^E "on-the-fly" according to the maximum $\Psi_p^{P,acc}$. *Modification III* tracks $\Psi_p^{P,acc}$ during the training process, and the training terminates when $\Psi_p^{P,acc}$ does not improve any further for $\iota_{max} = 10$ epochs in a row. The model parameters $\Theta^{M_j,*}$ with the highest classification accuracy are stored as shown in Algorithm 2, line 21. Later, these model parameters $\Theta^{M_j,*}$ will be used to further learn new incoming experiences. The experiments are conducted with fixed $L = 0.0006$. The results of these experiments are shown in Fig. 8 and Table 4. Fig. 8 shows the effect of the dynamic termination of experience training. The black line corresponding to *modification III* shows that training intelligently terminates compared to the static case of $p_{max} = 100$. This reduction in p_{max} implies improved training time $O2$. For most cases of experience training, there is no major change in $\Psi^{E,acc}$ $O3$ because of *modification III*. However, when looking at Ω_3^E in Fig. 8, we observe *modification III* addressing over-fitting concerns. The experience evaluation accuracy $\Psi^{E,acc}$ starts degrading when trained with a pre-determined number of epochs $p_{max} = 100$. Table 4 shows the number of training epochs for each experience p_{max} and highest experience classification accuracy recorded for each experience Ω_j^E for *modification III*. We see that there is a high degree of variance in choosing an optimal number of p_{max} for each Ω_j^E signifying the importance of *modification III* when training new event classes incrementally.

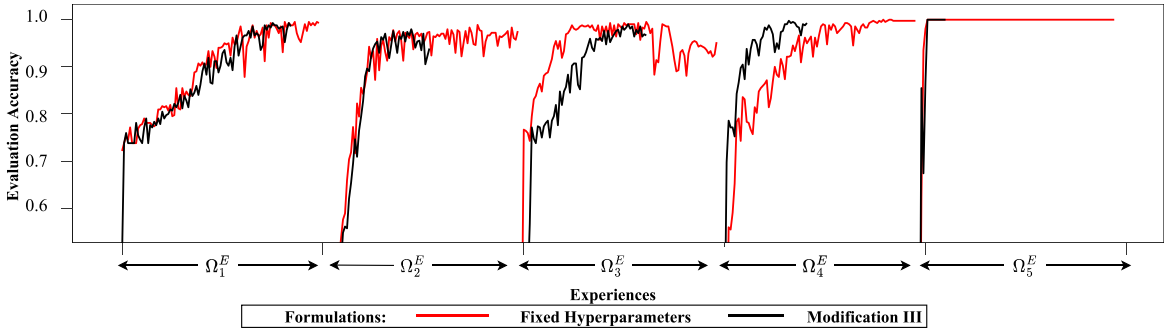

Fig. 8. The fast termination of experience's training is highlighted by comparing Form. B with the base case (Algorithm 1). It is preferred that evaluation accuracy $\Psi_p^{P,acc}$ should reach close to '1' within the least number of epochs. Experience training is terminated by *modification III* sub-algorithm once $\Psi_p^{P,acc}$ reaches a maximum and no further improvement is observed.

Table 4
Performance of *modification III* measured as p_{max} and $\Psi^{E,acc}$ during training after each experience.

Evaluation metrics	Ω_1^E	Ω_2^E	Ω_3^E	Ω_4^E	Ω_5^E
Number of training epochs p_{max}	84	55	64	45	14
Experience classification accuracy $\Psi^{E,acc}$	99.3%	98.0%	99%	99.8%	100%

Table 5
Performance analysis of the Dynamic IL method. Improvement in training time and maximum experience and scenario classification accuracy obtained for each event experience set when considering each objective individually.

Objectives	Experiences				
	Ω_1^E	Ω_2^E	Ω_3^E	Ω_4^E	Ω_5^E
$O1$:Scenario classification accuracy $\Psi^{S,acc}$	98.9%	96.9%	96.0%	95.1%	94.3%
$O2$: Learning efficiency (% improv. in training time)	9.1%	39.0%	65.3%	74.3%	88.8%
$O3$: Experience classification accuracy $\Psi^{E,acc}$	98.9%	98.6%	99.1%	98.6%	100%

4.5. Form. ABC — Dynamic incremental learning method

The objective of this study is to investigate the performance of the Dynamic IL method, described in Algorithm 2.

Table 5 shows the obtained results for objectives $O1$, $O2$, and $O3$ under the combined influence of *modifications I, II* and *III*. This is different from the previous case studies where these modifications were activated individually. All other experiment settings remain the same as in previous studies. $O1$ is analyzed by the scenario classification accuracy $\Psi^{S,acc}$ which is computed by taking into account all the trained classes seen so far. We observe from Fig. 9.a that the catastrophic forgetting is greatly reduced compared to the naive strategy. The overall achieved scenario accuracy after training the model incrementally in near real-time is 94.3%. From Table 5, row 3 it can be seen a slight decrease in $\Psi^{S,acc}$ after each experience training, signifying the presence of catastrophic forgetting concern. It is important to note that the memory buffer limit is $\kappa = 0.7$. The effect of varying κ on $O1$ shall be discussed in form. ABC^{**}.

In the absence of reservoir sampling, the dataset grows proportionally with each new experience, and the execution time of each training epoch increases, thereby increasing the experience training time. However, in form. ABC, we see that the training time is maintained below 25 s due to efficient learning modifications, which were discussed in Sections 3.1–3.3. The results of $O3$, the experience classification accuracy, are shown in Table 5, row 5. The values provided in this row are computed without taking into account the previous class's accuracy. *Modification I* and *modification II* collectively contribute to these results as L and p_{max} are tuned "on-the-fly". The results show that the experience classification accuracy $\Psi^{E,acc}$ remains similar or sometimes is improved due to the collective contribution of *modification I* and *modification II*. Additionally, by comparing Tables 4 and 5, we observe that for $\Omega_{1,4}^E$, experience classification accuracy $O3$ value is reduced whereas learning efficiency $O2$ is improved for the form. ABC. However, it is important to have $O3$ prioritized over $O2$ since classification accuracy holds importance over learning efficiency for event classification tasks. This behavior points out the need for advanced metric prioritizing algorithms where the user can prioritize performance metrics based on his requirement.

The discussed results are specific to the event datasets of Section 4.1, DL model Appendix and other strategy-specific parameters as described in Section 3. However, similar results can be expected when the Dynamic IL method is applied together with the rest of the real-time event classification workflow, as indicated in Fig. 1.

4.6. Form. ABC^* — Special case considering one class per experience

This section aims to demonstrate the performance of the Dynamic IL method when each experience contains only one event class as described in scenario set B.

It is common in IL (or DL in general) to combine multiple classes in the form of batches before feeding them into the model. However, in an actual power system, the occurrence of events is rare, non-deterministic, and unique. The recorded signatures at any time belong to an individual class (rarely two classes, due to cascading effect). Hence, when testing the Dynamic IL method with scenario set B, we corroborate the learning of events close to real-life scenarios. The findings presented in Table 6 show the performance of the Dynamic IL method for the case of scenario set B (one class per experience). The achieved scenario accuracy $\Psi^{S,acc}$ at the end of the experiment is 94.9%, and it is similar to that of $\Psi^{S,acc}$ which is 94.3% in form. ABC. This demonstrates clearly dynamic IL's implementation in an actual power system. The time taken for real-time event classification on already trained disturbance events ranges from 0.36 ms to 0.49 ms. This is for the disturbance events described in Table 2 and for the 1D CNN model in Appendix. The time taken for new disturbance event learning is shown in Table 6. The varying training time (expressed in training epochs) is due to the varying complexity of the input datasets and pre-trained model state. One should note that we consider a balanced dataset during the entire learning process. However, in reality, the number of disturbance datasets collected will be proportional to the impact of a disturbance event on the grid. Here, our findings on training the model incrementally with one class per experience are primary analyses of the possibility of training incrementally with a few datasets. This model training corresponds to the rarest and most severe events. IL has a specialized branch called one-shot learning [37] that investigates this operational setting in more detail.

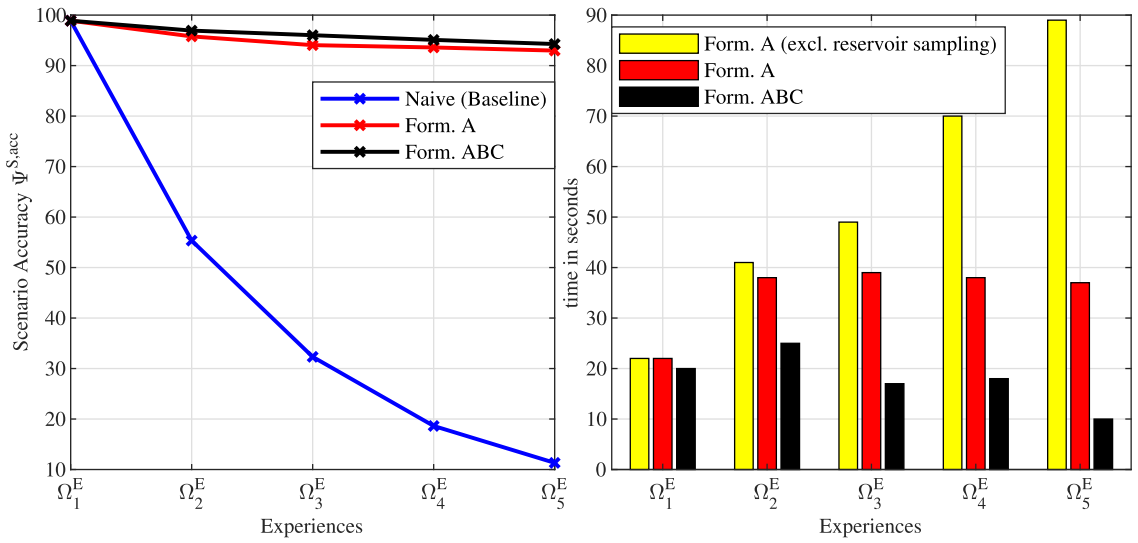

Fig. 9. (a) Scenario classification accuracy is computed after each experience's training to evaluate the effect of catastrophic forgetting $O1$ caused by IL. (b) Training time comparison between form. *ABC* and form. A shows that *modifications I, II, and III* collectively result in reduced training time below 25 s for form. *ABC* and improving learning efficiency $O2$.

Table 6
Performance metrics of Dynamic IL method in case of scenario set B (one class per experience).

Experience	Event class	Optimized L	Training epochs	Scenario accuracy
Ω_1^E	0,1	0.0008	13	100%
Ω_2^E	2	0.001	12	100%
Ω_3^E	3	0.001	12	100%
Ω_4^E	4	0.001	12	100%
Ω_5^E	5	0.001	12	100%
Ω_6^E	6	0.001	13	100%
Ω_7^E	7	0.001	14	99.6%
Ω_8^E	8	0.001	19	99.4%
Ω_9^E	9	0.0008	17	99.01%
Ω_{10}^E	10	0.001	15	98.92%
Ω_{11}^E	11	0.001	13	98.80%
Ω_{12}^E	12	0.001	17	98.06%
Ω_{13}^E	13	0.0008	18	97.8%
Ω_{14}^E	14	0.001	18	96.99%
Ω_{15}^E	15	0.00064	66	96.37%
Ω_{16}^E	16	0.0008	29	95.84%
Ω_{17}^E	17	0.00051	32	95.61%
Ω_{18}^E	18	0.0008	13	94.93%

4.7. Form. ABC^{**} — Enhancement of dynamic IL performance by increasing buffer size

This section investigates the impact of the memory buffer limit κ on scenario classification accuracy $\Psi^{S,acc}$.

Rehearsal-based IL strategies present various approaches to optimize the buffer memory design to achieve the highest classification accuracy with the most diminutive memory buffer size. Here, we adapt parametric buffer and reservoir sampling (dynamic memory buffer) to efficiently manage buffer memory requirements. Fig. 10 presents the scenario classification accuracy achieved by applying scenario set C to the Dynamic IL method with a variable κ. A direct co-relation between $\Psi^{S,acc}$ and κ indicates the need of designing more strategic sample selection policy to optimize the memory buffer size that can achieve higher scenario classification accuracy. With this strategic sample selection policy the memory buffer can be designed based on the user's event learning efficiency $O2$ requirement. In other words, $O2$ and $O1$ can be tuned in a way to meet the best solutions with advanced high-processing control room equipment of the future.

4.8. Limitations of dynamic IL method

In this section, the limitations of the developed model are discussed. Although the Dynamic IL method addresses the problem of catastrophic forgetting and, at the same time, achieves near-real-time training of newly detected disturbance events, it still suffers from gradual miss-classification. As this gradual reduction of classification accuracy due to re-training on-the-fly is a limitation, it might not be acceptable in the future. Hence, the threshold for model re-training can be defined as a performance metric, and further actions can be suggested. Our model and all deep learning-based models for event classification have a fixed input size, which requires the disturbance event signatures to be of fixed length. However, the event signatures range from transient to steady-state events captured at different sampling rates. Hence, this requires a pre-processing stage of online over-sampling or under-sampling. This stage might alter the 'truth' of a disturbance event. For example, high-frequency switching events and cyber-attack events might require a different architecture or more preprocessing steps with human intervention to add to the training data. Another aspect is that Our present work requires labeled datasets that are not always available in actual power grids. Our Dynamic IL method follows a supervised learning procedure, where the event type (y label data) is used during the training. Then, the model's classification accuracy is measured against this labeled data. In actual power grids, the event datasets acquired from the grid for training will be without labels. Our future work will include unsupervised learning, which may be able to assign these labels.

5. Conclusion

This paper deals with the development and the demonstration of a Dynamic IL method that can train a DL model incrementally in near real-time. The method is analyzed and validated by devising various formulations which investigate its implementation in an automated real-time event classification workflow that detect, classify, and adapt to unknown disturbance events. The performed analysis shows the model's learning capability of new unforeseen disturbance events without catastrophic forgetting of the old events. Prompt recognition and

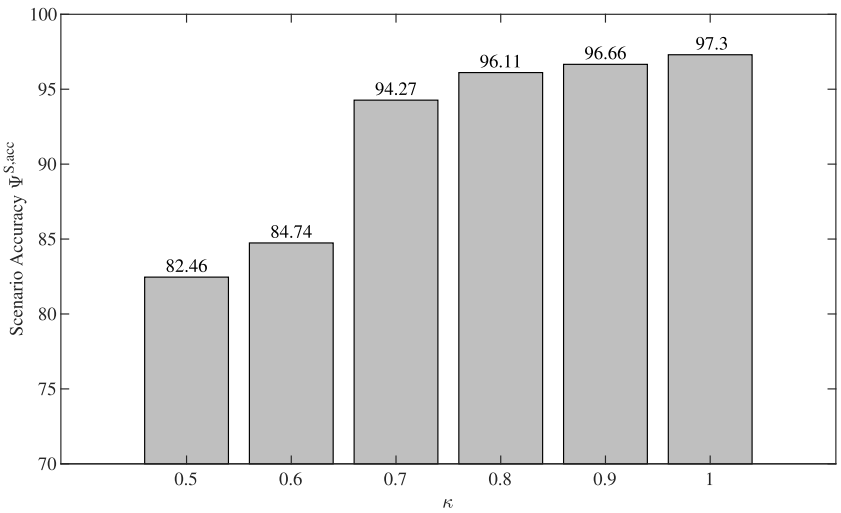

Fig. 10. Scenario classification accuracy $\Psi^{S,acc}$ compared to the dynamic buffer size κ. The direct co-relation signifies the importance of κ in achieving higher $\Psi^{S,acc}$.

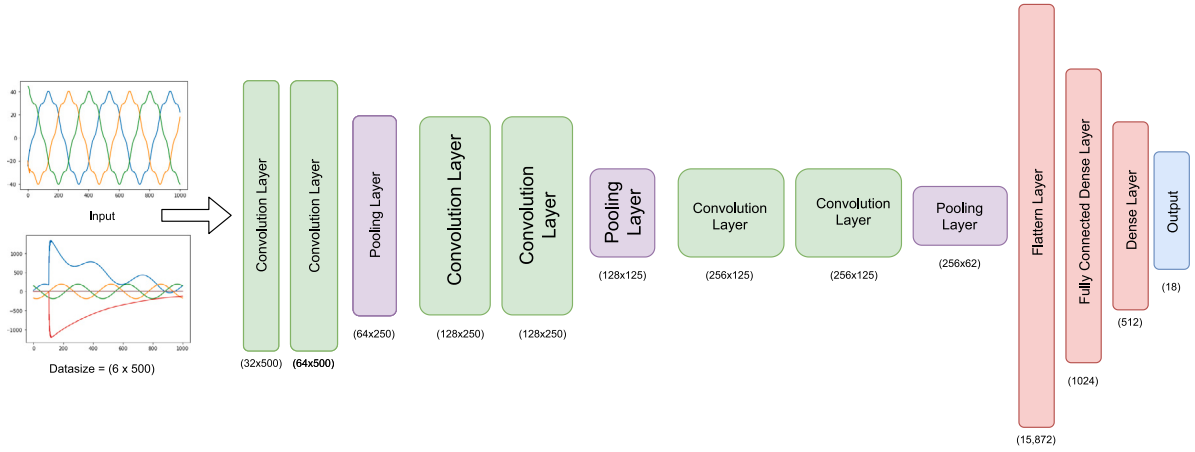

Fig. A.11. 1D Convolutional Neural Network model.

learning new events are of utmost importance for the control room operators to prevent cascading effects that may lead to blackouts.

To achieve a high classification accuracy and computationally efficient learning in terms of training time and new events/experience data size, three modifications are applied to the naive class IL strategy (explained in Algorithm 1). *Modification I* utilizes a replay strategy with a dynamic buffer to tackle catastrophic forgetting. In this way, it improves the classification accuracy and optimizes the experience data size. *Modification II* automates the update of learning rates L, which significantly reduces the training time and improves the classification accuracy as the sub-algorithm also minimizes loss. *Modification III* automates the training process termination when the best model parameters resulting in the highest classification accuracy are obtained. All these modifications enable the Dynamic IL method (in Algorithm 2) to fulfill a near-real-time learning and an associated event classification objectives.

To study the robustness of replay strategy with a dynamic buffer against catastrophic forgetting, replay strategy is compared by other rehearsal-based IL strategies — LWF and GDumb. The evolving confusion matrix depicts the superiority of our replay strategy with a dynamic buffer against other strategies by achieving the highest scenario (overall) classification accuracy. Furthermore, two special sections are prepared to validate our method for real-life scenarios. Form. ABC^{*} demonstrates the dynamic IL's capability to operate for one event class/experience scenario, which is common in practice. Form. ABC^{**} demonstrates the direct proportionality between the memory buffer size and the classification accuracy indicating the importance of buffer memory requirements. However, an increase in memory buffer size, increases the training time, and thereby decreases learning efficiency. Hence, our proposed method allows the user to choose between learning efficiency and classification accuracy during model training.

Lastly, we follow a modular approach for the development of the Dynamic IL method. Each sub-module can investigate more complex optimization techniques, thus providing an opportunity to expand our research continuously. To this end, advanced architectural and regularization-based IL strategies will be explored to improve the performance metrics of the Dynamic IL method. As a future work, we intend to expand the Dynamic IL method with a user-defined priority metric that can prioritize between training time, memory buffer size, and classification accuracy. This user-defined priority metric can be based on the severity and importance of the event.

CRediT authorship contribution statement

Nidarshan Veerakumar: Conceptualization, Methodology, Software, Investigation, Writing – original draft. **Jochen L. Cremer:** Conceptualization, Formal analysis, Writing – review & editing. **Marjan Popov:** Resources, Writing – review & editing, Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This research work is financially supported by the Nederlandse Organisatie voor Wetenschappelijk Onderzoek NWO in collaboration with TSO TenneT, DSOs Alliander, Stedin, Enduris, VSL and General Electric in the framework of the Energy System Integration & Big Data program under the project "Resilient Synchromeasurement-based Grid Protection Platform, No. 647.003.004" and the TU Delft AI Labs Programme, NL, The Netherlands.

Appendix. Model architecture

The model chosen for this particular multivariate time series analysis comprises 1D Convolutional Neural Network (CNN) models as depicted in Fig. A.11. The 1D model is chosen over the 2D CNN model since the time-series signals of voltages and currents are to be analyzed individually without interference from each other. The number of layers and the size of each layer is selected by trial and error analyzing the overall accuracy on the dataset.

References

[1] Xu W, et al. Electric signatures of power equipment failures. In: Proc. 2016 IEEE power energy soc. gen. meeting. 2016.
[2] Ibarra L, Rosales A, Ponce P, Molina A, Ayyanar R. Overview of real-time simulation as a supporting effort to smart-grid attainment. Energies 2017;10(6):1–24.
[3] Costa FB, Souza BA, Brito NS. Real-time detection and classification of power system disturbances based on maximal overlap discrete wavelet transform. In: 17th IEEE power systems computation conference. (1). 2011.
[4] Igual R, Medrano C. Research challenges in real-time classification of power quality disturbances applicable to microgrids: A systematic review. Renew Sustain Energy Rev 2020.
[5] Sevilla FRS, Liu Y, Barocio E, Korba P, Andrade M, Bellizio F, et al. State-of-the-art of data collection, analytics, and future needs of transmission utilities worldwide to account for the continuous growth of sensing data. Int J Electr Power Energy Syst 2022;137:107772.
[6] Amutha AL, Annie Uthra R, Preetha Roselyn J, Golda Brunet R. Anomaly detection in multivariate streaming PMU data using density estimation technique in wide area monitoring system. Expert Syst Appl 2021;175:114865.
[7] Hao Y, Chen Y, Zakaria J, Hu B, Rakthanmanon T, Keogh E. Towards never-ending learning from time series streams. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining. vol. Part F1288, 2013, p. 874–82.
[8] Ashraf SM, Gupta A, Choudhary DK, Chakrabarti S. Voltage stability monitoring of power systems using reduced network and artificial neural network. Int J Electr Power Energy Syst 2017;87:43–51.
[9] Bellizio F, Cremer J, Strbac G. Machine-learned security assessment for changing system topologies. Int J Electr Power Energy Syst 2022;134.
[10] Monedero I, León C, Ropero J, García A, Elena JM, Montaño JC. Classification of electrical disturbances in real time using neural networks. IEEE Trans Power Deliv 2007;22(3):1288–96.
[11] Hannon C, Deka D, Jin D, Vuffray M, Lokhov AY. Real-time anomaly detection and classification in streaming pmu data. In: 2021 IEEE madrid powertech, powertech 2021 - conference proceedings. Institute of Electrical and Electronics Engineers Inc.; 2021.
[12] Khaledian E, Pandey S, Kundu P, Srivastava AK. Real-time synchrophasor data anomaly detection and classification using isolation forest, KMeans, and LoOP. IEEE Trans Smart Grid 2021;12(3):2378–88.
[13] Dong Y, Japkowicz N. Threaded ensembles of autoencoders for stream learning. Comput Intell 2018;34(1):261–81.
[14] Saurav S, Malhotra P, Vishnu TV, Gugulothu N, Vig L, Agarwal P, et al. Online anomaly detection with concept drift adaptation using recurrent neural networks. In: ACM International Conference Proceeding Series. Association for Computing Machinery; 2018, p. 78–87.
[15] Yu K, Shi W, Santoro N. Designing a streaming algorithm for outlier detection in data mining—An incrementa approach. Sensors 2020;20(5):1261.
[16] Zang W, Zhang P, Zhou C, Guo L. Comparative study between incremental and ensemble learning on data streams: Case study. J. Big Data 2014;1(1).
[17] Pandey S, Srivastava AK, Amidan BG. A real time event detection, classification and localization using synchrophasor data. IEEE Trans Power Syst 2020;35(6):4421–31.
[18] Ahmed A, Sajan KS, Srivastava A, Wu Y. Anomaly detection, localization and classification using drifting synchrophasor data streams. IEEE Trans Smart Grid 2021;12(4):3570–80.
[19] Amutha AL, Uthra RA, Roselyn JP, Brunet RG. Streaming data classification using hybrid classifiers to tackle stability-plasticity dilemma and concept drift. In: 4th IEEE conference on information and communication technology. 2020.
[20] Hasselmo ME. Avoiding catastrophic forgetting. Trends in Cognitive Sciences 2017;21(6):407–8.
[21] Masana M, Liu X, Twardowski B, Menta M, Bagdanov AD, van de Weijer J. Class-incremental learning: survey and performance evaluation on image classification. 2020, Preprint arXiv:2010.15277.
[22] De Lange M, Aljundi R, Masana M, Parisot S, Jia X, Leonardis A, et al. A continual learning survey: Defying forgetting in classification tasks. IEEE Trans Pattern Anal Mach Intell 2022;44(7):3366–85.
[23] Liu Y, Meliopoulos AP, Tan Z, Sun L, Fan R. Dynamic state estimation-based fault locating on transmission lines. IET Gener. Transm. Dist. 2017;11(17):4184–92.
[24] Pellegrini L, Graffieti G, Lomonaco V, Maltoni D. Latent replay for real-time continual learning. In: IEEE International Conference on Intelligent Robots and Systems. Institute of Electrical and Electronics Engineers Inc.; 2020, p. 10203–9.
[25] Maltoni D, Lomonaco V. Continuous learning in single-incremental-task scenarios. Neural Netw 2018;116:56–73.
[26] Mundt M, Lang S, Delfosse Q, Kersting K. CLEVA-compass: A continual learning evaluation assessment compass to promote research transparency and comparability. 2021, Preprint arXiv:2110.03331.
[27] van de Ven GM, Siegelmann HT, Tolias AS. Brain-inspired replay for continual learning with artificial neural networks. Nat Commun 2020;11(1):1–14.
[28] Prabhu A, Torr P, Dokania P. Gdumb: A simple approach that questions our progress in continual learning. In: The European conference on computer vision. ECCV, 2020.
[29] Li Z, Hoiem D. Learning without forgetting. IEEE Trans Pattern Anal Mach Intell 2018;40(12):2935–47.
[30] Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, et al. Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci USA 2016;114(13):3521–6.
[31] Tang B, Matteson DS. Graph-based continual learning. 2020, Preprint arXiv:2007.04813.
[32] Zenke F, Poole B, Ganguli S. Continual learning through synaptic intelligence. In: 34th international conference on machine learning. vol. 8, International Machine Learning Society (IMLS); 2017, p. 6072–82.
[33] Ebrahimi S, Elhoseiny M, Darrell T, Rohrbach M. Uncertainty-guided continual learning in Bayesian neural networks – Extended abstract. In: IEEE computer society conference on computer vision and pattern recognition workshops. vol. 2019-June, 2019, p. 75–8.
[34] Titsias MK, Schwarz J, Matthews AGdG, Pascanu R, Teh YW. Functional regularisation for continual learning with gaussian processes. 2019, Preprint arXiv:1901.11356.
[35] Save N, Popov M, Jongepier A, Rietveld G. Pmu-based power system analysis of a MV distribution grid. In: 24th international conference on electricity distribution, 12–15 June 2017, Paper 1035.
[36] Pan S, Morris T, Industrial UAITo, 2015 u. Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data. Ieeexplore.Ieee.Org.
[37] Lüders B, Schläger M, Korach A, Risi S. Continual and one-shot learning through neural networks with dynamic external memory. In: Lecture notes in computer science. Lecture notes in artificial intelligence and lecture notes in bioinformatics, vol. 10199 LNCS, Springer Verlag; 2017, p. 886–901.