TOWARDS REALTIME DISTRIBUTED VIRTUAL FLOW METER VIA COMPRESSED CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A robust-accurate estimation of fluid flow is the main building block of a distributed virtual flow meter. Unfortunately, a big leap in algorithm development would be required for this objective to come to fruition, mainly due to the inability of current machine learning algorithms to make predictions outside the training data distribution. To improve predictions outside the training distribution, we explore the Continual Learning (CL) paradigm for accurately estimating the characteristics of fluid flow in pipelines. A significant challenge facing CL is the concept of catastrophic forgetting. In this paper, we provide a novel approach of how to address the forgetting problem via compressing the distributed sensor data to increase the capacity of CL memory bank using a compressive learning algorithm. Through extensive experiments, we show that our approach provides around 8% accuracy improvement compared to other CL algorithms in the real-field distributed sensor dataset. Noticeable accuracy improvement is also achieved when using our proposed approach with the CL-benchmark datasets, achieving state-of-the-art accuracies of 94.95% and 77.27% for the MNIST and CIFAR-10 datasets, respectively.

1 INTRODUCTION

The distributed virtual flow meter can provide a game-changing functionality in the oil and gas industry, and other industries requiring accurate characterisation of fluid flows. With a distributed measurement capability applied to oil and gas production, it is possible to detect and locate water/gas breakthrough, monitor fractures and pressure drops as they occur, and perform a non-invasive well integrity inspection (Arief et al., 2021). A distributed flow meter can be defined as a metering solution that measures volume, velocity, and the fraction of fluid components in every location in the pipe. The distributed measurements can be in the form of temperature, chemical, strain, or acoustic signals. In this paper, the distributed measurements are acoustic signals acquired using a technology called Distributed Acoustic Sensing (DAS). A DAS system consists of a fiber optic cable wrapped around (or inside) the pipe with one end of the cable attached to the Interrogation Unit (IU). The IU sends light pulses along the glass fibers, and interprets acoustic events around the cable via the Rayleigh backscatter mechanism.

Using the acoustic signals, researchers can characterize the fluid flow inside the pipe including the fluid volume, flow velocity, and phase-fraction of the fluids. Several techniques have been proposed to accurately estimate these quantities from the acoustic signals, including using geophysical formulations (speed of sound, Joule-Thompson Coefficient, acoustic signal cross correlation, Doppler Effect), and data-driven machine learning techniques from kernel function to deep learning based algorithms. The latest work from (Arief et al., 2022) shows that Deep Neural Network (DNN) algorithms can be used to estimate the phase-fraction of the fluids from the acoustic data. Unfortunately, the accuracy of such an approach degrades rapidly when the model tries to infer from the data outside its training distribution; the situation is known as the Out of Distribution (OOD) problem. Unfortunately, OOD situations are common for real world application of modelling the DAS data.

In this paper, we explore the potential of using a different machine learning paradigm that can work well with the OOD problem. Instead of using fixed model for inference, our proposal uses a technique called Continual Learning or CL (Mai et al., 2022). In the CL approach, the model is trained in a continuous manner, therefore the model is always being adapted as unfamiliar fluid

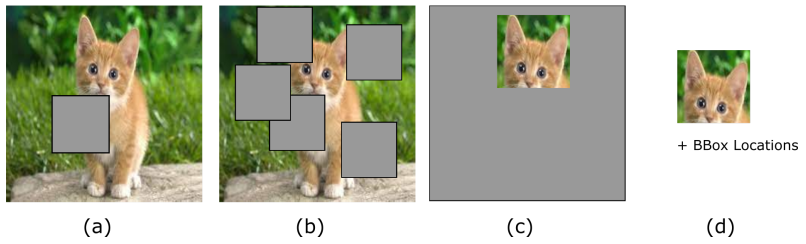

Figure 1: Cutout based algorithms, including (a) Vanilla cutout, (b) Multi-Step Cutout with $n = 5$, (c) Inverse-cutout, and (d) Compressed output of the inverse-cutout for the compressed learning algorithm.

dynamic situations are encountered. The CL model never stops being trained and newer data paired with their label are (continuously) fed to update the model parameters, even if the new data come from OOD events. For a real world implementation, the CL approach is suitable as a building block of a distributed flow meter, because pointwise labels characterising the fluid flow events are often available during oil production thanks to a test separator instrument. The labels the separator provides, can be used to update the trained model in a continuous manner.

Unfortunately the CL approach has its own shortcomings, the most critical one being catastrophic forgetting. The forgetting problem occurs when the updated CL model learns new task but forgets the previously learned tasks, therefore limiting the range of learned situations for which the model inference can provide accurate estimations. Several algorithms have been proposed to address the catastrophic forgetting problem, for example by enhancing regularization (Kirkpatrick et al., 2017; Ahn et al., 2019; Gomez-Villa et al., 2022), by using parameter isolation (Chen et al., 2020; Feng et al., 2021; Yan et al., 2022), and by rehearsing samples from a memory bank (Chaudhry et al., 2018; Prabhu et al., 2020; Bang et al., 2021). In this paper, we explore the last approach by using memory rehearsal to address the forgetting problem. This approach not only provides high accuracy in the CL benchmarks, but also provides seamless adaptation for non-common data types (including the DAS data).

Several CL-based algorithms to model the DAS data for distributed fluid flow estimation are explored in this paper, including GDumb (Prabhu et al., 2020), Riemannian Walk (Chaudhry et al., 2018), iCaRL (Rebuffi et al., 2017), BiC (Wu et al., 2019), and Rainbow Memory (Bang et al., 2021). A fixed memory size is used in the experiments, so that the efficacy of each algorithm on selecting training samples and on managing memory usage can be fairly compared. Unfortunately, given the complexity and the size of the DAS data, most of the algorithms failed to provide satisfactory results. To address the data size and memory problem, we present and test a novel sample compression algorithm that allows for the rehearsal training to store representative data within the fixed size storage. The algorithm uses the inverse-cutout technique to compress and isolate the most relevant part of the input data, and use them for rehearsal, see Fig. 1.

The experimental results show that our proposed algorithm achieves a significant improvement over other CL algorithms. Using the real-field DAS data, we obtain around 8% accuracy improvement and a lower forgetting value compared with other CL algorithms, which paves the way for real world implementation of the algorithm in the distributed fluid flow measurement applications. Even though the algorithm is designed to model and compress the DAS data, it also provides state-of-the-art accuracies on other CL benchmark datasets, including the MNIST and CIFAR datasets.

Our contributions in this paper are the following:

- We develop a novel compressed learning algorithm based on the cutout technique, to search and isolate the most relevant parts of the input data applicable to memory rehearsal.
- We provide a simple and intuitive augmentation technique that enriches image representation for non-natural image datasets.
- We perform extensive experiments on a real world dataset to build a distributed virtual flow meter based on DAS and machine learning CL paradigm.

- We show that our proposal can also be used for natural image datasets by achieving state-of-the-art accuracies on the CL benchmark datasets.

2 Background

Suppose we have streaming DAS data (x, y) coming from an oil-production well, where x denotes the input data in the form of raw or transformed DAS data, and y denotes the fluid flow measurement from the well separator; y can be in the form of measurements for oil-volume, gas-volume, water-volume, or combinations of them. Estimating y given x is a multihead regression task, but due to the limited research on continual learning in regression tasks, the estimation here is structured as a classification task using the bin-based regression approach with a number of classes, C.

2.1 Continual Learning

For a classification task, the Deep Neural Network (DNN) model minimizes the empirical error by measuring the difference between the class estimation likelihood with the actual class value, and it assigns a higher penalty when the model estimates the wrong class with high likelihood. The Cross-entropy (CE) loss function is used to measure the DNN classification error, and is defined as:

$$L = -\sum_{i}^{C} y_i \log(s_i), \tag{1}$$

where C denotes the number of classes, s_i denotes the likelihood-score of class i, while y_i denotes the the correct label for class i. Both the DNN model and CL-based DNN model are trained using the CE loss. The main difference between the two is the way the accuracy is measured for each model. The accuracies for the DNN model are measured using the mean-class accuracy and the mean-total accuracy defined at (He et al., 2016), while the CL model accuracies are measured using the highest accuracy from the final task and the average degradation of accuracies over multiple tasks, see (Lopez-Paz & Ranzato, 2017). Note that the CL model is trained on a time-filtered dataset. This setting tries to imitate real world streaming data where each time-filtered set represents a different class distribution (in-time) from the other sets.

In this paper, the time-filtered set is referred to as task (t) where each task consists of several classes. The task can be structured as a disjoint task, where the classes in each task are disjoint with the other classes in the other tasks. This setup is referred to as a disjoint CL. To represent a more realistic real world phenomenon, the CL approach can also be structured as a blurry CL, where the same class can exist in two or more tasks.

It is worth noting that both blurry or disjoint CL can be trained in an online or offline fashion. Online training means that the CL model only reads the streaming data once, while the offline training reads the streaming data multiple times until it reaches satisfactory accuracy. Furthermore, both the online and offline training can have a memory bank. The memory bank stores the samples of selected streaming data. When needed, the stored data can be rehearsed to update the CL model to avoid catastrophic forgetting. In this paper we focus on the offline-blurry CL setting because it has similar characteristics with the real world implementation of the oil-production environment. We refer the reader to (Bang et al., 2021) for an in-depth explanation of different CL settings.

Catastrophic forgetting occurs when the CL model learns new tasks but forgets the previously learned tasks. This is a common phenomenon in the CL training due to the heavy reliance on the iterative learning process in many machine learning algorithms, including the DNN-based algorithms. The iterative process uses incremental updates based on the new data for updating its trainable parameters. Therefore, the longer the model trains on the new data, the larger the bias towards the new tasks. The CL approaches can be divided into three categories based on the way the algorithms address the forgetting problem. They are based on (1) Regularization, (2) Parameter isolation strategy, and (3) Memory management. The regularization approaches control the network parameter updates to mitigate the forgetting, either by introducing additional loss penalties or constraining the parameter updates by changing the parameter gradients during optimization. On the other hand, the parameter isolation strategy, limits the parameter updates only to a selected range of useful parameters while keeping the other parameter unchanged. Another type of parameter isolation strategy is by adding new parameters to the existing architecture to increase the network's ability to learn new information without forgetting old one. Finally, the third approach is by using memory bank and rehearsal. As the term suggests, this approach stores the old streaming data to a temporary memory, and when the model is introduced to new data, the old data is also rehearsed either together with the new data or subsequently during optimization.

2.2 Memory Based Continual Learning

The general framework of the memory based CL algorithm adapted from (Mai et al., 2022) is shown in Appendix A. The algorithm relies on two main functionalities, namely a sampling procedure and a rehearsal process. The sampling procedure allows the algorithm to selectively choose which data to store in the limited space of the memory bank, while the rehearsal process controls how the old data are being used to update the existing model. Several algorithms based on the rehearsal and memory approaches have been proposed to address the catastrophic forgetting problem, including Greedy Sampler and Dumb Learner (GDumb) (Prabhu et al., 2020), Bias Correction method (BiC) (Wu et al., 2019), and Rainbow Memory (RM) (Bang et al., 2021).

The GDumb approach focuses on a sampling procedure according to which the spread of class distribution in the memory bank is balanced. A full retraining with the updated data before performing inference is then performed. This approach is simple and intuitive, and has shown considerable performance gains in the CL benchmark. The BiC approach, introduces an additional final layer into the DNN model for calibrating the training bias. Finally, the RM approach proposes the use of Monte-Carlo (MC) based uncertainty measurements to better select which samples to store in the memory bank, and provides an extensive Data Augmentation (DA) proposal to enrich the replayed results.

2.3 Evaluation Metrics

The CL model is evaluated by using several metrics. In this paper, we use the common CL metrics defined in (Lopez-Paz & Ranzato, 2017), including Final Accuracy (ACC_T) for measuring the model accuracy on classifying the streaming data, Backward Transfer (BWT_T) for measuring the catastrophic forgetting, and Intransigence (I_T) for measuring the gap-accuracy between offline and CL- training.

ACC_T is defined as the average reporting accuracy after the T tasks have been trained, thus it evaluates the final model accuracy after all classes have been exposed to the model. BWT_T measures the effect of learning additional tasks on the predictive capability of a model on the previous tasks. A large negative BWT_T is a sign of catastrophic forgetting. I_T, on the other hand, measures how much on average the accuracy of each task differs compared to the upper-bound accuracy of a model from the non-CL setting. Following the formal definition of the CL metrics at (Lopez-Paz & Ranzato, 2017), denoting by $R \in \mathbb{R}^{T \times T}$ of the matrix of task accuracies, the ACC_T and BWT_T metrics are defined as:

$$ACC_T = \frac{1}{T} \sum_{i=1}^{T} R_{T,i}, . \tag{2}$$

$$BWT_T = \frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}, \tag{3}$$

where the $R_{i,j}$ denotes the intermediate accuracy of a model on task t_j after observing all samples from task t_i. Finally, with ACC_{joint} denoting the upper-bound accuracy from the standard offline training (non-CL setting), I_T is defined as:

$$I_T = ACC_{joint} - ACC_T. \tag{4}$$

3 Compressed-Continual Learning

The CL memory algorithms optimize the process of data selection (for memory storage) while also providing effective rehearsal methods. Those algorithms provide a competitive result for modelling the image data, including MNIST and CIFAR dataset. Unfortunately, the DAS data is significantly

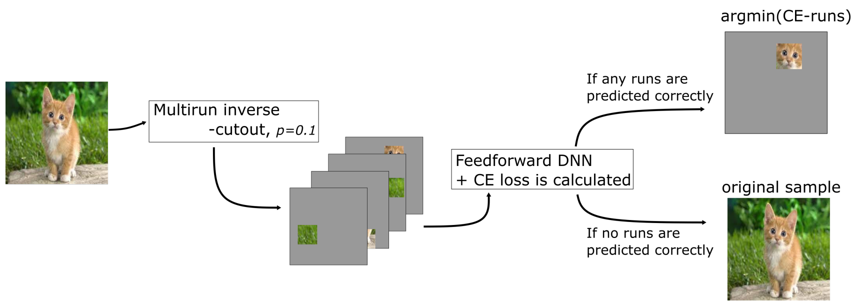
Multirun inverse -cutout, p=0.1
Feedforward DNN + CE loss is calculated
If any runs are predicted correctly
argmin(CE-runs)
If no runs are predicted correctly
original sample

Figure 2: The cutout-CL pipeline for generating compressed samples and selecting non-compressible samples.

different from the image data in terms of size and complexity. The size of the DAS data is several orders of magnitude higher than the image data and the data itself has a lot of redundancy while containing considerable amount of noise (Arief et al., 2021). In the next section, we elaborate on why the existing memory-rehearsal algorithms would not work on the DAS data, and then explain how our proposal addresses the memory issues. Finally, we show how to train compressed DAS data using our proposed algorithm.

3.1 Rethinking Rehearsal on DAS

Rehearsal memory aims to allow the machine learning model to remember previously learned tasks by storing and retraining representative datasets with a fairly balanced distribution from the memory bank. This approach assumes that the data size is small enough, therefore enough representations can be stored within the available memory. The large size of DAS datasets presents a challenge. 10 seconds of DAS data representing 1-data point requires (around) 1.2 GB of memory, while a spectrogram DAS data containing similar information, requires around 120 MB of memory. For bin-based regression with 72 classes, the memory requirement for storing at least 10-representations per class is extensive. As the complexity of data representation (i.e. the changes in temperature, pressure, humidity, presence of noise, and others) increases, the number of representations in the memory bank required to overcome forgetting increases dramatically. With the current state of memory-based CL algorithms, storing large data with enough representations in the memory bank, is not feasible. This is not only due to the limited amount of memory but also due to the limitations in the augmentation strategies available for non-natural image type datasets.

3.2 Multi-step Cutout Augmentation

It has been pointed out in (Bang et al., 2021) that DA plays a crucial role on memory rehearsal because it can significantly increase the amount of sampled data through augmentation. The RM algorithm enjoys a significant accuracy improvement from extensive implementation of the DA strategy. Unfortunately, most of the DA techniques for natural images are not applicable for the DAS data because those techniques assume that the image data is invariant under translation, color-inversion, rotation, and even data blending operations. Unfortunately, the DAS data is not invariant under these operations.

Here, we use a simple but intuitive DA strategy that can be applied to non-natural image data, such as DAS data. The algorithm is called Multi-Step Cutout; a more extensive version of the Cutout algorithm introduced in (DeVries & Taylor, 2017). The cutout algorithm uses a fixed size window randomly placed on the input data to remove (zeroing) the overlapping area between the window and the input data, and uses the modified data as the augmentation output. The Multi-Step Cutout, on the other hand, moves the cutout window multiple times (n_cut times) on the input data randomly, while zeroing the overlapping location each time the window moves. For the rest of the paper, the number of cut windows used in this augmentation technique is denoted as n_cut. Fig. 1 depicts the difference between the cutout and multi-step cutout outputs.

The motivation behind the Multi-Step Cutout is to teach the model to utilize the minimum (but) relevant part of the input data responsible for making the decision on alleviating most of the noise in the input data, therefore it can generalize better. This approach is simple and applicable to non-natural image data for augmentation because the algorithm does not change the structure within the data itself.

3.3 Cutout Compressed Learning

One solution to addressing the data size problem is by compressing the input data. A classical *jpeg* compression is used by (Wang et al., 2022) to allow more data in the memory bank, and has shown an improvement CL performance. Unfortunately, this image compression technique is not suitable to compress the DAS data, due to the nature of the data itself. In this section, we provide a novel approach on how we can compress the DAS data while preserving its ability to be used for rehearsal. We henceforth refer to this approach as Cutout Compressed Learning (Cutout-CL).

The Cutout-CL algorithm works by finding the most relevant part of the input data for performing classification, and only stores that part for memory rehearsal. The most relevant part is defined as the part that occupies the smallest area of the input data but gives the lowest likelihood error (from Eq. 1). The Multi-Step Cutout algorithm is used for training the CL-model to enhance augmentation as well as for maximizing the utilization of the most relevant part of the input data. An inverse-cutout algorithm is (then) used for finding the most relevant part of the data. Instead of zeroing the overlapping window like in the cutout algorithm, the inverse-cutout leaves the overlapping part unchanged and zeroes all the other non overlapping part of the input data. The inverse-cutout uses one parameter (p) defined as the percentage of data to leave-out for compression. This approach is not only efficient for isolating important parts of the input data for sampled compression by only storing the leave-out part, but also provides fast indexing for recovery by only storing the isolated part and the bounding box location of the isolated part. By utilizing the compressed samples, the memory size in our proposal can be increased by p^{-1}, therefore the new memory size K_n is defined as K_n = K x p^{-1}, where K denotes the initial memory size.

The proposed compression technique uses both the Multi-Step Cutout and Inverse-Cutout algorithms, and is implemented as prescribed in the following. First, the model is trained using the Multi-Step Cutout augmentation; for natural image datasets, the first step is optional due to the abundance of DA techniques for natural images. Second, at the end of each task, multiple runs of the Inverse-Cutout algorithm are applied for each sample; here, the CE score for each run is also calculated. For each sample-run with correct classification, the run with minimum CE score is used to update the corresponding sample in the sample selection, replacing the non-compressed version of the sample. It should be noted that samples without correct classification in the multiple runs, denoted as non-compressed samples, are given low-priority for the episodic memory selection. Fig. 2 depicts the implementations of the Cutout-CL pipeline for selecting and generating compressed and non-compressed samples. For the episodic memory selection, the compressed samples are selected randomly based on the number of maximum number of samples per class (n_c). It is important to mention that the number of samples per class needs to be balanced to avoid imbalanced class prediction. The memory selection for each class is shown in Algorithm 1.

Algorithm 1: Compressed Learning Memory Selection

```
Input : Compressed samples $X1$, Non-Compressed samples $X2$, Number of samples per class $N_t$,
Compression rate $p$, Total number of seen classes $C_t$;
Initialize : Memory $M \leftarrow \{\} * M$;
1: for <$c = 0, 1, 2, ..\ C_t$> do                                  ▷ Iterate through all seen classes.
2:     $n_c,\ X1_c,\ X2_c \leftarrow N_t[c],\ X1[c],\ X2[c]$     ▷ Populate data from the compressed learning process.
3:     $M \leftarrow M + RandSelect(X1_c,\ min(n_c,\ len(X1_c)))$     ▷ Random sampling through the Compressed samples.
4:     $n_c \leftarrow (n_c - min(n_c,\ len(X1_c))) * p$     ▷ Update the maximum number of available spaces for the Non-Compressed samples.
5:     $M \leftarrow M + RandSelect(X2_c,\ min(n_c,\ len(X2_c)))$     ▷ Random sampling through the Non-Compressed samples.
```

(a) MNIST Images. (b) CIFAR-10 images.

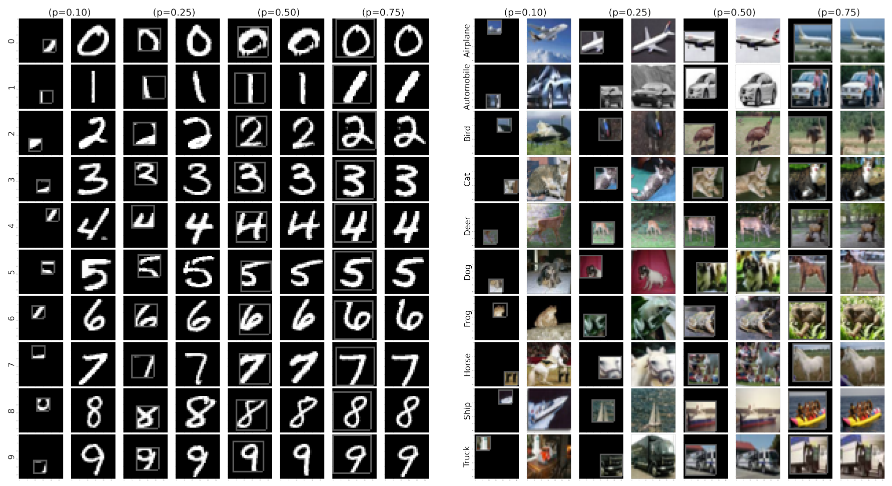

Figure 3: The cutout-CL outputs where the leave-out percentage is defined by p. The rows show side-by-side depiction of the compressed and original images, while the columns represent different classes.

3.4 UNDERSTANDING CUTOUT-CL

At the high level, our proposal works similarly to the random selection episodic memory algorithm (Bang et al., 2021). The main difference is that the Cutout-CL benefits from the extra space the compressed data provides. It has been well established that an increase in the memory space for CL-learning leads to an increase in final accuracy and better handling of the forgetting problem. Moreover, it has also been pointed out in (Selvaraju et al., 2017), as well as in the attention-based DNN (Vaswani et al., 2017) that a machine learning model builds its feature representation by highlighting the specific part of the input data and (tends) to ignore the whole space of the input data. Our proposal capitalizes on this knowledge, by providing a simple but intuitive compression technique that provides more sample spaces in the fixed memory bank by isolating only the relevant part of the input data. Therefore, we hypothesize that ”by only storing relevant parts of the data and using them for retraining a model, we can acquire a similar (or even better) model than a model trained using the full-size of input data”.

The compression rate in our proposal is controlled by p. A small p yields a larger pool of memory but it comes with a cost that the small p often cannot isolate all the relevant details in the images. Fig. 3 shows the behaviour of compressed learning output towards different values of p on different type of datasets. Optimizing the value of p requires understanding of the nature of each class in the dataset. For simplicity, in this paper we use a fixed value p for all classes, and show that even with a fixed p and random selection of samples, a reasonable accuracy improvement can be achieved over the other CL-algorithms.

4 EXPERIMENTS

4.1 DATA SOURCE AND PREPARATIONS

The DAS dataset we used in this paper was from two-phase gas production with varying gas volume fractions acquired in an offshore production well. The reference data (oil and gas volumes) was acquired using a test separator measurement located on the platform topside. For the CL-setup, we used bin-based regression from the combination of different range of values for both oil and gas, resulting in 72 classes.

To simplify the training process, the DAS data was transformed to stacks of spectrogram images with size of (256, 486, 576) with the first, second, and third dimensions represent spatial, frequency,

Table 1: Experiment results from offline and continual learning using DAS-Spectrogram dataset on different augmentation settings

(a) Offline learning results in ACC_{joint}

n_cut = 1	**n_cut = 5**	**n_cut = 20**
40.83	46.45	**48.08**

(b) CL learning results

Methods	ACC_8	BWT_8	I_8
n_cut = 1	24.11	-40.66	**16.72**
n_cut = 5	25.44	-40.16	21.01
n_cut = 20	**29.73**	**-39.51**	18.35

and time domains, respectively. Each spectrogram contains 10-seconds of DAS recording at 10 KHz sampling rate, and 256 spatial samples at around 1 m sampling rate. A 50% overlapping window both in time and space is used to enrich the preprocessing output. For the CL-setup, the 72 classes were spread out into 8 tasks randomly using blurry10 setup. We refer the reader to (Bang et al., 2021) for details on how the blurry setup is performed.

4.2 Continual Learning on DAS Data

In this paper, we used a ResNet18 architecture (He et al., 2016) to train the spectrogram DAS data. The main objective is to find out how well our proposed approach performs on the CL-setting. Unless otherwise mentioned, we used initial learning rate of 0.001, memory size of 1000, batch size of 6, cutout size of 25% of initial input data, and maximum 256 training epochs per tasks. To conserve training time for each task, the training was moved to the next task after there was no training-accuracy improvement over 4 consecutive epochs. Additionally, due to the slow training time, only 10% of uniformly sampled DAS data was used for the experiment. Training 8 CL-tasks with over 10% of DAS data required (on average) more than 1 week on a single 16GB NVIDIA P100.

Augmentation results. We started the experiments by measuring the upper bound or joint accuracy (ACC_{joint}) on the DAS data. Table 1a shows the results of the joint accuracy on several different n_cut settings of the Multi-Step Cutout Augmentation. The result confirms the usability of such augmentation algorithm on the spectrogram data by providing around 5-8% on the overall accuracy improvement. We also performed the same augmentation technique on the CL-setting using random search for sample selections; the result is presented in Table 1b. Even though, the improvement is smaller compared to the Offline-setting, the proposed augmentation performs well on the CL-setting with n_cut=20 providing the highest ACC_8 and BWT_8 of 29.73% and -39.51%, respectively. It is worth mentioning that n_cut=1 achieves the lowest I_8 due to the low ACC_{join} n_cut=1 setting achieve.

Table 2: Spectrogram-DAS CL accuracy

Methods	ACC_5	BWT_5	I_5
RM	24.85	-40.50	23.23
RWalk	25.89	-	22,19
EWC	25.15	-	22.93
<u>Cutout-CL</u>			
No-Aug*	27.22	-42.66	20.86
$p = 0.05$	**32.10**	**-28.05**	**15.98**
$p = 0.10$	29.73	-32.49	18.35
$p = 0.25$	26.63	-42.90	21.45

Cutout-CL results. Following the results on Augmentation, we then performed Cutout-CL on multiple compression rates from $p = 0.05$ to $p = 0.50$. We used n_cut=20 of Multistep augmentation as the augmentation technique, and report the results in Table 2. Additionally, we also performed Cutout-CL with $p = 0.25$ without Multistep Augmentation as a comparison with other

Table 3: Last Accuracies on (a) MNIST and (b) CIFAR-10 datasets

(a) Results on MNIST dataset

Methods	ACC_5	I_5
GDumb	88.21 ± 0.63	10.03 ± 0.61
BiC	91.11 ± 0.13	7.13 ± 0.05
RM	94.35 ± 0.36	3.88 ± 0.34
EWC	89.51 ± 1.51	8.72 ± 1.60
Cutout-CL		
$(p = 0.25)$	89.19 ± 0.51	9.05 ± 0.60
$(p = 0.50)$	94.51 ± 0.97	3.73 ± 1.08
$(p = 0.75)$	**94.95 ± 0.82**	**3.29 ± 0.93**

(b) Results on CIFAR-10 dataset

Methods	ACC_5	I_5
GDumb	44.87 ± 0.82	48.95 ± 1.04
BiC	60.89 ± 5.13	32.93 ± 4.94
RM	72.27 ± 1.31	21.55 ± 1.08
EWC	74.12 ± 1.51	19.70 ± 1.28
Cutout-CL		
$(p = 0.10)$	**77.27 ± 0.58**	**16.55 ± 0.45**
$(p = 0.25)$	75.08 ± 0.93	18.74 ± 0.70
$(p = 0.50)$	73.91 ± 0.79	19.91 ± 0.67

CL-algorithms. The results in Table 2 confirm our hypothesis that we can generate a quality model by combining the Multi-Step Augmentation with cutout compression to perform CL-learning. Our proposal with $p = 0.05$ achieves the highest accuracy of 32.10, which is only 15.98 point different from the offline accuracy of 48.08 from Table 1a. It is interesting to note that considering our baseline accuracy from n_cut=1 of 24.11, the increase in overall improvement is around 8%.

4.3 Continual Learning on Other Datasets

In order to compare our proposal with other CL-algorithms, we also experimented with CL-benchmark datasets, such as the MNIST and CIFAR-10 datasets. We follow the data preparation and hyper-parameter settings in (Bang et al., 2021); additional settings include K=500 trained on disjoint, blurry10, and blurry30 settings with Cutmix (Yun et al., 2019) and AutoAug (Cubuk et al., 2019) as the DA techniques; the results are presented in Table 4 in Appendix B. Our proposal with $p = 0.1$ achieves state-of-the-art accuracies both for blurry10 and blurry30 of 80.83 ± 0.53 and 88.91 ± 0.20, respectively. It is worth noting that the results from other algorithms in Table 4 were populated from (Bang et al., 2021) which uses the highest test accuracies from the final task as the reporting accuracies.

We also reproduced the results both for MNIST and CIFAR-10 and used the last epoch accuracies as the reporting accuracies. The results are presented in Tables 3a and 3b both for MNIST and CIFAR-10 datasets, respectively. The results show that our proposal achieves the highest accuracies of 94.95 ± 0.82 and 77.27 ± 0.58 for MNIST and CIFAR-10 datasets, respectively. Interestingly, the values of the compression rate p need to be adjusted properly due to the different behaviour on how objects in the two datasets are represented. It requires at least 50% of the whole area of the image to fully isolate the main object in the MNIST images. In CIFAR-10, however, even when isolating 10% of the image area, the classifier can recognize the object in the image correctly. Fig. 3 shows how the compression rate affects the output between the two datasets.

5 Conclusions

We have presented the Cutout-CL, an iterative compressed continual learning pipeline based on the inverse-cutout technique to isolate only the relevant part of the input data and use them for rehearsal, increasing the available memory space in the process. In addition, we also proposed a simple and intuitive data augmentation method applicable to non-natural image datasets. Throughout extensive evaluation, we have shown that our proposals provide significant improvements in term of higher ACC_T and BWT_T for modelling the DAS data, aiming to deliver a robust distributed virtual flow meter application.

Additionally, using the CL-benchmark datasets, including MNIST and CIFAR-10 datasets, we have shown that our proposal can also be applied to such datasets by achieving state-of-the-art accuracies in both datasets. Finally, using a relatively small p, we have shown that the experimental evidence supports our hypothesis that by only storing relevant parts of the data and using them for retraining a model, we can acquire a better model than the model trained using the full-size of input data.

REFERENCES

Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 4392–4402, 2019.

Hasan Asy'ari Arief, Peter James Thomas, and Tomasz Wiktorski. Better modelling out-of-distribution regression on distributed acoustic sensor data using anchored hidden state mixup. *IEEE Transactions on Industrial Informatics*, 2022.

Hasan Asy'ari Arief, Tomasz Wiktorski, and Peter James Thomas. A survey on distributed fibre optic sensor data modelling techniques and machine learning algorithms for multiphase fluid flow estimation. *Sensors*, 21(8):2801, 2021.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8218–8227, 2021.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Hung-Jen Chen, An-Chieh Cheng, Da-Cheng Juan, Wei Wei, and Min Sun. Mitigating forgetting in online continual learning via instance-aware parameterization. *Advances in Neural Information Processing Systems*, 33:17466–17477, 2020.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Yingchao Feng, Xian Sun, Wenhui Diao, Jihao Li, Xin Gao, and Kun Fu. Continual learning with structured inheritance for semantic segmentation in aerial imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–17, 2021.

Alex Gomez-Villa, Bartlomiej Twardowski, Lu Yu, Andrew D Bagdanov, and Joost van de Weijer. Continually learning self-supervised representations with projected functional regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3867–3877, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision*, pp. 524–540. Springer, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Liyuan Wang, Xingxing Zhang, Kuo Yang, Longhui Yu, Chongxuan Li, HONG Lanqing, Shifeng Zhang, Zhenguo Li, Yi Zhong, and Jun Zhu. Memory replay with data compression for continual learning. In *International Conference on Learning Representations*, 2022.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

Qingsen Yan, Dong Gong, Yuhang Liu, Anton van den Hengel, and Javen Qinfeng Shi. Learning bayesian sparse networks with full experience replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 109–118, 2022.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

A Memory Based CL algorithm

Algorithm 2: General step of memory-based CL

```
Input : DAS data $x$, Label $y$, Parameters $\theta$;
Initialize : Memory $M \leftarrow \{\} * M$; Batch size $b$;
1: $x_M, y_M \leftarrow$ MemoryRetrieval($M$, $b$)
2: $X, Y \leftarrow x \cup x_M, y \cup y_M$
3: $\theta \leftarrow$ ModelUpdate($X, Y, \theta$)
4: $M \leftarrow$ MemoryUpdate($M$, $x$, $y$)
```

B Benchmark Results of CL-algorithms on CIFAR-10

Table 4: ACC_5 of CL-algorithms on CIFAR-10

Methods	**Disjoint**	**Blurry10**	**Blurry30**
GDumb	42.47 ± 1.15	43.16 ± 0.77	45.72 ± 0.64
BiC	59.53 ± 4.30	61.45 ± 6.25	71.93 ± 2.45
RM	61.91 ± 0.63	76.86 ± 0.04	85.10 ± 0.16
RWalk	65.04 ± 0.11	78.59 ± 1.37	85.18 ± 0.57
iCaRL	**65.61 ± 2.57**	57.07 ± 2.74	64.90 ± 7.95
EWC	64.00 ± 1.34	78.67 ± 1.06	85.00 ± 0.42
Cutout-CL ($p = 0.1$)	65.43 ± 2.83	**80.83 ± 0.53**	**88.91 ± 0.20**