ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

Compression-based kNN for Class Incremental Continual Learning

Valerie Vaquet, Jonas Vaquet, Fabian Hinder, and Barbara Hammer *

Machine Learning Group
Bielefeld University, Bielefeld - Germany

Abstract. Catastrophic forgetting is a key challenge in continual learning. In the adjoining field of stream machine learning, few methods target the related problem of re-occurring drift by avoiding forgetting old data. In this work, we investigate whether we can transfer such strategies from the stream machine learning to the continual learning setup. Based on our consideration, we propose a simple yet efficient compression-based kNN scheme and evaluate it experimentally.

1 Introduction

Most work on machine learning algorithms considers the batch setup where the data is generated by a static underlying data distribution making all information available at training time. This assumption does not hold in many real-world scenarios, like in natural and industrial processes where we observe data over time and have to account for different types of non-stationarities [1, 2]. Work in this area can be categorized in the field of stream machine learning [1] and continual learning [3].

The goal of *stream machine learning* (SML) is to keep an accurate model representing the current data distribution well. Usually, a stream of instances arriving one by one is considered whereby the data-generating distribution might change over time – a phenomenon commonly referred to as *concept drift* [1] or drift for shorthand. Drift poses considerable challenges when keeping a model that approximates the most recent concept well. Thus, in this field, simple base classifiers or ensembles thereof constitute common model choices as they can be flexibly adapted requiring only little data [1, 2]. There also exist a few methodologies that are not only flexibly adaptable but equipped to handle re-occurring drift, i.e. the re-occurrence of previously seen patterns [1].

In contrast, *continual learning* (CL) leverages deep neural networks. The goal is to train a model that discriminates all data seen so far. While there are different CL setups, generally one considers data arriving as a stream of batches where different batches contain distinct tasks, e.g. data from different classes. One major challenge of CL is *catastrophic forgetting* where information learned on earlier batches is eroded in the training process of the more recent ones [3, 4]. A considerable body of work investigates how to mitigate catastrophic forgetting, for instance by using replay strategies [4].

In this work, we will investigate another strategy: We explore whether and how we can transfer strategies avoiding forgetting in SML to the CL setup. More

*Funding in the scope of the BMBF project KI Akademie OWL under grant agreement No 01IS24057A and the MKW NRW project SAIL under grant agreement No NW21-059A is gratefully acknowledged.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

precisely, we focus on the Self-Adjusting Memory kNN (SAM-kNN) model [5] which is particularly suitable for re-occurring drift due to its dedicated memory storing old concepts. A first challenge in this investigation is the inability of SML algorithms to handle high-dimensional data such as images adequately. We can rely on a previously proposed pipeline combining deep foundation models with SML algorithms that enables SML models to process images effectively [6].

This paper is structured as follows. After summarizing the target setup of CL and recapping the SAM-kNN model in Section 2, we examine which characteristics prevent us from using SAM-kNN in the CL setup[1] (Sections 3 and 4). Based on our insights we propose a novel compression-based kNN for CL (Section 5) and conclude this work (Section 6).

2 Foundations – Continual Learning and Stream Learning

CL considers data arriving as N consecutive batches D_t. In class incremental CL each batch represents a task t, i.e. a subset of classes. The goal is to learn from the batches and retain all the acquired knowledge. Returning to previously seen batches is not possible. A key challenge is so-called catastrophic forgetting: classes learned on earlier batches are compromised during the training process on the more recent batches [3]. Usually, in CL this challenge is tackled by applying strategies like replay, etc. [4]. CL algorithms are evaluated by considering the accuracies $a_{l,j}$ of the model being evaluated on the j-th task after being trained with the batches of tasks $1, \ldots, l$ ($j \leq l$). These scores can be summarized to the average accuracy $AA_l = \frac{1}{l}\sum_{i=1}^{l} a_{l,i}$. We refer to AA_{End} as the mean accuracy after training on all batches ($l = N$). Besides, there are scores measuring the forgetting, e.g. the maximal accuracy differences $f_{j,l} = \max\{a_{i,j} - a_{l,j} \mid i \leq l\}$ [3]. We denote the averaged forgetting after training on all batches as $AF_{\text{End}} = \frac{1}{N}\sum_{i=1}^{N} f_{N,i}$.

As previously discussed, forgetting earlier concepts is also a challenge for stream learning algorithms that face re-occurring drift [1]. To avoid this behavior [5] proposed the SAM-kNN model. Building upon a simple and quickly adaptable kNN classifier paired with an intelligent memory construction SAM-kNN is well equipped to model the current concept while keeping information on earlier concepts: SAM-kNN contains a size adaptable sliding window representing the most recent samples called short term memory (STM). Whenever predictive power declines, for instance, due to drift, the window is resized such that it only represents the current concept. Older samples are not deleted but stored in the long term memory (LTM) which contains a compressed representation of all data samples not directly contradicting the state of the STM. The LTM is kept in a size limit by applying k-means clustering to its samples whenever the designated storage is exceeded. At prediction time either the kNN prediction of the STM, the LTM, or their combination (CM) is selected based on previous performance of these three options. In experiments this model showed good performance to heterogeneous drifts and in particular to reoccurring patterns [5]. In this contribution, we investigate how we can leverage the strengths of this model in the CL setup.

[1] Code is available at `https://github.com/jvaquet/Compression-Based-kNN`.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

3 SAM-kNN in the Continual Learning Setup

Leveraging the pipeline for applying SML methods to streams consisting of image data [6], we raise

Hypothesis I: Combining deep embeddings with SAM-kNN can be used in the continual learning setup while avoiding catastrophic forgetting.

To test this, we evaluate the performance of SAM-kNN in the setting of class incremental learning. We replicate the experimental setup described in [4]. In this work, we focus on the Split Cifar-100 dataset consisting of 20 tasks with disjoint classes. Each task contains 2,500 images of size $3 \times 32 \times 32$ representing 5 classes. 500 images per task are available for evaluation in a separate test set. We follow the described setup and repeat the experiment 15 times with randomly selected class composition in each run. We use the following pipeline:

Pipeline First, we embed the images using a ResNet50 model which has been pre-trained on ImageNet to obtain a 2,048-dimensional representation. As this is quite high dimensional for a kNN-based model or other simple base learners used for SML algorithms, we perform a PCA and obtain 50-dimensional data points. The PCA has been pre-trained on the tiny ImageNet training set. This preproccessing yields a low-dimensional representation which can be fed to SAM-kNN. To stick to the one-by-one stream processing, we shuffle each task batch and concatenate all of them into one data stream. Evaluation is performed on the test sets provided by [4].

SAM-kNN Since in this case, we aim to obtain a SAM-kNN model which can accurately predict the data from all tasks presented during the training, we skip the cleaning process in which information contradicting the STM is deleted from the LTM.

In our experiments, we obtain an AA_{End} of 0.065 ± 0.088 and an AF_{End} of 0.5155 ± 0.01605. Thus, when applying SAM-kNN with the deep embedding pipeline we observe severe catastrophic forgetting. This is likely due to using the vanilla compression process: As for each class the number of representatives is halved whenever the LTM exceeds its size we have an exponential decay in the class representation, hurting the performance on earlier tasks.

SAM-kNN (balanced) Thus, in a second experiment, we enforce a balanced LTM by always distributing the available storage capacity equally between all classes seen so far in training.

We observe a AA_{End} of 0.107 ± 0.0052 and an AF_{End} of 0.309 ± 0.0056. While this constitutes a considerable improvement and is comparable to the weaker-performing models with a very limited buffer evaluated in [4], the model is still performing poorly. Our observations might be caused by (i) a poor embedding quality, (ii) issues of the kNN-based model in discriminating a large number of classes, or (iii) that SAM-kNN is not suitable for the CL setup. We can exclude options (i) and (ii) by evaluating how well a batch kNN performs on the embedded data representation. Running this experiment we obtain an AA_{End} of 0.251 ± 0.000 which is a reasonable score compared to the well-performing CL models considered in [4]. This leaves us with option (iii) which is additionally the only of the aspects that might explain the forgetting effect. We will investigate this in detail in the next section.

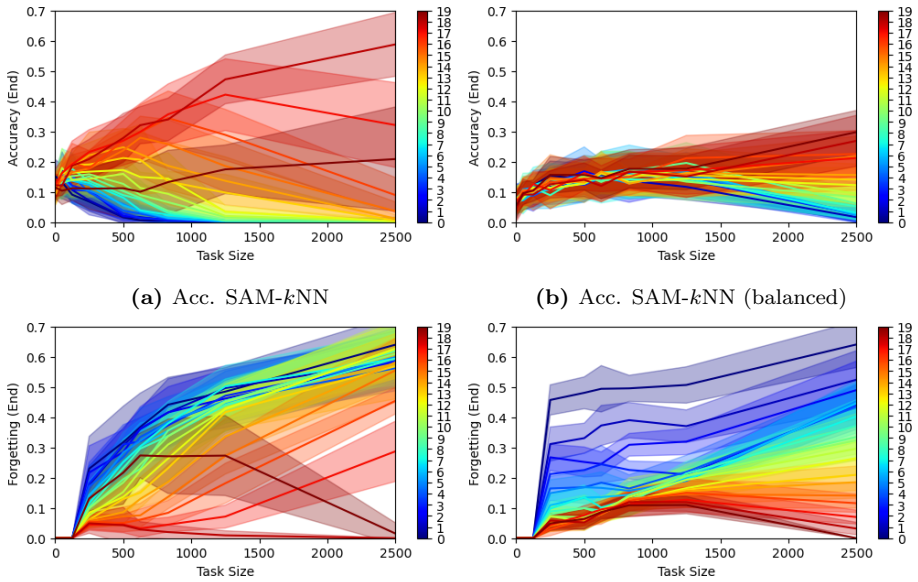
ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

4 The Role of the Task Size

As we already described CL and SML differ in their main objectives but also in the considered setups: instead of considering a potentially drifting data stream in CL different tasks are presented as batches over time. In this section, we evaluate whether this is causing the observed performance decline and the effect of catastrophic forgetting when using SAM-kNN. More precisely, we test whether:

Hypothesis II: Large task sizes cause catastrophic forgetting.

We aim to evaluate the effect of task sizes by interpolating between the setups. A stream of batches used in CL can be adapted to a data stream by decreasing the task batches to very small sizes. To still provide the same amount of data, the tasks will repeat over time. In our experimental evaluation, we consider the range from a task size of 5, i.e. each class being present once, to the considered CL task size of 2,500. To avoid sampling effects we enforce that the single task batches contain the same number of samples per class. The results of our experiments are visualized in Fig. 1. As one can see, for both SAM-kNN and SAM-kNN (balanced) the task-wise end accuracies decrease while the forgetting scores increases with increasing task size, thereby confirming Hypothesis II. We observe a considerably better performance for those tasks that were considered late in training, underscoring the effect of catastrophic forgetting. While this effect is not as pronounced for SAM-kNN (balanced) the obtained accuracies are still not optimal. We suppose that this effect is again connected to the



(a) Acc. SAM-kNN **(b)** Acc. SAM-kNN (balanced)

(c) Forgetting SAM-kNN **(d)** Forgetting SAM-kNN (balanced)

Fig. 1: Results of experiments of the vanilla SAM-kNN and the version with balanced LTM for varying task sizes. Tasks are indicated by colors and are ordered, i.e. task 0 was the first seen in training, task 19 last.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

compression step of the LTM. Repeatably performing clustering might result in unsuitable class representatives as we are increasingly deviating from real class representatives.

5 Compression-based kNN for Continual Learning

Summarizing the experiments presented in the last section, we hypothesize that

Hypothesis III: Applying clustering repeatably is causing a weak performance of SAM-kNN in the CL setup.

Testing this hypothesis, we propose a compression-based kNN model that directly works in the CL setup. Since in task-incremental CL we are presented with batches of tasks, we can eliminate the SML instance by instance processing of SAM-kNN. In this setup, we only need the data to be suitably represented in the LTM and, thus, can discard the STM and the CM. We obtain the following strategy:

Compression-based kNN For each task, we save a compressed representation of each class to the LTM. In this step, we again rely on k-means clustering and select c clusters per class. However, in this case, we only apply the clustering procedure once for each class instead of repeatably. We stick to the preprocessing pipeline presented above.

We experimentally evaluate this compression-based kNN methodology in the CL experimental setup for different choices of c. We compare our method to the two best-performing methods iCaRL [7] and GDumb [8] in the study [4] (results are taken from Tab. 7). Both rely on replay to avoid catastrophic forgetting. Our results are presented in Fig. 2. As one can see, we observe considerably better results than by applying SAM-kNN to the CL setup. We also outperform SAM-kNN and SAM-kNN (balanced) on smaller task sizes which confirms our hypothesis. Besides, our simple strategy performs on par or better compared to SOTA methods with a replay buffer of 5,000 images. Furthermore, due to the fact

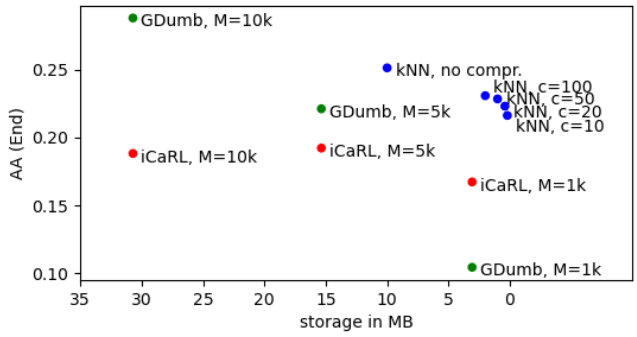


Fig. 2: Results of the experiments of the compression-based kNN with different compression levels. y-axis AA_{End}, x-axis (inverted, such that upper right corner is optimal) shows the required storage space for the kNN memory/the replay memory.

ESANN 2025 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 23-25 April 2025, i6doc.com publ., ISBN 9782875870933. Available from http://www.i6doc.com/en/.

that our methodology relies on a compressed representation of low-dimensional data, our method is more storage-efficient than the SOTA replay-based methods which need to store actual images in their memory buffers.

6 Conclusion

In this work, we found that one cannot directly use SML strategies like the LTM in SAM-kNN to prevent catastrophic forgetting in CL. In our experiments, we observed that the large task sizes considered in CL pose issues to SAM-kNN and that repeatedly applying compression is a problem. Thus, we proposed a compression-based kNN strategy that is more storage efficient than current replay-based CL strategies while performing comparable to methods with limited replay.

This work can be understood as a proof-of-concept showing that relying on suitable embeddings and compression is an efficient strategy to approach CL. Future work exploring more advanced compression and classification techniques is required. Besides, rethinking whether end-to-end deep learning is required in every CL task might be beneficial: Future work might explore whether using a similar embedding strategy paired with a shallow neural network architecture as a classification head that still benefits from replay techniques is advantageous over classical end-to-end CL.

References

[1] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):44:1–44:37, March 2014.

[2] Viktor Losing, Barbara Hammer, and Heiko Wersing. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing*, 275:1261–1274, January 2018.

[3] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A Comprehensive Survey of Continual Learning: Theory, Method and Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, August 2024.

[4] 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, January 2022.

[5] Viktor Losing, Barbara Hammer, and Heiko Wersing. KNN Classifier with Self Adjusting Memory for Heterogeneous Concept Drift. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 291–300, Barcelona, Spain, December 2016. IEEE.

[6] Valerie Vaquet, Fabian Hinder, Jonas Vaquet, Johannes Brinkrolf, and Barbara Hammer. Online learning on non-stationary data streams for image recognition using deep embeddings. In *IEEE symposium series on computational intelligence, SSCI 2021, orlando, FL, USA, december 5-7, 2021*, pages 1–7. IEEE, 2021.

[7] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental Classifier and Representation Learning. pages 2001–2010, 2017.

[8] Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. GDumb: A Simple Approach that Questions Our Progress in Continual Learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, volume 12347, pages 524–540. Springer International Publishing, Cham, 2020.