Online Continual Learning Without the Storage Constraint

Ameya Prabhu *ameya@robots.ox.ac.uk*
University of Oxford

Zhipeng Cai *zhipeng.cai@intel.com*
Intel Labs

Puneet Dokania *puneet@robots.ox.ac.uk*
University of Oxford

Philip Torr *phst@robots.ox.ac.uk*
University of Oxford

Vladlen Koltun *vkoltun@apple.com*
Apple

Ozan Sener *ozansener@apple.com*
Apple

Abstract

Online continual learning (OCL) research has primarily focused on mitigating catastrophic forgetting with fixed and limited storage allocation throughout the agent's lifetime. However, the growing affordability of data storage and the availability of universal pre-trained feature extractors, particularly in natural images and language domains, highlight a broad range of applications that do not adhere to these assumptions. In these cases, the primary concern lies in managing computational budgets rather than storage. In this paper, we target such settings, investigating the online continual learning problem by relaxing storage constraints and emphasizing fixed, limited economical budget.

Recent literature has shown that it is still challenging to prevent catastrophic forgetting or achieving rapid adaptation to incoming data despite accessibility of all past data. We provide a simple algorithm that can compactly store and utilize the entirety of the incoming data stream under tiny computational budgets using a kNN classifier and pre-trained representations. Our algorithm provides a consistency property attractive to continual learning: It will never forget past seen training data. We set a new state of the art on two large-scale OCL datasets: Continual LOCalization (CLOC) that has 39M images over 712 classes and Continual Google Landmarks V2 (CGLM) that has 580K images over 10,788 classes, beating methods under far higher computational budgets than our proposed method in terms of both reducing catastrophic forgetting of past data and quickly adapting to rapidly changing incoming data streams.

1 Introduction

In online continual learning, a learner processes a continuous stream of data originating from a non-stationary distribution. The learner is required to solve a number of problems: it needs to successfully learn the main task (accuracy), adapt to changes in the distribution (rapid adaptation), and retain information from the past (backward transfer). A key motif in recent work on online continual learning is the search for algorithms that control the trade-off between these possibly competing objectives under resource constraints.

To establish the resource constraints for typical commercial settings, we assess what is required of continual learners. A continual learner must deliver accurate predictions, scale to large datasets encountered during its operational lifetime, and operate within the system's total cost budget (in dollars). The economics of data

Table 1: The cost of storing data has decreased rapidly, allowing the storage of a large dataset for a negligible cost compared to the cost of computation (>$3000).

	1987	1997	2007	2017
Storage Cost				
Unit price ($)	30K	2K	80	49
Unit capacity	180MB	9GB	250GB	2TB
$/MB	83.33	0.22	0.0003	0.00002
Cost of storing YFCC ($)	350M	920K	1250	83
Compute Cost				
Training ER on YFCC		>2000$		

storage have been studied since 1987 (Gray & Putzolu, 1987; Gray & Graefe, 1997; Graefe, 2009; Appuswamy et al., 2017). Table 1 summarizes the trends, show a rapid decline in storage costs over time ($\sim$ $100 to store CLOC, the largest dataset for OCL (Cai et al., 2021), in 2017). In contrast, running ER (Cai et al., 2021), the state-of-the-art OCL method on the subset of YFCC-100M currently costs over $2000 on a GCP server. Consequently, computational costs are the primary budgetary concern, with storage costs being relatively insignificant. Therefore, as long as computational costs are controlled, economically storing the entire incoming data stream is feasible.

However, online continual learning has primarily been studied under limited storage constraints (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019a; Aljundi et al., 2019b), with learners only allowed to store a subset of incoming data. This constraint has led to many algorithms focusing on identifying a representative data subset (Aljundi et al., 2019b; Yoon et al., 2022; Chrysakis & Moens, 2020; Bang et al., 2021; Sun et al., 2022; Koh et al., 2022). Although limited storage is a practical constraint for biological learning agents and offline embodied artificial agents, deep learning-based systems are predominantly compute-constrained with a high throughput requirement. They must process incoming data points per second faster than the incoming stream speed to successfully keep up with the data stream. Cai et al. (2021) shows that even with unlimited storage, the online continual learning problem is hard as the constraint of limited computation implicitly restricts the effective samples used for training.

In this paper, we argue that storing the entirety of the data stream while meeting these requirements is possible. We propose a system based on approximate k-nearest neighbor (k-NN) algorithms, which are well-known for their scalability and inherent incremental nature (using only insert and lookup operations). The computational cost of this system has a graceful logarithmic scaling with data size even though it stores the entirety of past data. The further rationales for kNN are threefold. i) With the right representation, the nearest neighbour rule is an effective predictor at scale. ii) It has the consistency property, i.e., no forgetting on seen data. In other words, if a data point from history is queried again, the query yields the same label. iii) All past data is compactly stored in low-dimensional feature representations.

A critical aspect of the aforementioned system is obtaining an effective feature representation. While feature learning is the standard approach, it is not viable for our system due to the need to recompute the stored data's representation, resulting in a quadratic computational cost. Instead, we simply propose to use existing pretrained features which stem from extensive and diverse datasets, yielding robust representations. Remarkably, we find that even pretrained representations trained on rather small-scale ImageNet1K can provide effective features on datasets like Continual YFCC-100M (CLOC) which are comparatively more complex and far larger in size. Additionally, our approach overcomes a significant limitation of existing gradient-descent-based methods: the ability to learn from a single example. While using one gradient per example (i.e., batch size 1) is computationally infeasible for deep networks on large-scale datasets, our method efficiently stores a single feature extracted from the pretrained model in memory. The kNN mechanism immediately utilizes this data point, enabling rapid adaptation. We argue that the capacity to adapt to a single example is essential for truly online operation, allowing our simple method to outperform existing continual learning baselines.

Problem formulation. We formally define the online continual learning (OCL) problem following Cai et al. (2021). We further discuss the metrics we use to evaluate forward transfer (adaptability) and backward transfer (information retention). In classification settings, we aim to continually learn a function $f: \mathcal{X} \rightarrow \mathcal{Y}$, parameterized by θ_t at time t. OCL is an iterative process where each step consists of a learner receiving information and updating its model. Specifically, at each step t of the interaction,

1. *One* data point $x_t \sim \pi_t$ sampled from a non-stationary distribution π_t is revealed.
2. The learner makes the scalar prediction $\hat{y}_t = f(x_t; \theta_t)$ using a compute budget, B_t^{pred}.
3. Learner receives the true label y_t.
4. Learner updates the model θ_{t+1} using a compute budget, B_t^{learn}

A critical aspect of OCL is the budget in the second and forth steps, which limits the computation that the learner can expend. A common choice in past work is to impose a fixed limit on storage and computation per operation (Cai et al., 2021). We remove the storage constraint and argue that storing the entirety of the data is cost-effective as long as impact on computation is controlled. We relax the fixed computation constraint to a logarithmic constraint. In other words, we require that the computation time per operation fit within $B_t^{pred}, B_t^{learn} \sim \mathcal{O}(\log t)$. This construction results in total cost scaling $\mathcal{O}(n \log n)$ with the amount of data [1]

2 Related Work

Formulations. Parisi et al. (2019) and De Lange et al. (2020) have argued for improving the realism of online continual learning benchmarks. Earliest formulations Lopez-Paz & Ranzato (2017) worked in a task-incremental setup, assuming access to which subset of classes a test sample is from. Subsequent mainstream formulation Aljundi et al. (2019b;a) required models to predict across all seen classes at test time, with progress in the train-time sample ordering Bang et al. (2021); Koh et al. (2022). However, Prabhu et al. (2020) highlighted the limitations of current formulations by achieving good performance despite not using any unstored training data. Latest works Hu et al. (2022); Cai et al. (2021); Lin et al. (2021) overcome this limitation by testing the capability for rapid adaptation to next incoming sample and eliminate data-ordering requirements by simply using timestamps of real-world data streams. Our work builds on the latest generation of formulation Cai et al. (2021). Unlike Cai et al. (2021), we perform one-sample learning; in other words, we entirely remove the concept of task by processing the incoming stream one sample at a time, in a truly online manner. Some parallel works stress on computation as the key limited resource Harun et al. (2023). However, we further remove the storage constraint which is the key to eliminating degenerate solutions like GDumb Prabhu et al. (2020).

Methods. Traditional methods of adapting to concept drift (Gama et al., 2014) include a variety of approaches based on SVMs (Laskov et al., 2006; Zheng et al., 2013), random forests (Gomes et al., 2017; Ristin et al., 2015; Mourtada et al., 2019), and other models (Oza & Russell, 2001; Mensink et al., 2013). They are the most similar to our proposed method and offer natural incremental additional and querying properties, but have not been leveraged in the deep learning-based continual learning literature Ostapenko et al. (2022). Note that this direction is explored in works like Streaming LDA (Hayes & Kanan, 2020) and ExStream (Hayes et al., 2019). However, these approaches perform worse than partial training of a deep network (Hayes et al., 2020; Ostapenko et al., 2022). In contrast, we outperform fully finetuned deep networks with unrestricted access to past samples and far larger computational budgets.

The (online) continual learning methods designed for deep networks are typically based on experience replay (Chaudhry et al., 2019b) and change a subset of the three aspects summarized in Table 2: (i) the loss function used for learning, (ii) the algorithm to sample points into the replay buffer, and (iii) the algorithm to sample a batch from the replay buffer. Methods to sample points into the replay buffer include GSS (Aljundi et al., 2019b), RingBuffer (Chaudhry et al., 2019b), class-balanced reservoir (Chrysakis & Moens, 2020), greedy balancing (Prabhu et al., 2020), rainbow memory (Bang et al., 2021), herding (Rebuffi et al., 2017), coreset selection (Yoon et al., 2022), information-theoretic reservoir (Sun et al., 2022), and

[1] Although we believe $\mathcal{O}(n \log n)$ complexity is not prohibitive for practical applications, a further reduction (i.e $\mathcal{O}(n \log \log n)$) can be obtained by carefully introducing additional levels of hierarchy for astronomically large n.

Table 2: Recent online continual learning approaches, with key contributions in red. Most methods focus on better techniques for sampling into storage, while in our framework there is no storage constraint.

Works	MemSamp	BatchSamp	Loss	Other Cont.
ER (Base)	Random	Random	CEnt	-
GSS (Aljundi et al., 2019b)	GSS	Random	CEnt	-
MIR (Aljundi et al., 2019a)	Reservoir	MIR	CEnt	-
ER-Ring (Chaudhry et al., 2019b)	RingBuf	Random	CEnt	-
GDumb (Prabhu et al., 2020)	GreedyBal	Random	CEnt	MR
HAL (Chaudhry et al., 2021)	RingBuf	Random	CEnt	HAL
CBRS (Chrysakis & Moens, 2020)	CBRS	Weighting	CEnt	-
CLIB (Koh et al., 2022)	ImpSamp	Random	CEnt	MR, AdO
CoPE (De Lange & Tuytelaars, 2021)	CBRS	Random	PPPLoss	-
CLOC (Cai et al., 2021)	FIFO	Random	CEnt	AdO
InfoRS (Sun et al., 2022)	InfoRS	Random	CEnt	-
OCS (Yoon et al., 2022)	OCS	Random	CEnt	-
AML (Caccia et al., 2022)	Reservoir	PosNeg	AML/ACE	-

samplewise importance (Koh et al., 2022). These approaches do not apply to our setting because we simply remove the storage constraint. Approaches to sampling batches from the replay buffer include MIR (Aljundi et al., 2019a), ASER (Shim et al., 2021), and AML (Caccia et al., 2022). These require mining hard negatives or performing additional updates for importance sampling over the stored data, which are simply unscalable to large-scale storage as in our work. In our experiments, we compare with several of these approaches including ER as proposed in Cai et al. (2021) scaled appropriately to our setting. Additionally, our kNN based method offers attractive properties for OCL like never forgetting previously seen samples, not possible in most previous parametric approaches including the deep OCL methods presented above.

Pretrained Representations. Pretrained representations Yuan et al. (2021); Caron et al. (2021); Chen et al. (2021); Ali et al. (2021) have been utilized as initializations for continual learning, but in settings with harsh constraints on memory Wu et al. (2022); Ostapenko et al. (2022). Inspired by Ostapenko et al. (2022), we additionally explore effects of different pretrained representations along with comparison among traditional classifiers like logistic regression and online SVMs and discuss interesting findings.

Another emerging direction for using pretrained models in continual learning has been prompt-tuning as it produces accurate classifiers while being computationally efficient Wang et al. (2022b;a); Chen et al. (2023). However, Janson et al. (2022) show that simple traditional classification models outperform these complex prompt tuning strategies by significant margins.

Lastly, the direction most similar to ours is methods which use kNN classifiers alongside deep networks for classification Nakata et al. (2022); Iscen et al. (2022). We find this direction to be very promising. We note that we compare in a very different setting with no storage constraints, use far weaker pretrained classifiers trained on ImageNet1K when testing on large-scale datasets (upto 39M samples) where high performance with kNN classifiers is surprising in contrast to training and show that approximate-kNNs can achieve a high accuracy performance tradeoff at-scale not done previously. Additionally, Nakata et al. (2022) imposes memory restrictions on compared methods but uses all past data to achieve high performance using kNN.

3 Approach

We utilize pre-trained features as representations and k-nearest neighbor rule as a learning algorithm. Hence, our algorithm is rather simple. The key to operationalizing our algorithm is utilizing an efficient memory structure which satisfy cost constraints. We present our algorithm which we call Adaptive Continual Memory (ACM) below, with the kNN index being referred to as Memory.

At each time step, our learner performs the following steps:

1. *One* data point $x_t \sim \pi_t$ sampled from a non-stationary distribution π_t is revealed.
2. Learner extracts features $z_t = f(x_t; \theta_t)$
3. Learner retrieves nearest neighbors $\mathcal{N}_t = \texttt{Memory.Retrieve}(z_t, k)$.
4. Learner makes the prediction $\hat{y}_t = \texttt{majority-vote}(\mathcal{N}_t)$

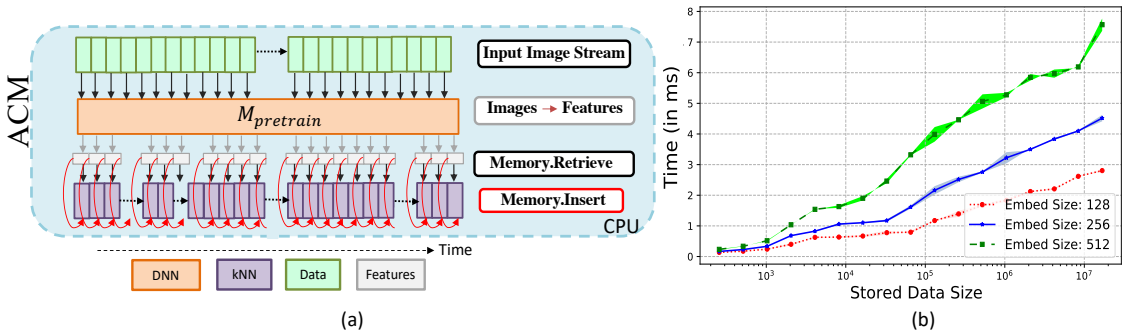

Figure 1: (a) Adaptive Continual Memory (ACM) performs `Memory.Retrieve` and `Memory.Insert` operations on features of new incoming samples, extracted by a static, pretrained deep network on a CPU server. (b) Wall clock time overhead of ACM after forward pass through deep network (x-axis is log-scaled). The longest observed overhead time is 8ms on 40 million samples in memory.

5. Learner receives the true label y_t.

6. Learner inserts new data: `Memory.Insert`(z_t, y_t).

We summarize this approach in Figure 1. Before explaining further implementation details, we discuss the number of advantages of this method.

Fast Adaptation. Suppose the learner makes a mistake in a given time step. If the same data point is received in the next time step, the learner will produce the correct answer. By leveraging nearest neighbors, we enable the system to incorporate new data immediately and locally modify its answers in response to as little as a single datapoint. Such fast adaptation, a core desideratum in online continual learning, is difficult with pure gradient descent and is not presently a characteristic of deep continual learning systems.

Consistency. Consider a hypothetical scenario in which a data point is queried at multiple time instants. Our learner will never forget the correct label for this data-point and will consistently produce it when queried, even after long time spans. While learning and memory are much more general than rote memorization, producing the correct answer on previously seen data is an informative sanity check. For comparison, existing continual learning systems forget a large fraction of previously seen datapoints even with a minimal delay (Toneva et al., 2019).

3.1 Efficient Memory Implementation

In the presented algorithm above for our method, feature extraction (step 2) and prediction (step 4) have a fixed overhead cost. However, nearest-neighbour retrieval (step 3) and inserting new data (step 6) can have high computational costs if done naively. However, literature in approximate k-nearest neighbours (Shakhnarovich et al., 2006) has demonstrated a high performance while scaling down computational complexity from linear $\mathcal{O}(n)$ to logarithmic $\mathcal{O}(\log n)$, where n is the number of data points in memory. We leverage approximate-kNN to allow our proposed approach to operate on the entirety of the data received so far under the constraint of logarithmic computational complexity, with approximate guarantees on preserving the above two discussed properties. We use approximate k-nearest neighbors based using HNSW algorithm to give high accuracy in minimal time based on the benchmarking results of Aumüller et al. (2020).

Furthermore, we present the wall-clock time of the overhead cost imposed by our ACM on datasets of 40 million samples to practically ground the logarithmic computational complexity in Figure 1. We observe that the computational cost of ACM scales logarithmically with the maximum time of 8ms. In comparison the time required for feature extraction of one sample for a ResNet50 model $\sim$ 10ms on an Intel 16-core CPU. Note that this implies the total inference cost of ACM classification is 18ms on 40 million samples using a Resnet50 feature extractor.

4 Experiments

Datasets: We benchmark using a subset of Google Landmarks V2 and YFCC-100M datasets. Both ordered by timestamps of upload date-time, with the task being online image classification, predicting the label of the incoming image.

i) Continual YFCC100M (CLOC): The subset of YFCC100M, which has date and time annotations Cai et al. (2021). We follow their dataset splits. We order the images by timestep and iterate over 39 million online timesteps, one image at a time, with evaluation on the next image in the stream. In contrast, CLOC uses a more restricted protocol assuming 256 images per timestep and evaluates on images uploaded by a different user in the next batch of samples.

ii) Continual Google Landmarks V2 (CGLM) : We use a subset of Google Landmarks V2 dataset Weyand et al. (2020) as our second benchmark. We use the train-clean subset, filtering it further based on the availability of upload timestamps on Flickr. We filter out the classes that have less than 25 samples. We uniformly in random sample 10% of data for testing and then use the first 20% of the remaining data as a hyperparameter tuning set, similar to CLOC. We get $430K$ images for continual learning with 10788 classes. We use the same hyperparameters as obtained on CLOC as Continual Learning algorithms should work with different data distributions.

Metrics: We follow Cai et al. (2021), using their average online accuracy until the current timestep k (a_k) as a metric for measuring rapid adaptation (forward transfer), given by $a_k = {}^1/_{N_t} \sum_{t=1}^{k} \mathbb{1}_{y_t=\hat{y}_t}$ where $\mathbb{1}_{(\cdot)}$ is the indicator function. Similarly, we measure information retention (preventing catastrophic forgetting) after finishing training by computing the average accuracy historically. Formally, information retention for a datapoint i timesteps earlier (IR_i) of the model in validation set, when we test using model at time T is similarly defined as $IR_i = {}^1/_i \sum_{s=T-i}^{T} \mathbb{1}_{y_t=\hat{y}_t}$

Approaches: We compare with a diverse range of methods as previously compared in Ghunaim et al. (2023), all of which are computationally capped to have the computational budget of one training pass over the CLOC dataset, termed as *fast stream* in the parallel work. We take their top two performing methods and compare their performance on the CGLM. We use the same hyperparameters as Ghunaim et al. (2023) for all methods unless specified otherwise, please refer to it for details about hyperparameters. All methods can use the full set of stored samples unless specified otherwise, however we noticed removing the memory restriction from 40K samples did not significantly change performance in the methods listed below as previously shown in (Cai et al., 2021), indicating OCL with limited computation is hard even without storage constraints. We describe each method below:

i) ER : ER performs online continual learning using an learning rate of 0.0005, training after every 64 samples with a training batch of size 128, consisting of the 64 incoming along with 64 sampled uniformly from all stored data. Each resultant model is used for predicting the classes of the incoming 64 samples. We observed reducing batch size contributes nearly all of the performance gain among interventions tested in Cai et al. (2021). The batch size is doubled for the CLOC dataset due to time constraints.
ii) PoLRS Cai et al. (2021): This adds PoLRS to the base ER model. PoLRS was the second most influencial component in improving performance among interventions tested in Cai et al. (2021).
iii) MIR Aljundi et al. (2019a): This adds MIR as the selection mechanism for choosing samples instead of uniform, to train on for training the base ER model. However, it is used in a task-free fashion as there are no task boundaries in tested datasets. We restrict the memory of this model to 40K for better sampling performance and results.
iv) GSS Aljundi et al. (2019b): This adds GSS as the sampling mechanism instead of lastK as used in the base ER model. We restrict the memory of this model to 40K due to computational constraints.
v) ACE Caccia et al. (2022): This replaces the loss function in the base ER model from Crossentropy to ACE loss for reducing the interference of classes not present in the current batch. It is done in a task-free fashion as there are no task boundaries in tested datasets.
vi) RWalk : This adds MIR as the selection mechanism for choosing samples to train on for training the ER model. However, this is done in a task-free fashion as there are no task boundaries in CGLM or CLOC.

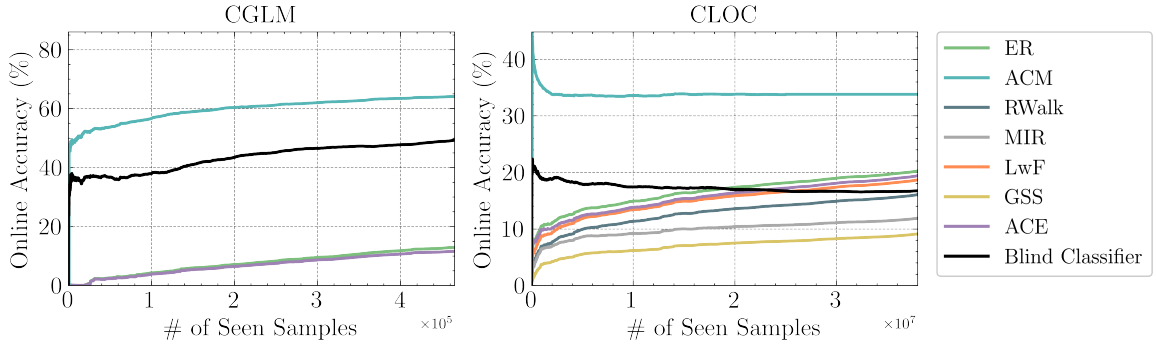

Figure 2: Performance on Online Adaptation on CGLM dataset (left) and CLOC (right): We observe that ACM outperforms existing methods by a large margin despite being far cheaper computationally.

Alongside these existing methods, we add two additional baselines which are far computationally cheaper as they involve no training.
vii) Blind Clf-k Cai et al. (2021): This is a baseline classifier with no access to the images, predicting the label of the current datapoint as mode of the recent k datapoints with a memory requirement of k (k=1 for CGLM and k=25 for CLOC).
viii) ACM (Ours): ACM uses DINO features pre-trained on ImageNet1K with similar performance as ResNet50-V2 used in the above methods. We train a mapping to the 256-dimensional embedding using the pretrain set and use it to extract features. We choose HNSW-kNN based on the benchmarking results of Aumüller et al. (2020) and guarantees offered by the structure but the method itself is agnostic to the specific approximate-kNN. We use the implementation in NMSlib (Malkov & Yashunin, 2018), with default hyperparameters.

4.1 Main Result: Evaluating ACM

Online Adaptation: We compare the average online accuracy over time of ACM to current state-of-the-art approaches on CGLM and CLOC in Figure 2. We observe that ACM significantly outperformed past approaches, by enabling efficient learning with a memory based approach. Moreover, the pre-trained features are universal enough to enable high performance. The final system has nearly no training compute cost compared to other methods resulting not only better accuracy but also significant cost effectiveness.

Information Retention: We compare the cumulative average performance of ACM to current state-of-the-art approaches on CGLM and CLOC in Figure 3. We observe that ACM outperforms existing methods on both datasets. Interestingly, ACM shows an nearly flat cumulative accuracy even over CLOC dataset with 39 million samples illustrating the benefits of utilizing past samples instead of encoding it in DNN parameters in backward transfer into history.

Take-away Messages: We illustrate the overwhelming benefit of storing information across time with a good initialization instead of trying to encode information of current tasks, making ACM a significantly more cost efficient yet accurate method in terms of both online adaptation and information retention. Pre-trained representations worked surprisingly well for challenging data streams scaling upto 39x the size of ImageNet1K, and from YFCC100M which is shown to be far more challenging than ImageNet1K, while simultaneously tackling the problem of adaptation to the distribution shifts over time.

Peformance of kNN across k. Lastly, we use $k = 2$ as default but present an ablation of kNN performance using different k values. We see that the kNN classifier is sensitive to the k value used but it is the only hyperparameter than needs to be optimized.=

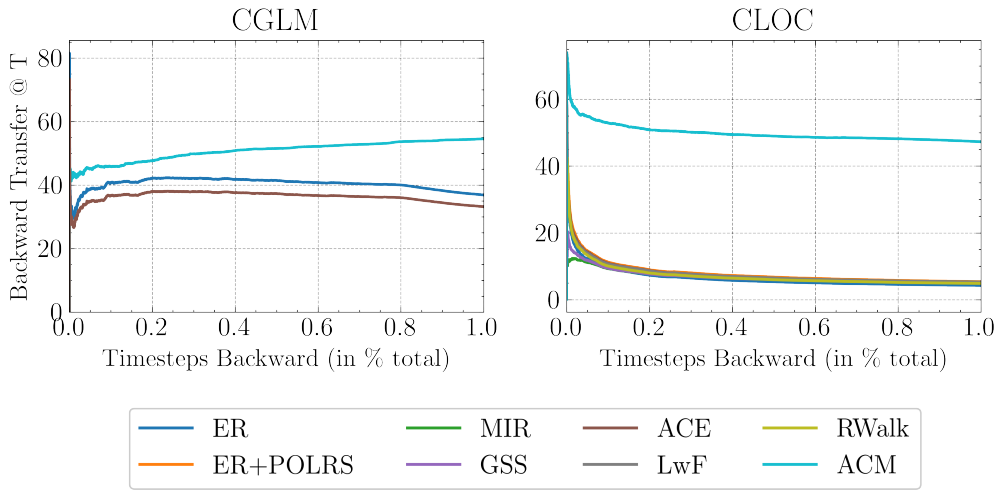

Figure 3: Performance on Information Retention on CGLM dataset (left) and CLOC (right): We observe that ACM outperforms existing methods by a large margin despite being far cheaper computationally.

5 Discussion

We would like to start our discussion by discussing the limitations of our approach. It is important to state that our method is not applicable to many interesting application scenarios like instances where pretrained features are not available or applications beyond cloud-based systems, such as end-devices and embodied agents. Although we believe this is a strong limitation, we do not think it invalidates the importance and impact of the settings our method tackles. Our setting, along with its constraints, are applicable in a broad set of deployment conditions, including but not limited to natural language and images, are so extensive, that we believe studying the problem of online continual learning under such a setting is impactful and important.

However, we do not believe memory should be restricted due to privacy constraints as often motivated in past literature . Recent progress in privacy-preserving machine learning and machine unlearning literature suggests that simply preventing access to old data is simply inadequate from fulfilling any reasonable privacy requirements, as the information needs to be erased from the model as well, antithetical to the goal of continual learning.

We believe it is also important to ground our system and its theoretical cost into a practical setting to guide practitioners and justify applicability. Consider a real-time data stream over an entire year, the storage size with our 512-dimensional representation would be roughly 1.5 TB and would cost $30 on a cloud. Moreover, the total computation time would enable real-time operation without any additional optimization for 71 years, extrapolating from Figure 1.

6 Conclusion

In this work, we provided a new set of constraints for online continual learning, with no restrictions on memory along with per-sample adaptation at every timestep. Our reformulation follows from the first principle analysis of modern computing systems' economic and computational requirements. We proposed an adaptive continual memory that stores the entirety of the data and still keeps computational efficiency. When evaluated on large-scale OCL benchmarks, our proposed system shows significant improvements over existing methods, illustrating a computationally cheap system which scales gracefully to large-scale datasets. Our findings showed that large-scale OCL benchmarks may benefit from methods which provide a knowledge bank to complement deep networks.

References

Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *NeurIPS*, 2021.

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. In *NeurIPS*, 2019a.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *NeurIPS*, 2019b.

Raja Appuswamy, Renata Borovica-Gajic, Goetz Graefe, and Anastasia Ailamaki. The five-minute rule thirty years later and its impact on the storage hierarchy. In *ADMS@VLDB*, 2017.

Martin Aumüller, Erik Bernhardsson, and Alexander John Faithfull. Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Inf. Syst.*, 87, 2020.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 2021.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. *ICLR*, 2022.

Zhipeng Cai, Ozan Sener, and Vladlen Koltun. Online continual learning with natural distribution shifts: An empirical study with visual data. In *ICCV*, 2021.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2019a.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. In *ICML-W*, 2019b.

Arslan Chaudhry, Albert Gordo, David Lopez-Paz, Puneet K. Dokania, and Philip Torr. Using hindsight to anchor past knowledge in continual learning, 2021.

Haoran Chen, Zuxuan Wu, Xintong Han, Menglin Jia, and Yu-Gang Jiang. Promptfusion: Decoupling stability and plasticity for continual learning. *arXiv preprint arXiv:2303.07223*, 2023.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021.

Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *ICML*, 2020.

Matthias De Lange and Tinne Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In *ICCV*, 2021.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. In *TPAMI*, 2020.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys*, 2014.

Yasir Ghunaim, Adel Bibi, Kumail Alhamoud, Motasem Alfarra, Hasan Abed Al Kader Hammoud, Ameya Prabhu, Philip HS Torr, and Bernard Ghanem. Real-time evaluation in online continual learning: A new paradigm. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Heitor M Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfharinger, Geoff Holmes, and Talel Abdessalem. Adaptive random forests for evolving data stream classification. *Machine Learning*, 2017.

Goetz Graefe. The five-minute rule 20 years later (and how flash memory changes the rules). *Communications ACM*, 2009.

Jim Gray and Goetz Graefe. The five-minute rule ten years later, and other computer storage rules of thumb. *ACM SIGMOD*, 1997.

Jim Gray and Franco Putzolu. The 5 minute rule for trading memory for disc accesses and the 10 byte rule for trading memory for cpu time. In *ACM SIGMOD*, 1987.

Md Yousuf Harun, Jhair Gallardo, Tyler L Hayes, and Christopher Kanan. How efficient are today's continual learning algorithms? *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop (CVPR-W)*, 2023.

Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *CVPR-W*, 2020.

Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *ICRA*, 2019.

Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *ECCV*, 2020.

Hexiang Hu, Ozan Sener, Fei Sha, and Vladlen Koltun. Drinking from a firehose: Continual learning with web-scale natural language. *IEEE Transactions on Pattern Analysis and Machine Intelligence, to appear*, 2022.

Ahmet Iscen, Thomas Bird, Mathilde Caron, Alireza Fathi, and Cordelia Schmid. A memory transformer network for incremental learning. *arXiv preprint arXiv:2210.04485*, 2022.

Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. *arXiv preprint arXiv:2210.04428*, 2022.

Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on class incremental blurry task configuration with anytime inference. *ICLR*, 2022.

Pavel Laskov, Christian Gehl, Stefan Krüger, Klaus-Robert Müller, Kristin P Bennett, and Emilio Parrado-Hernández. Incremental support vector learning: Analysis, implementation and applications. *JMLR*, 2006.

Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *NeurIPS*, 2021.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, 2017.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *TPAMI*, 2018.

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *TPAMI*, 2013.

Jaouad Mourtada, Stéphane Gaïffas, and Erwan Scornet. Amf: Aggregated mondrian forests for online learning. *Journal of the Royal Statistical Society*, 2019.

Kengo Nakata, Youyang Ng, Daisuke Miyashita, Asuka Maki, Yu-Chieh Lin, and Jun Deguchi. Revisiting a knn-based image classification system with high-capacity storage. In *European Conference on Computer Vision (ECCV)*, 2022.

Oleksiy Ostapenko, Timothee Lesort, Pau Rodríguez, Md Rifat Arefin, Arthur Douillard, Irina Rish, and Laurent Charlin. Continual learning with foundation models: An empirical study of latent replay. In *Conference on Lifelong Learning Agents (CoLLAs)*, 2022.

Nikunj C Oza and Stuart J Russell. Online bagging and boosting. In *International Workshop on Artificial Intelligence and Statistics*. PMLR, 2001.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *ECCV*, 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.

Marko Ristin, Matthieu Guillaumin, Juergen Gall, and Luc Van Gool. Incremental learning of random forests for large-scale image classification. *TPAMI*, 2015.

Gregory Shakhnarovich, Trevor Darrell, and Piotr Indyk. *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. The MIT Press, 2006.

Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *AAAI*, 2021.

Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. *ICLR*, 2022.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *ICLR*, 2019.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision (ECCV)*, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022b.

T. Weyand, A. Araujo, B. Cao, and J. Sim. Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval. In *CVPR*, 2020.

Tz-Ying Wu, Gurumurthy Swaminathan, Zhizhong Li, Avinash Ravichandran, Nuno Vasconcelos, Rahul Bhotika, and Stefano Soatto. Class-incremental learning with strong pre-trained models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. *ICLR*, 2022.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.

Jun Zheng, Furao Shen, Hongjun Fan, and Jinxi Zhao. An online incremental learning support vector machine for large-scale data. *Neural Computing and Applications*, 2013.