Online Stochastic Optimization for Data with Temporal Dependencies

Shivang Patel*, Ram Zaveri*, Samuel Chambers*, Zaigham Randhawa, and Gianfranco Doretto

West Virginia University, Morgantown, WV 26506, USA
{sap00008,rz0012,sdc00013,zar00002,gidoretto}@mix.wvu.edu

Abstract. Agents operating in the open-world sense data streams that are non-stationary and temporally correlated, from which they need to quickly learn new knowledge to adapt and be resilient to the changing environment they explore. Current learning-based vision approaches are not designed for agents operating in real-time in the open-world. They are based on stochastic optimizations performed offline with data samples independent and identically distributed according to a stationary probability distribution. Data streams processed by open-world agents have temporal dependencies, and are typically processed with a single-pass. In such conditions, vanilla SGD leads to slow convergence rates and biased estimations, and current approaches make simplifying assumptions, like having prior knowledge of the statistical properties of the stream. We overcome these limitations by introducing an online stochastic optimization approach that does not make any prior assumption and self-adapts to the current time dependency properties of the stream. We demonstrate the validity of the approach on synthetic data as well as on video datasets where we show very promising results for learning self-supervised representations and for classification downstream tasks.

Keywords: Stochastic Optimization · Time Dependent Data · Online Learning · Online Meta Learning · Self-Supervised Learning

1 Introduction

Let us consider an agent operating in the open-world. This agent needs to learn new knowledge as it becomes available through the stream of data it senses. It should do so as quickly as possible in order to adapt to the environment and be resilient against various distribution shifts [48]. Ideally, an agent should mimic the functionalities of biological agents as they learn very effectively for their entire existence based on processing the data streams of their sensory modalities. Such data streams are generally non-stationary, and temporally highly correlated. Notably, the work done in this setting is quite limited in the vision community [11, 58, 75], and the contribution of this work could benefit applications such as autonomous vehicles and robotics that work by processing data streams, like video, lidar, or other sensor data streams.

* Denotes equal contribution.

The aim of an open-world agent is to learn from and adapt to the incoming data stream, by updating its internal representations. Self-supervised learning (SSL) has proven effective in various vision applications [9, 10, 33, 59] as well as other machine learning tasks [15, 38, 60]. When performing downstream tasks, SSL representations either match or surpass fully supervised approaches [4, 12, 29, 30, 33]. On the other hand, the success of these methods cannot be directly transferred to open-world agents because they heavily rely on the ability to learn models based on minimizing an objective function, and this requires a *stochastic optimization* [53], which is a challenge, as we describe below.

The traditional workhorse used for stochastic optimization is stochastic gradient descent (SGD) [57]. SGD is a well studied tool, and it is known to perform optimally when it processes data samples that are *independent and identically distributed (i.i.d.)* according to a certain probability distribution [53]. Unfortunately, open-world agents are pressed to learn from data streams that are non-stationary and highly correlated, and need to do so quickly without reprocessing past data because they need to keep learning from future incoming data. SGD operating under this *non-i.i.d.* conditions where it processes *temporally dependent* data suffers from making updates based on biased gradients, which lead to poor convergence rates of the learning by a magnitude [18].

Current approaches that handle temporally dependent data make simplifying assumptions, including knowing in advance the nature of such dependency, usually by providing the *mixing time* conditions of the data stream [2, 42, 52, 52]. Intuitively, the mixing time refers to the time difference between two approximately independent samples. However, in practical applications this is not known by an agent in the open-world, as it may also vary. A recent approach, [18], does not make such assumptions; however, it makes a crucial simplification by assuming that the data stream is generated by a Markov chain.

In this work we introduce a novel stochastic optimization approach for processing temporally dependent data, which overcomes the limitations expressed above. It makes no requirements in terms of the statistics of the input data stream, so it requires neither prior knowledge of the mixing time of the stream nor for it to be a Markov chain. The proposed approach, which we name *adaptive stochastic optimization (ASO)*, makes no assumptions and is based on an online meta-optimization that optimally breaks the time dependence, which is self-adaptive to the current mixing conditions. In this way, the approach could adapt, for instance, to the changing dynamics of a video. The structure of the paper is as follows: in Section 3, we formulate the stochastic optimization problem with temporal dependent data and highlight the challenges. In Section 4, we introduce ASO, the meta-optimization procedure for addressing the challenges. In Section 5, we validate our framework on synthetically generated temporal data as well as real-world video datasets for SSL and recognition downstream tasks, where we provide very promising results.

2 Related Work

Online Learning. Online learning is a vital challenge in the vision community as it should enable a system to quickly adapt to the new data as it becomes available on the fly. Initially, it was driven by the increasing computational challenges posed by learning from vast and continuously expanding data sources [24, 36]. [25, 41, 64] further solidified online learning techniques against their inherent concept and distribution shifts [31]. However, none could truly tackle the issue of temporal dependence in continuous data streams. This becomes critical in various real-world applications such as for data streams coming from surveillance cameras, autonomous vehicles, etc. [7, 76] addressed these issues as aimed at classification tasks, where they show improvement over baseline methods. Additionally, [67] introduced a static method, namely, local drift degree, that tackles concept drifts as well as temporal dependence in regression tasks of the data streams. While many of these approaches explored various data mining techniques, very few have looked at it from an optimization point of view [54].

Online Optimization. Vanilla SGD assumes processing i.i.d. samples coming from a distribution, thus, inducing the conditional unbiasedness of the gradients during the stochastic optimization [17]. However, since temporally dependent streams lead to processing non-i.i.d. data, the gradients are conditionally dependent on (and biased upon) the updates up until the previous step. To ensure quicker convergence, this biasedness needs to be minimized. In this setting, [17] groups online optimizations in three categories: the first investigates adjusting the learning rate of the gradient updates with respect to the mixing time conditions of the temporally correlated data stream [20], the second one drops the temporally correlated data altogether [52], and the third one encodes the mixing conditions over time [2,42]. The first two categories require the knowledge of the mixing time conditions of the data stream, whereas the third one does not, instead, it utilizes a memory buffer to keep track of the temporally decorrelated samples, similar to the experience replay methods [46, 50].

One of the widely used online optimization strategies, AdaGrad [19], introduced the accumulation of covariance matrices of gradient updates to scale the learning rate at the current time step to account for various shifts in the mixing time conditions. This results in the learning curve slowing down as the gradient updates become smaller or vice-versa. MAG [18] makes further assumptions of the data following a Markov Chain [44], and introduces a gradient estimation method incorporating the AdaGrad [19] update rule. These approaches assume that the temporally dependent data is sampled from a generative ergodic/mixing stationary process that might not always be present in the real-world data; therefore, they are rather impractical in such scenarios. We approach this issue from a meta-learning perspective, where our algorithm adaptively detects the optimal mixing time conditions of the incoming data stream while not requiring any assumption on its distribution.

Meta-Learning. Meta-learning was originally defined as any method that would learn a "learning rule" [5,6,65]. Thus said, meta-learning approaches nowa-

days focus on learning in a few-shot manner, in which they see very limited data during the training phase and adapt to the new task [22,49,61,66,71]. Although these approaches show considerable performance gain, they still work under the assumption that the labeled data is available. [62] further suggests utilizing a semi-supervised learning framework to make use of both labeled and unlabeled data. Conversely, [35] employs unlabeled data to construct various meta-tasks to train the few-shot learners while [26] performs unlabeled clustering with a limited number of meta-parameters to induce meta-learning. More recently, works such as [14, 23] incorporate the meta-learning framework in an online setting where they perform adaptation under supervision on the current sequential tasks. Additionally, many meta-learning approaches [32,37,39,63] have focused on continual learning in an online sequential learning scenario by employing either memory buffers or regularization techniques. These conditions work under three major assumptions: supervision, storage for memory buffer, and availability of the pre-trained representations. In our setting, we focus on online learning where none of these conditions are met.

Self-Supervised Learning. Many approaches have focused on self-supervised learning (SSL) for representation learning and showcased superior performance in terms of compact representations learned. One advantage of such approaches is that one does not need to assume the downstream task as the information is unavailable. Initially, many SSL methods focused on reconstructing the data after corrupting it with noise or even artificially masking them [43, 56, 72, 74] to restore the missing information. Further, [16, 27] performed unsupervised training by either rotating the inputs or by shuffling the relative positions of the objects. However, these methods could not learn the fine-grained details of the objects. More recent approaches have shown significant performance gain in this regard [3], in which the deep metric learning methods [12,21] encourage similarity learning between semantically transformed versions of an input. Conversely, the self-distillation methods [10,13,30] map two augmented versions of the same image onto the other, whereas the canonical correlation analysis methods [4, 9, 73] balance three objectives of representation obtained from two different views: variance, invariance, and covariance. One key component of these algorithms is that they assume the data distribution to be stationary and do not work well if the data is from a time dependent non-stationary distribution. We show that with our adaptive optimization, these approaches can function well even with data that is temporally correlated.

3 Problem Formulation and Challenges

We assume that a data-generating process provides a time series $\{\xi_i\}$, where ξ_i, distributed according to π, might be a video frame at time i. We are interested in learning a model f from the data stream $\xi_1, \xi_2, \ldots$, which is parameterized by θ. Given a loss $\ell(\cdot)$, this is done by solving the following *stochastic optimization* problem

$$\min_{\theta \in \Theta} E_{\xi \sim \pi}[\ell(f(\theta), \xi)] \ , \tag{1}$$

where Θ is an optimization domain. Problem (1) is typically solved iteratively via stochastic gradient descent (SGD) according to the update rule

$$\theta^{t+1} = \Pi_{\Theta}(\theta^t - \alpha_t \nabla \mathcal{L}(\theta^t)) \ , \quad t = 1, \ldots, T \ , \tag{2}$$

where α_t is a learning rate, $\Pi_{\Theta}(\cdot)$ is a projection operator, and $\mathcal{L}(\theta)$ is an estimate of the expectation in (1). Such estimate is typically given by the empirical risk. However, it can be replaced by the loss of a single data point, $\ell(f(\theta), \xi)$, or, if consecutive instances from the data stream ξ_i, ξ_{i+1}, $\ldots$, ξ_{i+n-1}, are grouped into a batch B^t of size n, it can be the average of the losses on those instances. In this more general case the iterative update (2) processes a *data stream* $\mathcal{S}$ of batches $B^1, \ldots, B^t, \ldots$.

It is well known that SGD provides an effective solution to the stochastic optimization problem (1) if the update (2) processes data samples that are *independent and identically distributed (i.i.d.)* according to π. In this case, the parameter estimation θ^t and the gradient $\nabla \mathcal{L}(\theta^t)$ remain statistically independent, ensuring that the gradients are conditionally unbiased, and SGD reaches convergence with optimal rate [53, 57].

In this work, we aim to solve problem (1) while ingesting a stream $\{\xi_i\}$ of *time dependent* data. This means that the data stream is not i.i.d. Therefore, the gradient estimates, whether computed over a sample or a batch, are no longer unbiased in (2), and the convergence rate of SGD is suboptimal. A typical way to characterize the time dependency of a data stream is via the *mixing time* τ_m. Intuitively, τ_m represents the minimal time difference between two approximately independent samples. A more formal definition is reported in Section 5.1. Under certain mixing conditions the effects of gradient bias on convergence have been formally studied [1].

This motivates the need to look for a more efficient algorithm to solve (1) under the non-i.i.d. assumption. This is important in the practical scenario where an agent in the open-world would want to adapt to a changing environment with the highest speed, since being subjected to a distribution shift [51]. This would enable learning the best possible representation from instance $\xi_i \in B^t$, as soon as possible, and only once, since there is the need to learn from new streaming data in the future. Because of this *single-pass* setting, ξ_i will not be processed again, and we need to make the best possible use of it. This is in contrast with batch learning, where the same data is used in multiple passes or epochs.

4 Adaptive Stochastic Optimization for Dependent Data

A trivial and effective strategy to improve convergence is to simply run SGD with only one every κ samples. If the stream is made of datapoints sampled from a Markov chain, and κ is chosen as the mixing time of the process, SGD with this data drop strategy (SGD-DD) achieves the same convergence rates as vanilla SGD with i.i.d. data [8].

Another effective strategy to improve convergence is to run SGD with mini-batch sampling (SGD-MB), where $\xi_i, \xi_{i+1}, \ldots, \xi_{i+n-1}$ form batch B^t. Theoreti-

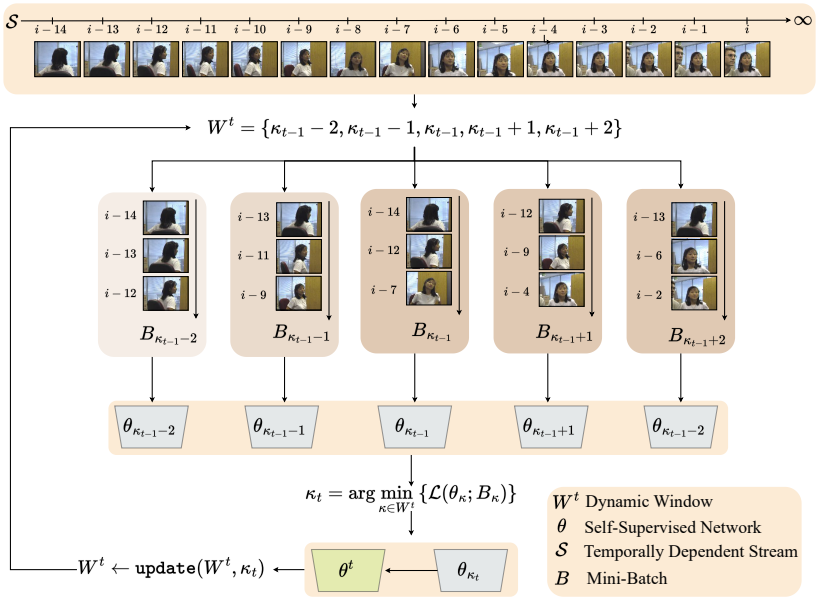

Fig. 1: Adaptive Stochastic Optimization. The dynamic window collects the data with different drop rates and feeds it to a pool of models where a drop rate with the least loss is considered to update the next dynamic window. Similarly, the model at current time-step is updated with the parameters obtained though the least loss. This cycle continues until the stream has concluded.

cal results under certain mixing conditions of the stream suggest that the convergence of SGD-MB is even faster than SGD-DD [47], and that larger batches are better, which is also intuitive. Similar results are also reported in [28], where they underline how a time varying size of the minibatches counteracts the effects of short and long range dependencies, including bias, variance, and convergence.

Inspired by that body of work, we propose a strategy that combines SGD-DD and SGD-MB, where minibatches act as if they had a time-varying size, in a way that is self-adaptive to the current dependency status of the stream. Unlike the only other adaptive method in the literature [17], ours does not assume data to be drawn from a Markov chain, it is generic and assumes no dependency restrictions. The key idea is to compose the minibatch B^t, by selecting one every κ samples, while leaving the size n fixed. Moreover, the minibatch is made time-varying, by adaptively selecting the drop rate κ, so that $B^t(\kappa) = [\xi_i, \xi_{i+\kappa}, \ldots, \xi_{i+(n-1)\kappa}]$. This is in contrast with prior approaches that assume prior knowledge of the drop rate, or mixing time, and therefore have limited practical applicability [28, 47].

The stochastic optimization will proceed by iteratively updating the parameters θ of the model f, while minimizing the empirical risk $\mathcal{L}(\theta)$, which depends on $\ell(\cdot)$. The optimal update would be based on the minibatch $B^t(\kappa)$, where the drop rate should be chosen to provide the highest decrease of the empirical risk.

Algorithm 1 Adaptive Stochastic Optimization for Streaming Data

procedure Adaptive Stochastic Optimization
  Let $\mathcal{S}$ be a data stream and $\mathcal{B}$ a FIFO circular buffer
  Initialize the dynamic window W^t and the model parameters θ^t
  Initialize the populations of batches $\{B_\kappa\}$ and models $\{\theta_\kappa\}$, $\kappa \in W^t$
  while $\mathcal{S}$ is available **do**
    $\mathcal{B} \leftarrow \texttt{fill}(\mathcal{S})$
    for all $\kappa \in W^t$ **do**
      $B_\kappa \leftarrow \phi_\kappa(\mathcal{B})$
      $\theta_\kappa \leftarrow \theta^t - \alpha \nabla_{\theta^t} \mathcal{L}(\theta^t; B_\kappa)$
    $\kappa_t = \arg\min_{\kappa \in W^t} \{\mathcal{L}(\theta_\kappa; B_\kappa)\}$
    $\theta^t \leftarrow \theta_{\kappa_t}$
    $W^t \leftarrow \texttt{update}(W^t, \kappa_t)$

This means that the update should be made according to the following rule

$$\kappa_t = \arg\min_{\kappa} \mathcal{L}(\theta^t - \alpha \nabla_\theta \mathcal{L}(\theta^t; B^t(\kappa)); B^t(\kappa)) \ , \quad \theta^{t+1} = \theta^t - \alpha \nabla_\theta \mathcal{L}(\theta^t; B^t(\kappa_t)) \ , \tag{3}$$

where $\mathcal{L}(\theta; B^t(\kappa))$ is the empirical risk associated with the minibatch $B^t(\kappa)$. In this way, the stochastic optimization over the stream of batches effectively becomes a meta-optimization [23] where the drop rate is optimally adapted over time to continually maximize the learning from the latest batch of data. Algorithm 1 summarizes the steps of the iteration.

We perform the meta-optimization by forming a population of models $\{\theta_\kappa\}$, where $\kappa \in W$, and W is a suitable set of drop rates. Then, at each update the risk is evaluated for every drop rate, and the one that provides the highest risk reduction is used to perform the model update. For efficiency reasons and keeping the population size low, the set W is in fact a dynamic window (i.e., W^t). It is initialized to $\{1, 2, \ldots, \kappa_{max}\}$, and at the end of iteration t, the drop rates for the next iteration are shifted so that κ_t is in the center. The function $\texttt{update}(W^t, \kappa_t)$ does that in Algorithm 1. This strategy allows to contain the size of the population of models, save computation, and allows for a large range of drop rates.

In Algorithm 1 the function $\texttt{fill}(\mathcal{S})$ refills the FIFO circular buffer $\mathcal{B}$ with the latest n samples from the stream $\mathcal{S}$. The size of $\mathcal{B}$ is $n\kappa_{\text{MAX}}$, where κ_{MAX} defines the largest drop rate that can be reached. Finally, the function $\phi_\kappa(\cdot)$ selects the n most recent samples, by choosing one every κ. Note however, that for $\kappa > 1$, $\phi_\kappa(\cdot)$ will chose $(\kappa-1)n/\kappa$ samples that were selected to form the previous batch. To better leverage the use of data, rather than a deterministic sampling, we make $\phi_\kappa(\cdot)$ implement a Bernoulli sampling, where each data sample is selected with probability $1/\kappa$, until n are chosen. Figure 1 describes an overview of the Algorithm 1 which we refer to as *Adaptive Stochastic Optimization (ASO)*.

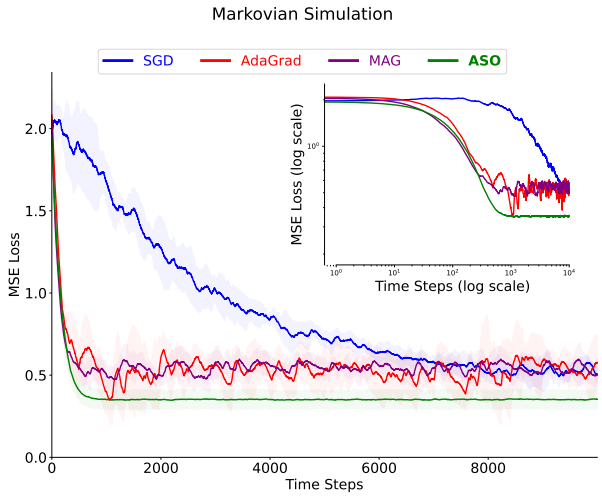

Fig. 2: Synthetic Markovian Temporal Data. Comparison of the MSE losses of a linear regression task during online learning from synthetic temporal data sampled from a two-state Markov chain as described in Section 5.1. MAG [18] shows quicker learning rate at the very beginning, but plateaus quickly after, whereas ASO shows quicker learning rate as well as convergence rate overall.

5 Experiments

In this section, we showcase the effectiveness of ASO for self-supervised learning with streaming data with an extensive experimental validation. First, we evaluate ASO with synthetic temporal data, where the stream has a non-i.i.d. distribution that is Markovian with imposed mixing conditions. Second, we evaluate ASO on real-world temporal data like video streams where the distribution is unknown and is generally non-Markovian. We perform these experiments on high-end workstations equipped with NVIDIA RTX A6000 GPUs.

5.1 Synthetic Temporally Dependent Data

In order to systematically evaluate the behavior of ASO with temporally dependent data, we follow an approach similar to [18]. We generate synthetic data that temporally evolves according to a Markov chain, which is a common way to characterize temporal data dependencies [45]. We assume that $s_t \in \Omega = \{1, 2\}$ is a 2-state Markov chain. The chain has a symmetric state transition matrix, defined by a probability of transition between states $p \in (0, 1)$. The state s_t selects the model weights $w_{s_t} \in \mathbb{R}^d$, randomly predefined. The weights are used to produce the output of a noisy regression model $y_t = X_t w_{s_t} + \epsilon_t$, where $X_i \in \mathbb{R}^{c \times d}$ is drawn from $\mathcal{N}(0, I)$, and ϵ_t is drawn from $\mathcal{N}(0, 10^{-3} I)$. In this way, we produce a Markovian data stream $y_1, y_2, \cdots$.

A typical way to characterize the time dependency of a data stream is via the mixing time τ_m. Intuitively, τ_m represents the minimal time difference between two approximately independent samples. More formally, for an ergodic time-homogeneous Markov chain s_t, over a finite state space Ω, the mixing time is defined with respect to the total variation distance of the chain from its stationary distribution μ. For two distributions P and Q over the same probability space $(\Omega, \mathcal{F})$, where $\mathcal{F}$ is the σ-algebra of the events, the total variation distance is defined as $D_{TV}(P,Q) \doteq \sup_{a\in\mathcal{F}} |P(a) - Q(a)|$. We then compute the distance between μ and the conditional distribution of s_{t+1} given s_1, $P(s_{t+1}|s_1)$, and when it goes below a threshold δ then $s_{t+1}|s_1$ is sufficiently indistinguishable from μ and therefore s_{t+1} is approximately independent from the initial state s_1. More precisely, the *mixing time* τ_m [45], is given by

$$\tau_m(\delta) \doteq \inf\{t : \sup_{s_1\in\Omega} D_{TV}(P(s_{t+1}|s_1), \mu) \leq \delta\} . \tag{4}$$

Given this definition of τ_m, it is also possible to characterize its order of growth [45]. For the case of the 2-state symmetric chain s_t that we have used, it is easy to show that such order of growth is completely defined by the probability of transition between states p. In particular, when p is small (i.e., $1 - 2p \approx 1$), then $\tau_m \doteq \tau_m(1/4) = \Theta(1/p)$.

With this framework we generate a synthetic Markovian temporal stream (X_1, y_1), (X_2, y_2), $\cdots$, from which we learn a regression model $y = Xw$, where w is optimized by minimizing the mean square error (MSE). We set $d = 100$ and $c = 250$. We also set $p = 10^{-4}$, which means that the mixing time is $\tau_m \approx 10^4$, and we generate a stream of $T = 10^6$ samples.

We compare ASO against three methods. We use SGD as the baseline, AdaGrad [19] as the online optimization baseline, and MAG [18], which is an approach that has been developed to work under the Markovian assumption with performance guarantees. Just like ASO, MAG does not require to know what the mixing time is. Under similar Markovian assumptions, there are other methods, like SGD-DD [8] and Ergodic Mirror Decent (EMD) [20], that have theoretical performance guarantees equivalent to MAG, but they are limited in that they do require prior knowledge of the mixing time to operate.

In our experiments, SGD and ASO use a learning rate of $1/\sqrt{T}$ [53], whereas AdaGrad and MAG use an adaptive learning rate according to their respective algorithms, which is $(\sum_{k=1}^{t} \|g_k\|^2)^{-1/2}$, where g_k is the gradient at the k-th update. SGD, AdaGrad, and MAG perform an update at the same time, which is driven by how MAG selects the size of the stream minibatch used to compute the gradient estimation. Such size is 2^J where J is drawn from a geometric distribution limited to the interval $[1, 5]$. Because of that, we set the size of the ASO minibatch to the mean size of the MAG minibatch, which is $n = 2^2 = 4$.

In Figure 2 we plot the mean and standard deviation of the MSE loss against time t, obtained from 5 runs. We can observe SGD performing with the lowest convergence rate, with AdaGrad significantly improving over SGD. AdaGrad also slightly underperforms MAG in terms of learning speed. Moreover, MAG manages to learn quicker at the very beginning, but it plateaus also quickly

Table 1: Varying Mixing Time Conditions. Comparative analysis on various mixing times. The numbers represent the average MSE and standard deviation (lower is better) at the last iteration. Bold suggests best, and underline suggests second best.

Method	Mixing Time ($\tau_m \approx 1/p$)			
	$p = 0.02$	$p = 0.002$	$p = 0.0002$	$p = 0.00002$
SGD	0.7569 ± 0.1268	0.7363 ± 0.1410	0.7733 ± 0.1428	0.7711 ± 0.2239
AdaGrad [19]	<u>0.4916 ± 0.1418</u>	<u>0.5038 ± 0.0874</u>	0.5309 ± 0.1733	0.4610 ± 0.3178
MAG [18]	0.6277 ± 0.1332	0.5879 ± 0.1045	<u>0.5546 ± 0.1459</u>	<u>0.5025 ± 0.1723</u>
ASO	**0.3054 ± 0.1101**	**0.3279 ± 0.0996**	**0.3752 ± 0.1099**	**0.2915 ± 0.1236**

immediately after. ASO on the other hand shows considerably quicker convergence. We should note that both MAG and AdaGrad exhibit noisier learning curves compared to ASO and SGD.

We also evaluate the behavior of SGD, AdaGrad, MAG, and ASO with respect to different mixing time conditions. Specifically, we generate four additional streams with $p = 2 \times 10^b$ and $b \in \{-2, -3, -4, -5\}$, and for each of them we report the average and standard deviation of the MSE loss over 5 runs at the end of the training. In Table 1, we show that MAG performs better under large mixing time conditions, whereas AdaGrad shows better stability when the mixing time conditions are smaller. SGD does not show significant changes in either of those cases. On the other hand, ASO shows robustness against all the mixing time conditions in terms of stability and convergence. Next, we will discuss experiments where the Markovian assumption might not hold.

5.2 Real-World Temporally Dependent Data

General setup. We consider the problem of self-supervised learning (SSL) performed online from a stream of real-world temporal data, such as video streams, which are very common in computer vision applications. We set out to learn SSL representations based on widely used techniques, such as SimCLR [12]. In this setting, a backbone model will learn from the stream in a single-pass only, and will not have the chance to reuse prior data, as usually done when training is performed over multiple epochs. Once the stream has concluded, the backbone is frozen and a linear probe [40] is attached to learn the downstream task. The backbone used in the following experiments is ResNet18 [34]. The experiments are based on SimCLR [12], and the downstream task is trained by attaching a projection head of size 512. For all the experiments and all methods, we use a batch size of $n = 256$ and a learning rate of 10^{-3} with momentum set to 0.9. We consider two metrics, one is the SSL loss throughout the single-pass training on the video stream, and the downstream task accuracy. We compare ASO with the approaches mentioned in Section 5.1. Note that MAG and AdaGrad also adapt the learning rate, while MAG selects batches of variable size 2^J, where we limit J to the interval $[1, 8]$ considering the memory limitations.

Datasets. We consider UCF101 [68] which is an action recognition dataset. It contains about 13,000 videos belonging to 101 classes. Since these videos are small in temporal scale, we concatenate them in a class-incremental fashion,

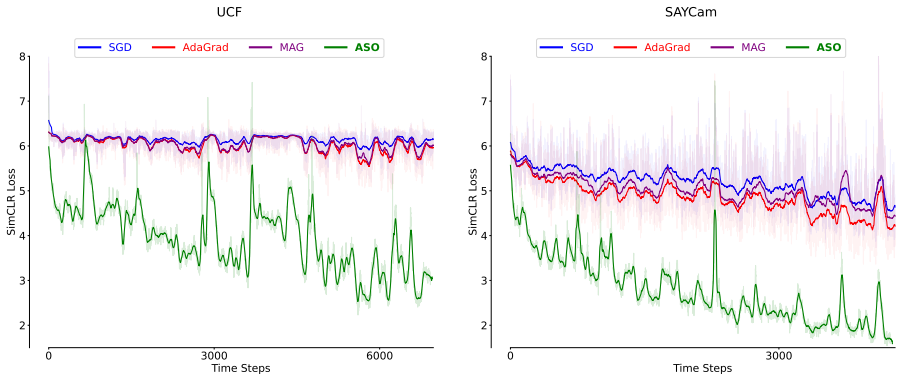

Fig. 3: Online Self-Supervised Learning. Comparison between the SimCLR losses of different approaches as a single-pass online training is performed on the USF101 [68] and SAYCam [70] datasets. ASO shows considerably quicker convergence rate compared to the other approaches.

resulting in one continuous video stream. We utilize the same train-test split described in [40]. Once the single-pass SSL training is concluded, we perform the action recognition downstream task. We follow the protocol in [40] and extract the average output representations from 8 consecutive frames of the same label before feeding it to the linear probe.

Even though UCF101 is a viable option for composing a video stream, it lacks natural distribution shifts among different videos. For that reason, we also use SAYCam [70] as our next large-scale video dataset. SAYCam contains videos collected from cameras head-mounted on three children, S, A, and Y, from 6 to 32 months, recorded for about 1 to 2 hours per week. This results in videos consisting of natural distribution shifts. Among these videos, S is the only video that is heavily annotated, and therefore, we only use S in this study. We choose image classification as our downstream task on this benchmark, where we utilize the class annotations proposed in [55].

Comparative Analysis on Online Self-Supervised Learning. We conduct online self-supervised learning on above-mentioned real-world datasets and compare against the following approaches: SGD (baseline), AdaGrad [19], and MAG [18]. Here, we should note that training is done in a single-pass, meaning that the models see an instance only once in these runs. For each experiment we perform 5 runs and show means and standard deviations of the SSL losses. As illustrated in Figure 3, ASO converges significantly faster compared to the others. We observe SGD progressing towards convergence relatively slowly compared to the others. Conversely, AdaGrad seems to outperform SGD and MAG although the difference between MAG and AdaGrad is minimal as they both share similar optimization strategies. While the others plateau very quickly, ASO continues to show a decreasing error rate throughout the stream showcasing its ability to

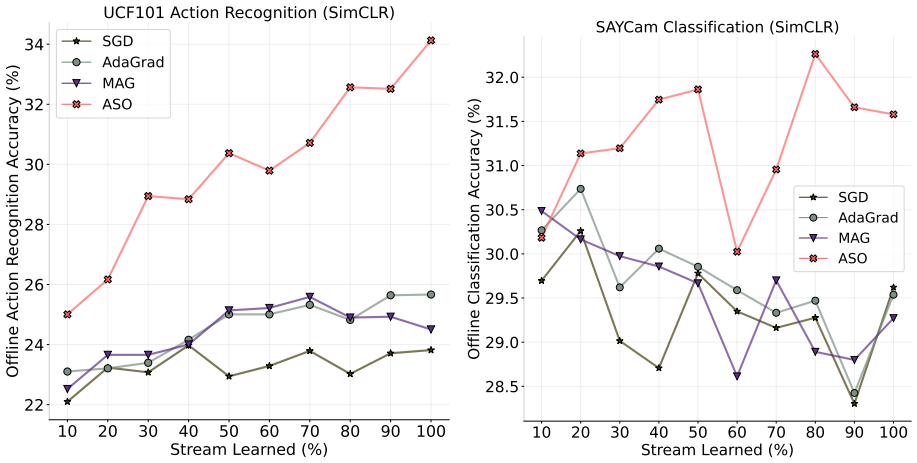

Fig. 4: Offline Downstream Accuracy. Offline downstream evaluation on UCF101 [68] and SAYCam [70] datasets at different time-steps during the single-pass online self-supervised training. Downstream task performance of ASO consistently shows improvement over the others.

Table 2: Downstream Evaluation under Barlow Twins. Downstream evaluation under Barlow Twins [73] online SSL training. The numbers on UCF101 [68] represent the downstream evaluation on action recognition, whereas SAYCam [70] numbers represent the downstream performance on a classification task. Bold suggests best, and underline suggests second best.

Datasets	Methods (BT training)			
	SGD	AdaGrad [19]	MAG [18]	ASO
UCF101 [68]	19.904	22.918	<u>23.447</u>	**28.099**
SAYCam [70]	26.333	29.332	<u>30.034</u>	**31.733**

keep learning as more data becomes available. In general, ASO shows its ability to learn from non-Markovian data as well, while others fall short in this regard.

A trend we notice is that UCF101 [68] experiments show generally slower convergence compared to the experiments on SAYCam. One explanation for such phenomena would be the varied distribution of UCF101 as it contains many different action sequences from various domains, making it difficult to converge in merely one pass. On the other hand, SAYCam [70] is a natural dataset that is reminiscent of the spirit of curriculum learning [69], in which children explore very limited items that might be revisited during the same stream, with more complex future explorations, making it somewhat easier to converge.

Comparative Analysis on Offline Down-Stream Performance. We perform a downstream offline evaluation on UCF101 and SAYCam to exploit the learned representations of ASO, and compare against the others. We evaluate each approach at different stages as they learn by progressively ingesting a continuous stream. Essentially, we perform downstream evaluation when the

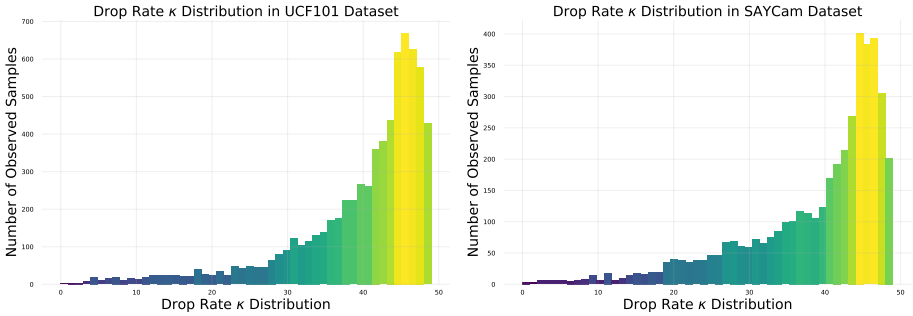

Fig. 5: Drop Rate Distribution. Drop rate distribution on UCF101 [68] and SAYCam [70] datasets. The distribution of the drop rate κ remains within the bounds of [0,50] for both datasets.

approach has learned from 10% of the stream, then 20%, and so on, up until it has seen the whole stream, and that is where we evaluate the final model. Each downstream evaluation is performed offline and evaluates the self-supervised representations learned by the model up to that point. Here, the backbone remains frozen and only the linear probe is learned using a standard SGD optimization strategy. We set the learning rate to 10^{-2}, momentum to 0.9, and batch size to 256. We report the best accuracy after training the linear probe for 800 iterations for UCF101, and 25 iterations for SAYCam.

Figure 4 describes the performances of ASO, SGD, AdaGrad, and MAG. On the experiments on UCF101, we notice that SGD does not improve performance as more temporally correlated samples become available. Although AdaGrad and MAG show increase in performance over time, ASO shows a significantly higher performance gain. Additionally, ASO appeared to continually get better at downstream tasks, whereas others plateau after observing 50-60% of the samples. We observe a similar trend in our experiments on SAYCam. We evaluate ASO also with Barlow Twins (BT) [73] to validate its applicability with other SSL approaches. Table 2 shows similar trends as observed in training with SimCLR [12] where ASO outperforms others in relvant downstream tasks. This confirms the superiority and the adaptability of ASO in terms of online self-supervised representation learning from video streams.

Ablation on Drop Rate. Since a core contribution of ASO is its adaptability towards unknown and variable mixing times, which is a direct consequence of detecting the optimal drop rates of the time dependent streams, we perform an extensive analysis on various fixed drop rates and compare them with, what we confirm to be, the upper-bound given by the ASO. First, we collect the best drop rates at each time-step of ASO training to determine the number of times those drop rates have been optimally used for training. Figure 5 shows the histograms representing the drop rate (κ) distributions on both UCF101 [68] and SAYCam [70] datasets. They show that for any given dataset κ_{MAX} has an upper-bound of 49 while 45 was the most used drop rate. Therefore, in Fig-

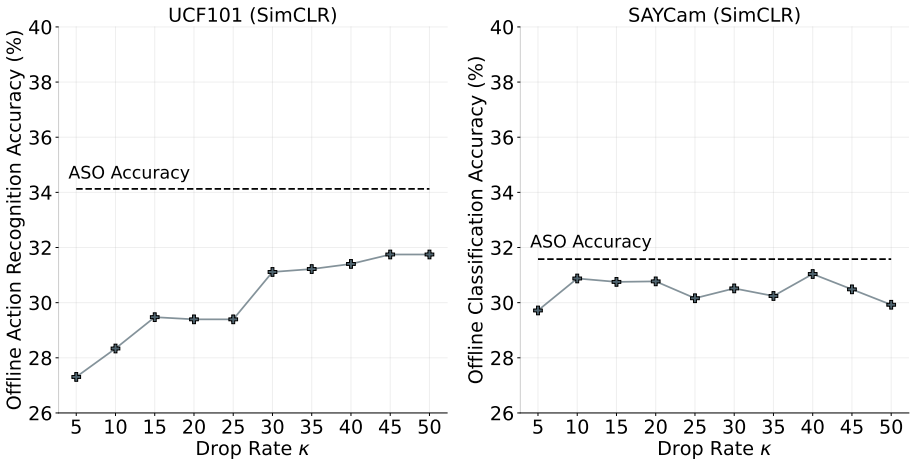

Fig. 6: Drop Rate Ablations. Drop rate ablations on UCF101 [68] and SAYCam [70] datasets. The larger drop rates, κ, show improved accuracies yet underperform ASO.

ure 6 we evaluate a range of fixed κ's in the interval $[5, 50]$. We notice that as the drop rate κ increases, the downstream accuracies also increase. Additionally, even though 45 appeared to be the best drop rate, the performance does not necessarily improve significantly at that fixed drop rate, as a big portion of the samples have many different optimal drop rates. This confirms the significance of a framework like ASO, which adapts to the current mixing conditions, which are unknown, and unforeseeable in real-world time dependent data streams.

6 Conclusions

We present Adaptive Stochastic Optimization (ASO), a framework for learning from temporally dependent data streams. Differently from current approaches, no assumptions are made on the prior knowledge and statistics of the stream. This allows to blindly apply the approach to any data stream and maximize the ability to learn from a single-pass. This is particularly useful for an agent operating in the open-world processing time dependent and non-stationary data. We demonstrated the use of ASO in controlled conditions with synthetic data, where we achieve significant convergence improvement compared with the baseline SGD algorithm, but also AdaGrad, and the adaptive MAG. Similarly, using real-world video data, we have shown that ASO significantly improves upon the compared approaches in two SSL tasks, and two downstream tasks. Although ASO enables fast adaptability to the current conditions, it does not prevent forgetting prior knowledge.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grants No. 2117575, and 2223793.

References

1. Agarwal, A., Duchi, J.C.: The generalization ability of online algorithms for dependent data. IEEE Transactions on Information Theory **59**(1), 573–587 (2012)
2. Agarwal, N., Chaudhuri, S., Jain, P., Nagaraj, D., Netrapalli, P.: Online target q-learning with reverse experience replay: Efficiently finding the optimal policy for linear mdps. arXiv preprint arXiv:2110.08440 (2021)
3. Balestriero, R., Ibrahim, M., Sobal, V., Morcos, A., Shekhar, S., Goldstein, T., Bordes, F., Bardes, A., Mialon, G., Tian, Y., et al.: A cookbook of self-supervised learning. arXiv preprint arXiv:2304.12210 (2023)
4. Bardes, A., Ponce, J., LeCun, Y.: Vicreg: Variance-invariance-covariance regularization for self-supervised learning. arXiv preprint arXiv:2105.04906 (2021)
5. Bengio, S., Bengio, Y., Cloutier, J., Gecsei, J.: On the optimization of a synaptic learning rule. In: Preprints Conf. Optimality in Artificial and Biological Neural Networks. vol. 2 (2013)
6. Bengio, Y., Bengio, S., Cloutier, J.: Learning a synaptic learning rule. Citeseer (1990)
7. Bifet, A., Read, J., Žliobaitė, I., Pfahringer, B., Holmes, G.: Pitfalls in benchmarking data stream classification and how to avoid them. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part I 13. pp. 465–479. Springer (2013)
8. Bresler, G., Jain, P., Nagaraj, D., Netrapalli, P., Wu, X.: Least squares regression with markovian data: fundamental limits and algorithms. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. pp. 16666–16676. No. Article 1398 in NIPS'20, Curran Associates Inc., Red Hook, NY, USA (Dec 2020)
9. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Advances in neural information processing systems **33**, 9912–9924 (2020)
10. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
11. Carreira, J., King, M., Pătrăucean, V., Gokay, D., Ionescu, C., Yang, Y., Zoran, D., Heyward, J., Doersch, C., Aytar, Y., Damen, D., Zisserman, A.: Learning from one continuous video stream (Dec 2023)
12. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
13. Chen, X., He, K.: Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15750–15758 (2021)
14. Denevi, G., Stamos, D., Ciliberto, C., Pontil, M.: Online-within-online meta-learning. Advances in Neural Information Processing Systems **32** (2019)
15. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019)

16. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: Proceedings of the IEEE international conference on computer vision. pp. 1422–1430 (2015)
17. Dorfman, R., Levy, K.: Adapting to mixing time in stochastic optimization with markovian data. International Conference on Machine Learning (2022)
18. Dorfman, R., Levy, K.Y.: Adapting to mixing time in stochastic optimization with markovian data. In: International Conference on Machine Learning. pp. 5429–5446. PMLR (2022)
19. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research **12**(7) (2011)
20. Duchi, J.C., Agarwal, A., Johansson, M., Jordan, M.I.: Ergodic mirror descent. SIAM Journal on Optimization **22**(4), 1549–1578 (2012)
21. Dwibedi, D., Aytar, Y., Tompson, J., Sermanet, P., Zisserman, A.: With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9588–9597 (2021)
22. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International conference on machine learning. pp. 1126–1135. PMLR (2017)
23. Finn, C., Rajeswaran, A., Kakade, S., Levine, S.: Online meta-learning. In: International Conference on Machine Learning. pp. 1920–1930. PMLR (2019)
24. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: a review. ACM Sigmod Record **34**(2), 18–26 (2005)
25. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17. pp. 286–295. Springer (2004)
26. Garg, V., Kalai, A.T.: Supervising unsupervised learning. Advances in Neural Information Processing Systems **31** (2018)
27. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. arXiv preprint arXiv:1803.07728 (2018)
28. Godichon-Baggioni, A., Werge, N., Wintenberger, O.: Learning from time-dependent streaming data with online stochastic algorithms. arXiv preprint arXiv:2205.12549 (2022)
29. Goyal, P., Duval, Q., Seessel, I., Caron, M., Misra, I., Sagun, L., Joulin, A., Bojanowski, P.: Vision models are more robust and fair when pretrained on uncurated images without supervision (Feb 2022)
30. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al.: Bootstrap your own latent-a new approach to self-supervised learning. Advances in neural information processing systems **33**, 21271–21284 (2020)
31. Gunasekara, N., Pfahringer, B., Gomes, H.M., Bifet, A.: Survey on online streaming continual learning. In: IJCAI. pp. 6628–6637 (2023)
32. Gupta, G., Yadav, K., Paull, L.: Look-ahead meta learning for continual learning. Advances in Neural Information Processing Systems **33**, 11588–11598 (2020)
33. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
34. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

35. Hsu, K., Levine, S., Finn, C.: Unsupervised learning via meta-learning. arXiv preprint arXiv:1810.02334 (2018)
36. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 97–106 (2001)
37. Javed, K., White, M.: Meta-learning representations for continual learning. Advances in neural information processing systems **32** (2019)
38. Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Torralba, A., Urtasun, R., Fidler, S.: Skip-thought vectors. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2. pp. 3294–3302. NIPS'15, MIT Press, Cambridge, MA, USA (Dec 2015)
39. KJ, J., N Balasubramanian, V.: Meta-consolidation for continual learning. Advances in Neural Information Processing Systems **33**, 14374–14386 (2020)
40. Knights, J., Harwood, B., Ward, D., Vanderkop, A., Mackenzie-Ross, O., Moghadam, P.: Temporally coherent embeddings for self-supervised video representation learning. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 8914–8921. IEEE (2021)
41. Kolter, J.Z., Maloof, M.A.: Dynamic weighted majority: An ensemble method for drifting concepts. The Journal of Machine Learning Research **8**, 2755–2790 (2007)
42. Kowshik, S., Nagaraj, D., Jain, P., Netrapalli, P.: Streaming linear system identification with reverse experience replay. Advances in Neural Information Processing Systems **34**, 30140–30152 (2021)
43. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14. pp. 577–593. Springer (2016)
44. Levin, D.A., Peres, Y.: Markov chains and mixing times, vol. 107. American Mathematical Soc. (2017)
45. Levin, D.A., Peres, Y.: Markov chains and mixing times. American Mathematical Society, Providence, RI, 2 edn. (Oct 2017)
46. Lin, L.J.: Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning **8**, 293–321 (1992)
47. Ma, S., Chen, Z., Zhou, Y., Ji, K., Liang, Y.: Data sampling affects the complexity of online sgd over dependent data. In: Uncertainty in Artificial Intelligence. pp. 1296–1305. PMLR (2022)
48. Mermillod, M., Bugaiska, A., Bonin, P.: The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. Front. Psychol. **4**, 504 (Aug 2013)
49. Mishra, N., Rohaninejad, M., Chen, X., Abbeel, P.: Meta-learning with temporal convolutions. arXiv preprint arXiv:1707.03141 **2**(7), 23 (2017)
50. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
51. Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognit. **45**(1), 521–530 (Jan 2012)
52. Nagaraj, D., Wu, X., Bresler, G., Jain, P., Netrapalli, P.: Least squares regression with markovian data: Fundamental limits and algorithms. Advances in neural information processing systems **33**, 16666–16676 (2020)

53. Nemirovski, A., Juditsky, A., Lan, G., Shapiro, A.: Robust stochastic approximation approach to stochastic programming. SIAM Journal on optimization **19**(4), 1574–1609 (2009)
54. Orabona, F.: A modern introduction to online learning. arXiv preprint arXiv:1912.13213 (2019)
55. Orhan, E., Gupta, V., Lake, B.M.: Self-supervised learning through the eyes of a child. Advances in Neural Information Processing Systems **33**, 9960–9971 (2020)
56. Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2536–2544 (2016)
57. Polyak, B.T., Juditsky, A.B.: Acceleration of stochastic approximation by averaging. SIAM J. Control Optim. **30**(4), 838–855 (Jul 1992)
58. Purushwalkam, S., Morgado, P., Gupta, A.: The challenges of continuous self-supervised learning. In: European Conference on Computer Vision. pp. 702–721. Springer (2022)
59. Radford, A., Kim, .J.W., Hallacy, .C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning
60. Radford, A., Narasimhan, K.: Improving language understanding by generative pre-training (2018), `https://api.semanticscholar.org/CorpusID:49313245`
61. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: International conference on learning representations (2016)
62. Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv:1803.00676 (2018)
63. Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., Tesauro, G.: Learning to learn without forgetting by maximizing transfer and minimizing interference. arXiv preprint arXiv:1810.11910 (2018)
64. Ross, G.J., Adams, N.M., Tasoulis, D.K., Hand, D.J.: Exponentially weighted moving average charts for detecting concept drift. Pattern recognition letters **33**(2), 191–198 (2012)
65. Runarsson, T.P., Jonsson, M.T.: Evolution and design of distributed learning rules. In: 2000 IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks. Proceedings of the First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks (Cat. No. 00. pp. 59–63. IEEE (2000)
66. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. Advances in neural information processing systems **30** (2017)
67. Song, Y., Lu, J., Lu, H., Zhang, G.: Learning data streams with changing distributions and temporal dependency. IEEE Transactions on Neural Networks and Learning Systems (2021)
68. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
69. Soviany, P., Ionescu, R.T., Rota, P., Sebe, N.: Curriculum learning: A survey. arXiv [cs.LG] (Jan 2021)
70. Sullivan, J., Mei, M., Perfors, A., Wojcik, E., Frank, M.C.: Saycam: A large, longitudinal audiovisual dataset recorded from the infant's perspective. Open mind **5**, 20–29 (2021)
71. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems **29** (2016)

72. Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by colorizing videos. In: Proceedings of the European conference on computer vision (ECCV). pp. 391–408 (2018)
73. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny, S.: Barlow twins: Self-supervised learning via redundancy reduction. In: International Conference on Machine Learning. pp. 12310–12320. PMLR (2021)
74. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14. pp. 649–666. Springer (2016)
75. Zhuang, C., Xiang, Z., Bai, Y., Jia, X., Turk-Browne, N., Norman, K., DiCarlo, J.J., Yamins, D.: How well do unsupervised learning algorithms model human real-time and life-long learning? In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (eds.) Advances in Neural Information Processing Systems. vol. 35, pp. 22628–22642. Curran Associates, Inc. (2022)
76. Žliobaitė, I., Bifet, A., Read, J., Pfahringer, B., Holmes, G.: Evaluation methods and decision theory for classification of streaming data with temporal dependence. Machine Learning **98**, 455–482 (2015)