

Learning from Streaming Video with Orthogonal Gradients

Tengda Han[◇], Dilara Gokay[◇], Joseph Heyward[◇], Chuhan Zhang[◇]
 Daniel Zoran[◇], Viorica Pătrăucean[◇], João Carreira[◇], Dima Damen^{◇†}, Andrew Zisserman^{◇‡}
[◇]Google DeepMind, [†]University of Bristol, [‡]University of Oxford

Abstract

We address the challenge of representation learning from a continuous stream of video as input, in a self-supervised manner. This differs from the standard approaches to video learning where videos are chopped and shuffled during training in order to create a non-redundant batch that satisfies the independently and identically distributed (IID) sample assumption expected by conventional training paradigms. When videos are only available as a continuous stream of input, the IID assumption is evidently broken, leading to poor performance. We demonstrate the drop in performance when moving from **shuffled** to **sequential** learning on three tasks: the one-video representation learning method DoRA, standard VideoMAE on multi-video datasets, and the task of future video prediction.

To address this drop, we propose a geometric modification to standard optimizers, to decorrelate batches by utilising orthogonal gradients during training. The proposed modification can be applied to any optimizer – we demonstrate it with Stochastic Gradient Descent (SGD) and AdamW. Our proposed orthogonal optimizer allows models trained from streaming videos to alleviate the drop in representation learning performance, as evaluated on downstream tasks. On three scenarios (DoRA, VideoMAE, future prediction), we show our orthogonal optimizer outperforms the strong AdamW in all three scenarios.

1. Introduction

Trained on Internet-scale data at powerplant-scale energy costs, the way deep learning models are created today is drastically different from the way humans acquire their visual intelligence. Humans perceive a single continuous visual input, starting from being infants in cribs. This visual input is highly redundant and temporally correlated. Such an input poses significant challenges for current deep learning paradigms. These paradigms were primarily developed for learning from images, and make assumptions on the informativeness of every training batch as an independently and identically distributed (IID) sample from the data dis-

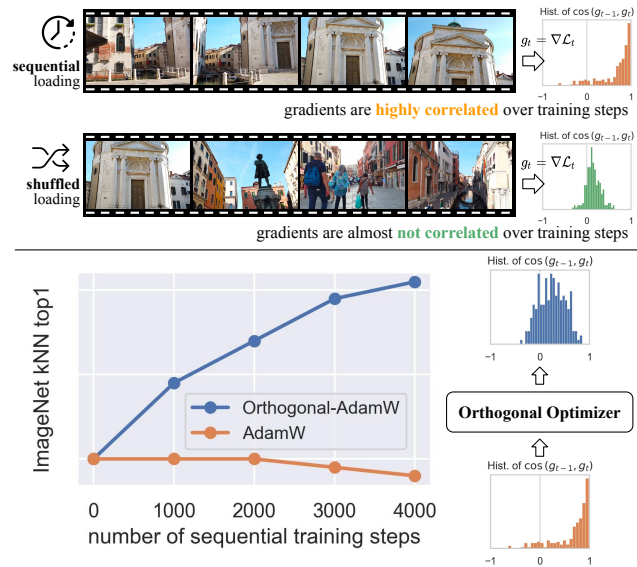


Figure 1. We address the task of learning from video by sequentially loading its clips in time (top). As neighbouring clips are very similar, consecutive gradients are highly correlated – we show the histogram of cosine similarity of gradients between consecutive batches. This causes model collapse. In contrast, current methods shuffle the video to simulate an IID input (middle). Consecutive gradients are accordingly decorrelated – cosine similarity is centred around 0. We propose to learn from the **orthogonal gradients** – which allow standard optimizers to recover the drop in performance when training from a sequential video stream (bottom).

tribution. These assumptions are immediately broken when learning from a continuous stream.

To accommodate the current learning paradigms, video models have to date been restricted to learning from short clips, by dividing any long video streams into short segments and shuffling these to enable learning. This gap between human learning and current video models is not only a computational burden from storing and accessing large videos, but is also potentially limiting the capabilities of models to achieve the human’s ability to generalise. Learning from a continuous stream is key to enabling intelligent agents that learn on-the-fly or adapt to new environments.

Additionally, models that learn from streaming videos can address privacy concerns as videos are not stored or shared.

Our paper is inspired by recent works that attempt to learn from a single video [43] or from streams of videos [7]. In [43], learning from a single long video is impressively shown to generalize but only when the video is stored in disk, such that random access is possible – with batching, random sampling and shuffling. In contrast, [7] concatenates videos to simulate a continuous stream that matches a day-long input and demonstrates the drop in performance when moving from shuffled to sequential learning on a number of self-supervised and supervised tasks.

In this work, we focus on the core obstacle to learning from streaming video: the redundancy of the data leading to highly correlated gradients. We make the following contributions:

- We quantify the drop in performance when learning from sequential, rather than shuffled, data on three video learning methods: DoRA, VideoMAE and future prediction.
- We propose to use a geometrically-principled optimizer, using the orthogonal gradient during learning.
- We augment commonly used optimizers – Stochastic Gradient Descent (SGD) and AdamW with learning from orthogonal gradients. We refer to these augmented optimizers as orthogonal optimizers.
- We showcase clear improvements using our orthogonal optimizers when learning from sequential data on all three video learning methods.

2. Related Work

Continual Learning. Existing continual learning literature focuses on the learning dynamics and the effects of introducing novel tasks over the training cycle, emphasizing knowledge accumulation as new tasks and data become available. The main objective of such works is quick adaptation while preserving performance on previously learned tasks (failure to do so is commonly termed as “Catastrophic Forgetting”) [25]. In the context of continual learning, task changes over the training progress – it could be a different objective function, or incremental annotated labels [5, 35]. But often the data in these tasks consists of independent images [28, 37], which are much less correlated than consecutive video frames.

Various approaches have been proposed to tackle these problems: input replay buffers, which make the learning problem closer to the IID case by accumulating a dataset to sample from; architectural adjustments [2, 30, 46], adapting the optimization algorithm [25], or redesigning the training paradigm *e.g.* adding pre-training with IID data [31, 36]. Orthogonal gradients have been explored in the context of continual learning, where the model learns a number of distinct tasks iteratively [14] – in this case orthogonal gradients were used to avoid catastrophic forget-

ting of previously learned classes when learning continual image classification. The orthogonal computation is only computed after training each task.

Different from prior work in continual learning, we address the problem of a *single* task learnt from *continuous* videos, where the learning process unfolds along the temporal dimension of visual sequences. This task is particularly challenging because in addition to the the problem of catastrophic forgetting due to the extensive temporal history, the continuity of the video frames introduces high correlations between consecutive learning steps which can be detrimental to the learning process. We revisit [14], extending it to multiple optimizers and testing it for the first time on video tasks in general and streaming videos in particular.

Learning from Video Streams. The majority of video models trained today are trained from randomly sampled short clips sampled from large video datasets, creating roughly IID samples which make training with stochastic gradient descent effective.

One exception is the work by Purushwalkam *et al.* [34], which explores learning a self-supervised model from continuous video streams. This work uses a ‘replay buffer’ to store recent training samples in order to overcome the high temporal correlation of streamed videos. Another work, closer to ours, is the ‘Baby Learning’ framework [7]. In that work, a future prediction model is trained on streaming video and is evaluated on both in-stream and out-of-stream tasks – trading off adaptation and generalization. This work includes experiments with a variety of common optimizers, but does not explicitly deal with the temporal correlation of the gradients. Another line of work automatically filters training samples [3, 13] – this can be applied to streamed video learning scenarios to handle the high temporal correlation of gradients. However, such methods effectively load more data than is actually used for training, and the result is quite similar to using a replay buffer, requiring extra compute and memory.

Video Representation Learning. Rapid progress has been made in visual representation learning from images and videos, especially in the family of self-supervised methods. These can be grouped into three main core ideas – contrastive learning (CPC [32], MoCo [21], SimCLR [8], DPC [20]), self-distillation (BYOL [18], DINO [6]) and reconstruction based methods (MAE [22], VideoMAE [15, 42], SiameseMAE [19]). However, all these techniques rely on training with IID data randomly sampled from large, shuffled training datasets, which contrasts with the sequential nature in which humans, for example, perceive visual information. This paper explores some of these models when applied to the streaming video input scenario.

Test-time adaptation. When encountering distribution shifts at test time, models often fail to adapt or produce reasonable results given the new data. Test-time adaptation methods attempt to address this issue by either introducing training objectives which can be applied at test time [26] - these usually would be self-supervised objectives [12, 39] - or by adding regularization terms to an already trained model [27]. These ideas have been recently transposed to language models [40] and connections to in-context learning and online reasoning are now actively being pursued [1, 12, 33, 44, 45]. Here we demonstrate that test-time adaptation on video streams benefits from using orthogonal gradients.

3. Method

Problem Setup. We focus on the hard problem of sequential learning from a single continuous video. Given a long video \mathcal{V} , our goal is to train a model f_θ on the video \mathcal{V} *sequentially* to minimize an objective function \mathcal{L} , where θ represents the network parameters. Since the entire video \mathcal{V} is too long to feed into the model at once, a practical approach is to cut the video into small chunks, $\{\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_n\}$, where each \mathcal{V}_i denotes i -th video clip with a short temporal window. Different to the common practice of randomly sampling clips as mini batches to train the model, we wish to learn from a video stream, so clips are fed in their *sequential* order.

The greatest obstacles to learning from video sequentially is the high temporal correlation of gradients. In most cases, the video changes slowly and the gradient of the current batch is almost identical to that of the previous batch. The stochastic optimization methods widely used in deep learning, such as SGD, are all based on the assumption that the global gradient can be approximated by the gradient of mini-batches to some extent, which does not hold true when learning from sequential videos. We focus on the task of learning from long videos in a *self-supervised* manner where the learning signal purely comes from the pixels, and the temporal correlation of gradient is severe. This is distinct from supervised learning where the supervisory signal might provide insights on where subtle changes or informative content is. Our objective is a mechanism that can learn from these subtle changes; in effect, able to continually decorrelate the gradients and learn from the residual.

Learning from Orthogonal Gradients. Our method is straightforward: as the gradients are temporally correlated, we propose to learn from the orthogonal components of the gradients. In detail, the gradients of two consecutive update steps can be written as $g_{t-1} = \nabla_\theta \mathcal{L}_{t-1}$ and $g_t = \nabla_\theta \mathcal{L}_t$, where θ denotes the model parameters and \mathcal{L} is the loss function. In an idealistic training scenario

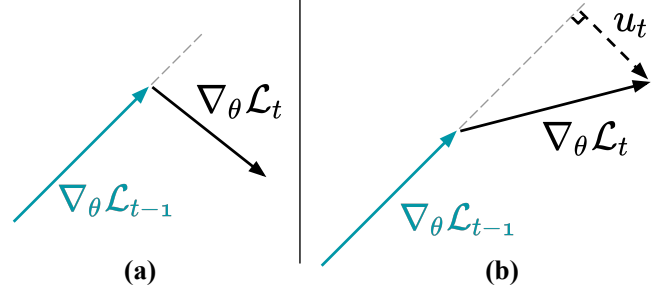


Figure 2. A simplified illustration of orthogonal gradients. (a) In common IID training, the gradient between consecutive steps are not very correlated due to the IID nature. (b) Whereas if learning from sequential videos, the gradients between consecutive steps are highly correlated, which harms the optimization. We propose to update the model parameters from the orthogonal components of the current gradient, denoted as u_t . In practice, the gradients and the orthogonal operation are in a high dimensional space.

where the data samples in subsequent mini-batches follow the IID distribution, these two gradients typically have low similarity, which can be measured by a cosine distance $\cos(g_{t-1}, g_t) \approx 0$.

When training sequentially, empirically we find the gradients between two consecutive update steps can be highly similar, *i.e.* $\cos(g_{t-1}, g_t) \rightarrow 1$, as shown in Figure 1. To decorrelate these gradients, we propose to only update with the orthogonal component of the gradient g_t w.r.t. the past gradient g_{t-1} for the optimization step. As illustrated in Figure 2, the actual gradient used for the update is

$$u_t = g_t - \text{proj}_{g_{t-1}} g_t \quad (1)$$

where $\text{proj}_{g_{t-1}}(g_t)$ is the projection operation onto the direction g_{t-1} :

$$\text{proj}_{g_{t-1}}(g_t) = \frac{g_t \cdot g_{t-1}}{g_{t-1} \cdot g_{t-1}} g_{t-1} = \frac{\|g_t\| \cos(g_t, g_{t-1})}{\|g_{t-1}\|} g_{t-1} \quad (2)$$

This orthogonal gradient update has ideal behaviour for two scenarios at either end of the correlation spectrum: (1) when the training data is close to an IID distribution, *i.e.* $\cos(g_{t-1}, g_t) \approx 0$, the orthogonal gradient u_t is close to the original gradient, since $u_t = g_t - \text{proj}_{g_{t-1}} g_t \approx g_t$. It means the orthogonal gradient based optimization rule is compatible with IID training scenario. In contrast, (2) when the consecutive data samples have high sequential similarity, *i.e.* $\cos(g_{t-1}, g_t) \approx 1$, the orthogonal gradient has a small magnitude on a new direction $u_t = g_t - \text{proj}_{g_{t-1}} g_t \approx g_t - \frac{\|g_t\|}{\|g_{t-1}\|} g_{t-1}$. A small gradient results in minor changes to the model's parameters. This avoids the model to be excessively optimized along one gradient direction, when there is insufficient new signal.

Practically, decorrelating the current gradient with the past *single* step can be sensitive to noise. Inspired by the

Algorithm 1 Orthogonal SGD

Require: Learning rate $\eta > 0$, momentum parameter $\beta \in [0, 1)$, initial parameters θ_0 , number of iterations T

- 1: Initialize velocity $c_0 = 0$
 - 2: **for** $t = 1$ to T **do**
 - 3: Sample a mini-batch of data \mathcal{B}_t from the training set
 - 4: Compute the gradient: $g_t = \nabla_{\theta} \mathcal{L}(\theta_{t-1}; \mathcal{B}_t)$
 - 5: Compute the orthogonal gradient: $u_t = g_t - \text{proj}_{c_{t-1}} g_t$
 - 6: Update the raw momentum: $c_t = \beta c_{t-1} + (1 - \beta) g_t$
 - 7: Overwrite the gradient: $g_t := u_t$
 - 8: Update the parameters: $\theta_t = \theta_{t-1} - \eta g_t$
 - 9: **end for**
-

common usage of ‘momentum’ in standard optimizers [41], we maintain an exponential moving average (EMA) of the original ‘clean’ gradients, denoted by c_t , with an update rule

$$c_t := \beta c_{t-1} + (1 - \beta) g_t \quad (3)$$

where β is the momentum factor, by default we use $\beta = 0.9$. The orthogonal gradient is then computed by

$$u_t = g_t - \text{proj}_{c_{t-1}} g_t \quad (4)$$

Notice that the EMA is computed on the original gradients, rather than the orthogonal component u_t , whereas u_t can be further fed into first/second order moment subject to the choices of optimizers (e.g. second-order optimizer AdamW in Algorithm 2).

Importantly, the aforementioned geometric modification is applicable to many optimizers. Here we show two commonly used optimizer algorithms modified by orthogonal gradients: SGD optimizer as an illustration (Algorithm 1), and the AdamW optimizer [29] (Algorithm 2). The text in green indicates the addition to the original algorithms. We mostly experiment with Orthogonal-AdamW due to its faster convergence speed.

Trade-off between algorithm and speed. In the convex optimization literature, there are relevant methods that might be more favourable than orthogonal operation, such as conjugate gradient method [23, 38]. However, orthogonal gradient is computationally cheaper than conjugation, since the orthogonal projection can be implemented as cosine distance and vector norms (Equation 2), which could take advantages from well-optimized pre-compiled kernels in deep learning toolboxes. We do not delve into this direction in this paper, but it could be an interesting future work.

Algorithm 2 Orthogonal AdamW

Require: Learning rate $\eta > 0$, weight decay coefficient $\lambda > 0$, decay rates $\beta, \beta_1, \beta_2 \in [0, 1)$, small constant $\epsilon > 0$, initial parameters θ_0 , number of iterations T

- 1: Initialize first moment vector $m_0 = 0$, $c_0 = 0$, and second moment vector $v_0 = 0$
 - 2: **for** $t = 1$ to T **do**
 - 3: Sample a mini-batch of data \mathcal{B}_t from the training set
 - 4: Compute the gradient: $g_t = \nabla_{\theta} \mathcal{L}(\theta_{t-1}; \mathcal{B}_t)$
 - 5: Compute the orthogonal gradient: $u_t = g_t - \text{proj}_{c_{t-1}} g_t$
 - 6: Update the raw momentum: $c_t = \beta c_{t-1} + (1 - \beta) g_t$
 - 7: Overwrite the gradient $g_t := u_t$
 - 8: Update biased first moment estimate: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
 - 9: Update biased second moment estimate: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
 - 10: Compute bias-corrected first moment: $\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$
 - 11: Compute bias-corrected second moment: $\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$
 - 12: Apply weight decay: $\theta_{t-1} = \theta_{t-1} - \eta \lambda \theta_{t-1}$
 - 13: Update parameters: $\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$
 - 14: **end for**
-

4. Experiments

In this section, we focus on providing empirical evidence using real video datasets to demonstrate the effectiveness of the orthogonal optimizer. We particularly experiment with Orthogonal-AdamW, on three scenarios: representation learning on a *single* long video, representation learning on video datasets, and future prediction tasks as same as [7].

4.1. DoRA on a Single Video

The DoRA paper [43] trains a vision transformer [11] image backbone on a single long video and achieves competitive performance. They apply aggressive frame augmentations and randomly sample short video clips, similar to other self-supervised works [6, 21], to obtain diverse training samples. Differently, we focus on learning video representation from a single video in a *sequential* manner, which poses a great challenge to these prior methods because of the high temporal correlation between batches.

Datasets. Following DoRA [43], we use the WalkingTour video at Venice (denoted as $\text{WT}_{\text{venice}}$) from the **Walking-Tour** dataset proposed by the same work. This video is extensively used by DoRA and enables us to conduct a through analysis. The $\text{WT}_{\text{venice}}$ video has a duration of 1 hour 50 minutes at 60fps, containing a continuous urban view around Venice city center filmed from a hand-held

initialization	pretraining dataset: WT _{venice}		downstream ImageNet	
	pretraining method	optimizer	linear probe top1	kNN top1
DINO _{ImageNet}	-	-	-	74.4
DINO _{ImageNet}	DoRA sequential (batch-along-time)	AdamW	6.1	1.8
DINO _{ImageNet}	DoRA sequential (batch-along-time)	Orthogonal-AdamW	64.5	51.8
VideoMAE _{SSV2}	-	-	-	3.7
VideoMAE _{SSV2}	DoRA sequential (batch-along-time)	AdamW	7.9	3.0
VideoMAE _{SSV2}	DoRA sequential (batch-along-time)	Orthogonal-AdamW	11.2	5.7
random	DoRA sequential (batch-along-time)	AdamW	3.5	0.8
random	DoRA sequential (batch-along-time)	Orthogonal-AdamW	8.2	3.1

Table 1. Experiments on DoRA [43] pretraining on WT_{venice}

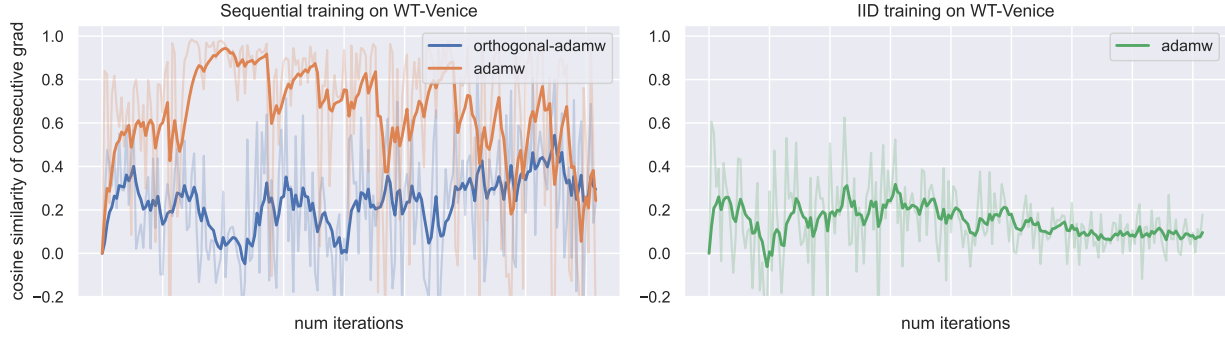


Figure 3. Effect of orthogonal optimizer on sequential training of DoRA on the WT_{venice} video. On IID training, the consecutive gradient has low cosine similarity (right). Sequential training (left) naturally brings a high similarity of consecutive gradient, but the orthogonal optimizer decorrelate the gradients over time. Notice that we plot $\cos(g_{t-1}, g_t)$ in this figure.

camera. As in [43], we train on this video, albeit in a sequential manner, and then evaluate the learnt representation on the downstream task of object recognition on ImageNet [10].

Task Setting. We train the DoRA method using their official codebase. Instead of randomly sampling short video clips, we sample clips sequentially from the beginning of the video. *i.e.* given a batch size N , our first batch $\mathcal{B}_1 = \{C_1, \dots, C_N\}$ and the second batch continues from C_{N+1} to C_{2N} , where C_i denotes the i -th short clip from the source video. Practically, each video clip contains 8 consecutive video frames sampled at 1 fps, and every two consecutive clips are shifted by 1 frame, or 1/60 second. For evaluation, we monitor the performance of ImageNet linear probe and k-nearest-neighbour classification performance, same as DoRA. We monitor the performance drop due to sequential video training, and observe how much gain the orthogonal optimizer can reclaim.

Architecture. We use the same architecture as DoRA, which consists of two ViT-S image backbones, forming a teacher-student structure. Each ViT-S backbone contains 12 transformer blocks with 384 embedding dimension. Their training scheme is inspired by DINO [6] – the ‘teacher’ module is updated from an exponential moving

average (EMA) of the student’s parameters. The DoRA architecture also contains a multi-object tracking module, which masks out objects based on the attention scores among the image patches produced by the teacher module. The teacher module has the privilege to observe the full video frames as input, whereas the student module takes as input either heavily cropped video frames, or partially masked frames, and is trained to produce a representation that is close to the teacher’s output; the student module is trained with gradient back propagation. For evaluation, we take the backbone of the teacher module and perform downstream tasks, same as DoRA.

Implementation Details. The original DoRA is trained from scratch for a long time (10+ days on 16 GPUs). We train DoRA with different initialization methods including DINO weights pretrained on ImageNet, and VideoMAE weights pretrained on Something-Something-V2 (SSV2) [16], and random initialization. By default, we train DoRA for 1 epoch on the WT_{venice} with a batch size of 32 video clips distributed on 4 Nvidia A100 GPUs. For ImageNet classification and kNN evaluation, we use the same setting as DoRA’s codebase. The full implementation detail can be found in the supplementary material.

DoRA Discussion. Figure 3 illustrates the cosine similarity between consecutive gradients (*i.e.* $\cos(g_{t-1}, g_t)$) when training DoRA on WT_{venice}. It is clear that in sequential training scenario, the proposed Orthogonal-AdamW is able to reduce the gradient correlation over time (orange vs. blue), getting closer to the low correlation in IID sampling scenario (green).

The experimental results of training DoRA sequentially are shown in Table 1. When initializing with a strong DINO_{ImageNet} checkpoint, the Orthogonal-AdamW optimizer is able to prevent the training failure; whereas with the baseline AdamW optimizer, the model parameters are damaged by the sequential training and cannot be trained further. With VideoMAE_{SSV2} initialization, in a short training schedule the Orthogonal-AdamW optimizer surpasses AdamW on the same setting (3.0 to 5.7 on kNN accuracy). We note that VideoMAE_{SSV2} initialization gives much worse results on downstream ImageNet classification performance. This is possibly because the SSV2 dataset does not have enough diversity for general objects. We also experimented training DoRA from scratch, although the sequential training of DoRA is inefficient, the Orthogonal-AdamW outperforms AdamW by a clear margin, and with AdamW the model does not train.

4.2. VideoMAE on Video Datasets

For general self-supervised video representation learning, VideoMAE [15, 42] remains a competitive method which learns from reconstructing video patches, but it was mostly applied on large scale video datasets with large diversity. In this section, we generalize the proposed orthogonal optimizer to VideoMAE training on common video datasets rather than a single video, but in a sequential manner.

Batching strategy for sequential videos. From one video, loading clips in a sequential manner is straightforward. But when the dataset has multiple videos, or there are multiple video streams available simultaneously, two different ways of forming mini-batches emerge. As shown in Figure 4 (a), one can batch video clips over the time axis, and go through videos one by one in the dataset, such as $\mathcal{B}_1 = \{V_1C_1, V_1C_2, \dots\}$. But if the video in the dataset is not long enough w.r.t. the batch size, the next batch might sample clips from a different video source (not from V_1). As shown in Figure 4 (b), one can also batch video clips over different videos, *e.g.* $\mathcal{B}_1 = \{V_1C_1, V_2C_1, \dots\}$, and the next batch will sample videos from the next timestamp. In this section, we experiment with both batching methods, named as ‘batch-along-time’ and ‘batch-along-video’.

Datasets. We use Something-Something-V2 and Kinetics-400 as pretraining datasets, to be comparable with VideoMAE [42]. **Something-Something-V2 (SSV2)** [16]

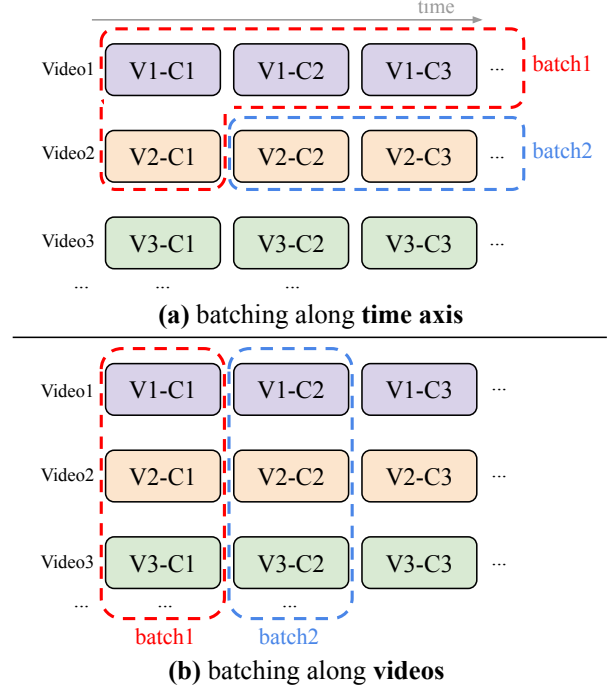


Figure 4. Two batch strategies for sequential video datasets, for videos V_i divided into clips $\{C_1, \dots, C_N\}$. **(a) batch along the time axis:** a more practical way of batching long video streams, where the samples within a batch have high correlation. But when the batch size is large, the temporal correlation between consecutive batches might be low. **(b) batch along videos:** samples within a batch are diverse but the temporal correlation between consecutive batches is high. Notice that in practice adjacent clips may have temporal overlaps, for clarity we do not show any overlaps in the figure.

is a fine-grained action classification dataset focusing on object manipulation. It consists of 220k short videos with duration between 2 to 6 seconds, which are labelled into 174 classes emphasising the action, such as ‘moving something from left to right’. **Kinetics-400 (K400)** [24] is a large scale action classification dataset sourced from internet videos. It contains 230k videos with duration of 10 seconds, spanning over 400 general human action classes. For downstream evaluation, we report action classification results on SSV2 dataset.

Task Setting. We pretrain a VideoMAE model *from scratch*, on both SSV2 and K400 datasets. Differently to common practice that randomly samples short video clips from each video in the dataset and applies shuffling, we load videos in a sequential manner with both batching strategies illustrated in Figure 4. To evaluate the quality of learned representation, we apply two methods: linear-probe and attn-probe. **Linear-probe** means a single linear layer on top of the frozen pre-trained visual encoder is trained for the action classification task; **Attn-probe** means attentive probing

pretraining dataset: SSV2		downstream SSV2	
video processing	optimizer	linear-probe top1↑	attn-probe top1↑
VideoMAE _{SSV2} [42]	AdamW	23.2	55.7
shuffled video clips	AdamW	19.0	54.9
shuffled video clips	Orthogonal-AdamW	21.0	54.7
sequential (batch-along-time)	AdamW	16.4	46.1
sequential (batch-along-time)	Orthogonal-AdamW	18.4	48.0
sequential (batch-along-video)	AdamW	9.5	30.3
sequential (batch-along-video)	Orthogonal-AdamW	10.4	32.6

pretraining dataset: K400		downstream SSV2	
video processing	optimizer	linear-probe top1↑	attn-probe top1↑
VideoMAE _{K400} [42]	AdamW	19.2	52.1
shuffled video clips	AdamW	20.3	46.3
shuffled video clips	Orthogonal-AdamW	21.4	48.4
sequential (batch-along-time)	AdamW	19.3	44.7
sequential (batch-along-time)	Orthogonal-AdamW	20.5	46.5
sequential (batch-along-video)	AdamW	18.7	43.5
sequential (batch-along-video)	Orthogonal-AdamW	18.2	43.6

Table 2. Experiments on VideoMAE pretraining on SSV2 and K400. The experiment in gray is our downstream evaluation results with the official checkpoint obtained from [42].

used in [4]: a single transformer block including attention operation and MLP layers is trained on top of the frozen pre-trained visual encoder for action classification task.

Architecture. We use a Vision Transformer [11] ViT-B as the visual encoder, which consists of 12 transformer blocks with an embedding dimension of 768. As part of the MAE training, we use a visual decoder which consists of 4 transformer blocks, which is trained to reconstruct visual patches from the encoder outputs, and will be discarded when evaluating for downstream tasks. We train the VideoMAE with the default 0.9 drop ratio, which means 90% of the visual patches will be discarded for the visual encoder and will be reconstructed by the visual decoder. The entire network is trained from scratch.

Implementation Details. The model takes 16 frames at 224×224 resolution as input. For sequential loading, we first take all the frames from each video, then sample 16-frame clips from that. In order to have a clear comparison for both batching methods, we sample the same number of clips from each video. For example, on SSV2 we first take 64 uniformly-sampled frames from each video, then take 4 clips without overlap, each clip containing 16 frames; Similarly, on K400 we first take 112 uniformly-sampled frames from each video, then take 7 clips without overlap, each clip containing 16 frames. With this, the models trained with both strategies observe exact same video clips, but only different in the batch arrangement. For IID sampling and downstream tasks, we use the default strategy

as in [42], where a 16-frame clip is randomly sampled from each video. For our pretraining experiments, the model is trained with a batch size of 512 clips for the same number of iterations (260k steps), for a fair comparison. Other implementation details are in the supplementary material.

Discussion. The results are shown in Table 2. First, notice that there is no big drop when switching from IID sampling to the ‘batch-along-time’ sequential sampling, *e.g.* linear probe $19.0 \rightarrow 16.4$ for SSV2, $20.3 \rightarrow 19.3$ for K400. The reason is the videos in SSV2 and K400 are relatively short compared with our batch size (512 clips), the consecutive batches actually contain clips from different video sources. Second, it is expected that ‘batch-along-video’ gives worse results than ‘batch-along-time’ due to larger temporal correlation between batches. Third, proposed Orthogonal-AdamW optimizer works better than the baseline AdamW on both sequential cases, *e.g.* attn probe top1 +2% when pretrained on SSV2, and +1% when pretrained on K400. Additionally, it is interesting that the Orthogonal-AdamW also works slightly better on shuffled clips, *e.g.* linear probe top1 +2% when pretrained on SSV2 and +1% when pretrained on K400. Probably it is because the inter-batch correlations from shuffled clips on SSV2 and K400 datasets are significant enough, that decorrelating the gradients bring some small gains.

4.3. Future Prediction on Video Streams

In this section, we reproduce the experiments of learning from video streams from Carreira *et al.* [7], and experiment

displacement +0.64s			Ego4D: Pixel MSE↓ / PSNR↑		ScanNet: Pixel MSE↓ / PSNR↑	
method (batch-along-time)	pretraining	optimizer	in-s.	out-of-s.	in-s.	out-of-s.
BabyLearning [7]	Guided Future Prediction	RMSProp	0.055 / -	0.066 / -	0.055 / -	0.061 / -
BabyLearning [7]†	ViT-L-I21K-CLS	RMSProp	0.059 / -	0.073 / -	0.061 / -	0.066 / -
BabyLearning (repro)	ViT-L-I21K-CLS	RMSProp	0.032 / 15.9	0.026 / 16.9	0.033 / 15.07	0.041 / 14.28
BabyLearning	ViT-L-I21K-CLS	AdamW	0.034 / 15.9	0.026 / 16.8	0.033 / 15.72	0.033 / 15.27
BabyLearning	ViT-L-I21K-CLS	Orthogonal-AdamW	0.031 / 16.4	0.023 / 17.6	0.032 / 15.77	0.033 / 15.28

Table 3. Performance on future frame prediction task on Ego4D-Stream and ScanNet-Stream datasets, compared with [7]. † this result are obtained by contacting the authors. The ‘(repro)’ denotes our reproduction of the experiment from [7] with a same setting.

with our orthogonal optimizer on this sequential training scenario.

Datasets. Following [7], we use ScanNet-Stream and Ego4D-Stream datasets. **ScanNet-Stream** is a continuous version of ScanNet-V2 proposed in [7], which simply stitches all the videos together to mimic a long video and to experiment with sequential loading. ScanNet-V2 [9] contains videos of in-door room scanning scenario, with an average duration of 1 minute, together with synchronized depth masks, semantic segmentation masks, and camera poses. We use the same train-val split as [7] – 1.2k original ScanNet videos for training and 312 for validation. Similarly **Ego4D-Stream** is a stitched version of Ego4D [17]. Ego4D is a large scale egocentric video dataset contains various daily activities. Each Ego4D video has an average duration of 9 minutes. We use the same train-val split as [7] – 21.7k original videos for training and 2.3k for validation.

Task Setting. We follow the same task setting as [7] but only change the optimizer. Specifically, the model takes 4 video frames as input, and is trained to predict another 4 video frames in the future, with a time displacement of 0.16s or 0.64s. We use the more challenging time displacement of 0.64s. We experiment on the pixel prediction task on both datasets, *i.e.* the model is trained to predict future pixels, in a sequential way. For evaluation, we monitor both the in-stream and out-of-stream performance introduced in [7], in other words, we report the temporally aggregated performance on the training video stream and also the validation video stream. This setting can be viewed as a test-time adaptation scenario, that a pretrained model is expected to adapt well on one video stream (in-stream performance), as well as keep its generalizability on other unseen video streams (out-of-stream performance).

Architecture. We use a ViT-L backbone pretrained on ImageNet-21K classification task as in [7]. Notice that [7] also uses a stronger ‘Guided Future Prediction’ pretraining checkpoint which we are not able to reproduce. The output of the ViT-L backbone is fed to a randomly-initialized linear layer for future pixel prediction task. The entire model

including the pretrained backbone and the linear layer is trained end-to-end.

Implementation Details. The model is trained on 24h of training video stream at 25fps, given that at each training step, the model takes 4 frames as input without overlapping, which would be 540k training samples ($24 \times 3600s \times 25fps/4$). Following [7] that accumulates gradients every 16 training steps, equivalently we train the model with a batch size of 16, using the ‘batch-along-time’ setting. All the experiments use a learning rate of 10^{-4} , and a cosine-decayed learning rate schedule with linear warm-up. We report pixel mean squared error (MSE) and peak signal-to-noise ratio (PSNR) for the future frame prediction task. A lower MSE and a higher PSNR indicate better performance.

Discussion. The results are shown in Table 3. Notice that our reproduction using the same setting as [7] (ViT-L-I21K-CLS, with RMSProp optimizer) performs better than the reported results on pixel MSE (0.032 / 0.026 vs. 0.059 / 0.073 on Ego4D in/out-of-stream). The proposed Orthogonal-AdamW optimizer further surpasses the baseline AdamW and RMSProp optimizer on both Ego4D-Stream and ScanNet-Stream, on both in-stream and out-of-stream performance. The in-stream improvements observed indicate our Orthogonal-optimizer can be used for other test-time adaptation tasks beyond representation learning.

5. Conclusion

We propose a simple geometric modification to standard optimizers that update with *orthogonal* gradients during training, in order to decorrelate consecutive batches when training from continuous streams of videos. We demonstrate three training scenarios which operates on sequential videos: representation learning from a single long video, representation learning from large-scale multi-video datasets, and the task of future frame prediction.

Our experiments show that the orthogonal optimizer, in particular Orthogonal-AdamW, is able to regularize the learning process and obtain better performance than baseline optimizers for all three tasks.

Acknowledgement

We thank James Martens for technical suggestions on the optimizers, and Jean-Baptiste Alayrac and Matthew Grimes for reviewing the manuscript. We also thank Carl Doersch, Ignacio Rocco, Michael King, Yi Yang and Yusuf Aytar for helpful discussions.

References

- [1] Ekin Akyürek, Mehul Damani, Linlu Qiu, Han Guo, Yoon Kim, and Jacob Andreas. The surprising effectiveness of test-time training for abstract reasoning. <https://ekinakyurek.github.io/papers/ttt.pdf>, 2024. 3
- [2] Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. In *NeurIPS*, 2019. 2
- [3] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *NeurIPS*, 2019. 2
- [4] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024. 7
- [5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *arXiv preprint arXiv:2004.07211*, 2020. 2
- [6] 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. 2, 4, 5
- [7] João Carreira, Michael King, Viorica Patraucean, Dilara Gokay, Catalin Ionescu, Yi Yang, Daniel Zoran, Joseph Heyward, Carl Doersch, Yusuf Aytar, et al. Learning from one continuous video stream. In *CVPR*, 2024. 2, 4, 7, 8
- [8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*. PMLR, 2020. 2
- [9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 8
- [10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5
- [11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4, 7, 2, 3
- [12] Nikita Durasov, Assaf Shocher, Doruk Oner, Gal Chechik, Alexei A Efros, and Pascal Fua. IT³: Idempotent test-time training. *arXiv preprint arXiv:2410.04201*, 2024. 3
- [13] Talfan Evans, Nikhil Parthasarathy, Hamza Merzic, and Olivier J Henaff. Data curation via joint example selection further accelerates multimodal learning. *NeurIPS*, 2024. 2
- [14] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020. 2
- [15] Christoph Feichtenhofer, Yanghao Li, Kaiming He, et al. Masked autoencoders as spatiotemporal learners. *NeurIPS*, 2022. 2, 6
- [16] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The “something something” video database for learning and evaluating visual common sense. In *ICCV*, 2017. 5, 6
- [17] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, pages 18995–19012, 2022. 8
- [18] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent—a new approach to self-supervised learning. *NeurIPS*, 33:21271–21284, 2020. 2
- [19] Agrim Gupta, Jiajun Wu, Jia Deng, and Fei-Fei Li. Siamese masked autoencoders. In *NeurIPS*, 2023. 2
- [20] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *ICCV workshops*, 2019. 2
- [21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 2, 4
- [22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 2
- [23] Magnus Rudolph Hestenes, Eduard Stiefel, et al. *Methods of conjugate gradients for solving linear systems*. NBS Washington, DC, 1952. 4
- [24] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 6
- [25] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2
- [26] Jian Liang, Ran He, and Tien-Ping Tan. A comprehensive survey on test-time adaptation under distribution shifts. *arXiv preprint arXiv:2303.15361*, 2023. 3

- [27] Wei Lin, Muhammad Jehanzeb Mirza, Mateusz Kozinski, Horst Possegger, Hilde Kuehne, and Horst Bischof. Video test-time adaptation for action recognition. *arXiv preprint arXiv:2211.15393*, 2022. [3](#)
- [28] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *NeurIPS datasets and benchmarks track*, 2021. [2](#)
- [29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. [4](#)
- [30] Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *CVPR Workshops*, 2021. [2](#)
- [31] Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. *Journal of Machine Learning Research*, 24(214):1–50, 2023. [2](#)
- [32] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. [2](#)
- [33] OpenAI. Introducing OpenAI O1 preview, 2024. [3](#)
- [34] Senthil Purushwalkam, Pedro Morgado, and Abhinav Gupta. The challenges of continuous self-supervised learning. In *ECCV*, 2022. [2](#)
- [35] Haoxuan Qu, Hossein Rahmani, Li Xu, Bryan Williams, and Jun Liu. Recent advances of continual learning in computer vision: An overview. *arXiv preprint arXiv:2109.11369*, 2024. [2](#)
- [36] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *ICLR*, 2022. [2](#)
- [37] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. [2](#)
- [38] Jonathan Richard Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain. 1994. [4](#)
- [39] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A Efros, and Moritz Hardt. Test-time training for out-of-distribution generalization. 2019. [3](#)
- [40] Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*, 2024. [3](#)
- [41] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, 2013. [4](#)
- [42] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, 2022. [2](#), [6](#), [7](#)
- [43] Shashanka Venkataramanan, Mamshad Nayeem Rizve, João Carreira, Yuki M Asano, and Yannis Avrithis. Is ImageNet worth 1 video? learning strong image encoders from 1 long unlabelled video. In *ICLR*, 2024. [2](#), [4](#), [5](#)
- [44] Kevin Wang, Junbo Li, Neel P. Bhatt, Yihan Xi, Qiang Liu, Ufuk Topcu, and Zhangyang Wang. On the planning abilities of openai’s o1 models: Feasibility, optimality, and generalizability. *arXiv preprint arXiv:2409.19924*, 2024. [3](#)
- [45] Renhao Wang, Yu Sun, Yossi Gandelsman, Xinlei Chen, Alexei A Efros, and Xiaolong Wang. Test-time training on video streams. *arXiv preprint arXiv:2307.05014*, 2023. [3](#)
- [46] Yujie Wei, Jiaxin Ye, Zhizhong Huang, Junping Zhang, and Hongming Shan. Online prototype learning for online continual learning. In *ICCV*, 2023. [2](#)