

Learning from One Continuous Video Stream

João Carreira^{1†}, Michael King^{1†}, Viorica Pătrăucean^{1†}, Dilara Gokay^{1†}, Cătălin Ionescu^{1†},
Yi Yang[†], Daniel Zoran[†], Joseph Heyward[†], Carl Doersch[†], Yusuf Aytar[†],
Dima Damen^{†‡}, Andrew Zisserman^{†◇}

[†]Google DeepMind, [‡]University of Bristol, [◇]University of Oxford
Corresponding author: joaoluis@google.com, ¹core contributor

Abstract

We introduce a framework for online learning from a single continuous video stream – the way people and animals learn, without mini-batches, data augmentation or shuffling. This poses great challenges given the high correlation between consecutive video frames and there is very little prior work on it. Our framework allows us to do a first deep dive into the topic and includes a collection of streams and tasks composed from two existing video datasets, plus methodology for performance evaluation that considers both adaptation and generalization. We employ pixel-to-pixel modelling as a practical and flexible way to switch between pre-training and single-stream evaluation as well as between arbitrary tasks, without ever requiring changes to models and always using the same pixel loss. Equipped with this framework we obtained large single-stream learning gains from pre-training with a novel family of future prediction tasks, found that momentum hurts, and that the pace of weight updates matters. The combination of these insights leads to matching the performance of IID learning with batch size 1, when using the same architecture and without costly replay buffers. An overview of the paper is available online at <https://sites.google.com/view/one-stream-video>.

1. Introduction

Humans gather knowledge about themselves and the world through a continuous stream of observations, from their early days as infants. Some experiences are seen once in a lifetime while others are daily repeats. With time, familiarity with environments, objects and experiences form a base to our knowledge, and memory maintains the most important experiences from our ever-further past. Learning successfully *adapts* to the current surroundings so we can make better predictions over time, while leading to *generalization* to unseen environments. Despite the naturalness

of this approach of learning for humans, video understanding approaches have rarely attempted a similar regime.

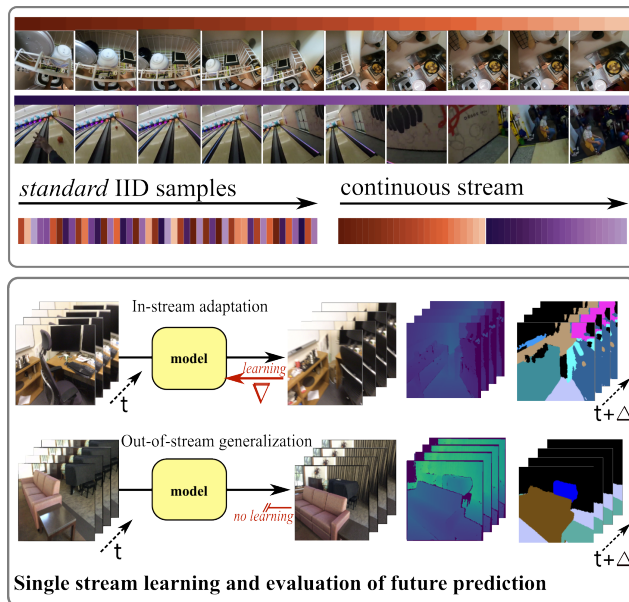


Figure 1. **Top:** We introduce a framework for studying continuous learning in a single video stream. This is a natural yet unstudied problem, different from standard independent and identically distributed (IID) learning in video where batches contain clips from random videos in a random order. **Bottom:** We propose pixel-to-pixel models to evaluate our approach across prediction tasks (prediction of future frames, depth, segmentation). We measure both adaptation to the video stream – the model here updates its weights (learns) continuously to improve prediction – as well as generalization to out-of-stream clips – the model being adapted on the first stream is now evaluated on a different held-out stream without being allowed to adapt to it. We propose to maximize both adaptation and generalization.

The most related fields are continual and lifelong learning, but these are still deeply fractured – no single problem formulation or benchmark is widely accepted. They also mostly focus on simplifications of the problem relative to

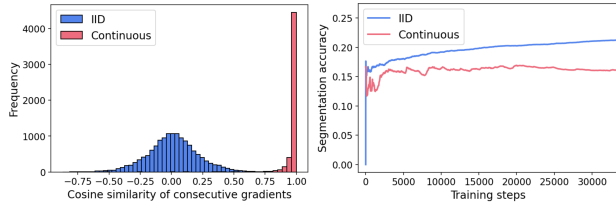


Figure 2. UNet training on ScanNet-stream for the semantic segmentation task. **Left:** The cosine similarity of consecutive gradients is normally distributed when training on IID data, but shows very strong correlations when training on a continuous video stream. **Right:** This is reflected in poor training performance. See the appendix for similar figures for Ego4D-stream.

human and animal experience, for example a popular task is learning one ImageNet class at a time and minimizing forgetting of previous classes [38].

But what happens if we attempt to learn continuously from a single video stream (Figure 1), meaning batch size 1 and high frame rate (e.g. 25 fps)? We do not see thousands of dogs per second, unlike the standard ImageNet setting, where images of even a same dog breed are quite uncorrelated at several scales – capturing a wide diversity of edges, textures, *etc.* Standard deep learning approaches lead to smooth optimization there, but it is very different from the setting of learning from a single video stream, where things happen slowly, at much longer timescales, with great correlation or even no change between frames for periods of time; see Figure 2. Do standard deep learning tools work in this setting? This is currently unknown territory, with very few attempts touching it at all [33] – and it is the focus of this paper.

We are interested in this problem because models should be able to adapt after deployment to their environment, hence receiving a single stream of information. This is needed for embodied intelligence (robotics), but also for any type of digital assistant that we would want to be able to adapt to a specific user’s needs.

A framework. We lay out a framework for learning from a single video stream, focusing on the entry-level problem of ... learning. Nevermind forgetting, can the models even learn in the first place? We study how successful learning is for different optimization settings, models, pretraining methods and tasks, when done from a single continuous stream. Our methodology follows the following principles:

- We consider future prediction tasks including pixels-to-pixels, pixels-to-segmentation and pixels-to-depth, all in RGB space (*i.e.* outputs are always 3 dimensional and with the same shape as the input frames). This way we do not need to change any parameter in the model between pretraining and single-stream learning, independently of what tasks are used in each. It enables using a simple L2 loss between pixel values in all cases.
- We evaluate performance in two ways - *in-stream*, and

out-of-stream. In-stream measures performance on the input video stream. For sufficiently difficult prediction tasks this should be enough. However, it conflates representation quality with adaptation to the scene, lighting condition, *etc.* For example if the lights in a scene are turned off for an hour, a model can learn to always predict black pixels – which is not ideal. To disentangle adaptation and generalization, we also evaluate out-of-stream, on a second stream on which the model does not learn. We propose to maximize both.

Contributions. With this methodology in place, we identified the following new results:

- On the optimization front, we observe that momentum, widely used in popular optimizers such as Adam, is unhelpful in highly correlated video streams. Instead methods without momentum such as RMSprop are more robust.
- Another result is a trade-off between adaptation and generalization induced by frequency of weight updates – slower leads to better generalization.
- We introduce a family of future prediction pretraining tasks, learnt in the standard independent and identically distributed (IID) setting, with large batches, and show that these lead to better single-stream performance compared to existing ImageNet pretraining tasks.
- We combine these results into a combo approach that we call *Baby Learning* (BL), and compare it to a standard deep learning setup (STDL) that uses Adam and ImageNet MAE pretraining. We show that BL on a sequential stream matches STDL on out-of-stream generalization when that same stream is permuted to be IID, while achieving better performance in-stream thanks to adaptation.

2. Related work

Online learning from a single video stream. One of the few papers dealing with this problem [33] proposed a minimum-redundancy “replay” buffer [27] to deal with temporal correlations. Replay buffers are not the sexiest research avenue for continual learning [41] and increase computation proportionally to the size of the buffer used. Additionally, in [33] performance on the video streams itself was never evaluated, only on unrelated image classification tasks. Test-time training is the next closest problem, such as presented in TTTVS [45] (also related [48]). TTVS’s motivation is to “improve **inference** quality” by leveraging self-supervised test-time optimization. Our motivation is to propose single-stream **learning**, which is reflected in much longer streams (24h in this paper, compared to seconds or few minutes). One challenge in single-stream learning is how to fully exploit available hardware parallelism in a batch size 1 setting [6, 23, 24].

Learning from a single video. Also relevant are experiments training ConvNets [21] or ViTs [8] on a single image [1] or a long video [43]. While outside of the streaming setting – using data shuffling, augmentation and large batches, these papers showed that learning can be quite successful in this setting (especially for the shallower layers), which was quite encouraging.

Continual learning. The continual learning literature (e.g. EWC [19]) tends to focus on learning a sequence of tasks, the goal being to learn new tasks as fast as possible without forgetting previous ones, such as in class incremental learning (e.g. iCarl [36]) and CLEAR [22]. Most relevant is on-line continual learning, where models learn from a stream of data, visited once. One challenge is model cheating by exploiting labels in the data [15], or by learning spurious features [46] that have limited generalisation power. Most of the work in this area has been on collections of images, rather than video.

Representation learning to the rescue. Recent continual learning papers reported that some of the challenges of the problem are partially mitigated by using features pretrained in IID settings [26, 35, 44]. Various pretraining strategies exist: self-distillation (e.g. BYOL [13], BRaVe [37], DINO [4]), contrastive (CLIP [34], VideoMoco [28], VITO [30]), and masked auto-encoding (e.g. VideoMAE [42], TubeViT [31], AudioVisualMAE [11], SiameseMAE [14], [2]). We investigate this aspect in our setup.

3. The Framework

Overview. The following is repeated throughout a video stream: a (potentially pretrained) model processes n frames (a *time step*) from an input video stream, and predicts frames for a future time step. Online (in-stream) performance metrics as well as an L2 loss function are then recorded by comparing predicted and target frames. Given the loss, gradients are computed and the model weights are updated by an optimizer. The target frames can come from the input stream itself or from an associated stream (e.g. for datasets having semantic segmentation or depth). In parallel with in-stream evaluation, we periodically assess performance on a held-out stream (composed of clips from the validation set of the dataset), to assess *generalization*.

3.1. Unified pixel-to-pixel modelling

We are interested in having a framework where switching between arbitrary tasks requires no changes to models or losses so we can abstract away decoder and loss function design (large research areas) and focus on the single-stream learning aspect. To achieve this we map all task target outputs to RGB space, so we can have a single pixel-to-pixel model and a single simple L2 pixel loss.

Models. We use 2 different popular backbones for our experiments: a UNet with self-attention in the bottleneck layer [17] and ViT-L [8]. To enable motion understanding, which is important for future prediction, we feed the models $n = 4$ consecutive frames stacked along the channel dimension. We also have the models predict the same number of frames, so 12 channels in total (4 frames times 3 channels, for RGB). For ViT we decode tokens to pixels using a channel to space transformation: each token gets mapped to an appropriate number of channels using a linear transformation, then gets reshaped into a patch (e.g. a 16x16 grid of 4x3 RGB channels). We provide details of the models in the Appendix, but it is worth noting that the UNet is considerably smaller at 8M parameters, compared to the 350M parameters of the ViT model. We always train the UNet from scratch, whereas we explore various pretraining schemes for ViT. To accommodate for the 12 input channels we use, we inflate [5] the first layer of the ViT (replicate the pretrained weights 4 times).

Memory. We did not explore explicit memory modules such as memory banks, LSTM cells, or long context in this paper, but we do think it should be very relevant going forward. In all cases, the model “sees” only a time step of frames and predicts another time step. Note however that the model weights are updated as it goes through the video stream – so both the weights and the optimizer (when stateful) provide some memory effects.

Replay buffer. We explored replay buffers [27], which keep a large cache of previous examples, then form batches by sampling from it. They have the disadvantage of increasing computational cost significantly over operating with batch size 1 (increases the theoretical computational cost of learning K times, where K is the batch size).

3.2. Video streams and tasks

We do not know of public datasets having very long video streams, for example days-long and beyond¹, so we create two different video streams: Ego4D-stream and ScanNet-stream, by concatenating videos from, respectively, Ego4D and ScanNet. We include Ego4D as it is large and has very long videos, and ScanNet because it has dense semantic segmentation and depth annotations, allowing us to experiment with different tasks.

Ego4D-stream. We concatenate the raw (un-trimmed) videos from the Ego4D dataset [12] to create a very long video stream. We use $\sim 90\%$ of the data to generate a training stream and $\sim 10\%$ for the validation stream. The videos in Ego4D were collected using a head or glass mounted camera, capturing activities of daily life. We use this stream for learning and evaluating future prediction in pixel space – no annotations are employed. See Table 1 for more details.

¹Other than Krishnacam [20], which is only 70 hours long.

Stream name	# videos train	# frames train	# videos val	# frames val	Max. length	Median length
Ego4D-stream	21,704	294M (3,265h)	2302	31M (348h)	1.95h	8.8 minutes
ScanNet-stream	1,199	1.8M (20h)	312	0.5M (5.7h)	5.5 minutes	1 minute

Table 1. The two streams we consider, formed out of Ego4D and ScanNet, together with properties of their original video clips.

ScanNet-stream. We use the videos from ScanNetV2 [7] to define 3 different prediction tasks, of pixels, semantic segmentation labels (40 classes) and depth. The availability of these additional target streams makes it possible for us to measure more explicitly higher-level understanding, which may not be clear from pixel prediction alone. The videos in this dataset were filmed with a camera navigating indoor scenes with one or more rooms. See Table 1 for more details.

Tasks. Overall we consider the following tasks across the two stream types: Ego4D-stream pixel prediction, ScanNet-stream pixel, semantic segmentation and depth prediction. The difficulty of most of these tasks can be controlled using a time displacement parameter Δ – how many steps in advance should the models predict. We consider 0, 1, and 4 time steps. A displacement of 0 only makes sense for semantic tasks as otherwise it corresponds to auto-encoding, which is easy. Note that this task is related to the near-future accuracy metric proposed in [15] to measure (forward transfer) adaptation in a more robust way.

RGB-fication. There is some hand design involved in mapping tasks into RGB space. For depth and semantic segmentation we use standard color maps typical for those tasks in ScanNet: the Viridis color map and the ScanNet label color map, respectively. We provide details of these mappings in the Appendix. For computing metrics other than the L2 pixel distance, we compute the nearest neighbor label / depth for each pixel color (e.g. a pixel’s color may not correspond exactly to one of the semantic segmentation labels, so we assume the predicted class is the label associated with the nearest color on the color map).

3.3. Evaluation

Learning a video stream task with a continuously updated model can be seen to decompose into: 1) discovering strong features that are general, for example learning that certain configuration of edges correspond to chair legs for semantic segmentation (**generalization**), which is likely to happen over long timescales. 2) by specializing to the particular scene being seen in the video, e.g. learning that the chair is the only red object in the scene and using redness to label the chair pixels (**adaptation**), which can happen over short timescales. Adaptation can be quite useful and lead to efficient visual mechanisms, but may not necessarily lead to generalization. Dummy models that only *e.g.* exploit temporal correlations in the target stream could misleadingly appear to perform very well [15]; see Table 4, *Blind* model.

We propose to evaluate both aspects by computing a score continuously **in-stream**, on the video stream task the model is learning from, to measure adaptation. Separately and periodically we compute the same score **out-of-stream**, on a held-out stream with the optimizer disabled (*out-of-stream*). We use the training sets of Ego4D and ScanNet to do in-stream evaluation, and the validation sets for out-of-stream evaluation.

Cumulative scores. Comparing single-stream learning models and approaches can be difficult if done naively. On the training stream, difficulty of the task may vary over time, for example certain parts of a stream may be more unpredictable due to camera motion or other factors, leading to natural oscillations. On the generalization side we care not just about performance at a particular moment in time, but also how fast it ramps up as training on the sequential video stream progresses. With this in mind, we use a global score over the whole stream – we do this by averaging the performance over 10,000 evenly-spaced points interpolated from the steps in both in-stream and out-of-stream settings.

Task-specific evaluation. We evaluate pixel prediction with average L2 pixelwise distance, the same that we use as loss for training. For semantic segmentation we use mean per-frame IoU and recall [3, 32] and depth using log relative mean square error (logRMSE) [9]. For both segmentation and depth we mask out pixels that were not annotated from the evaluation and loss.

4. Generalized Future Prediction

One of the most promising directions for continual learning, as identified in prior work, is representation learning. We are particularly interested in pixel-to-pixel approaches keeping consistent with the overall proposed framework where we never change neither the architecture nor the loss. We propose a family of pretraining methods that generalize future video prediction [40], and that follow a same pattern: given one input video clip the model is trained to predict a future clip from the same video (both clips are 4 frames long). We consider three variants, illustrated in Fig. 3:

- The easiest one, *Guided future prediction*, replaces a few patches from the input clip with patches in the same position from the future clip, hence narrowing down the range of possibilities.
- The intermediate one, *Vanilla future prediction* is just the standard task of predicting the future.
- *Masked future prediction* is a variation of Masked Auto-

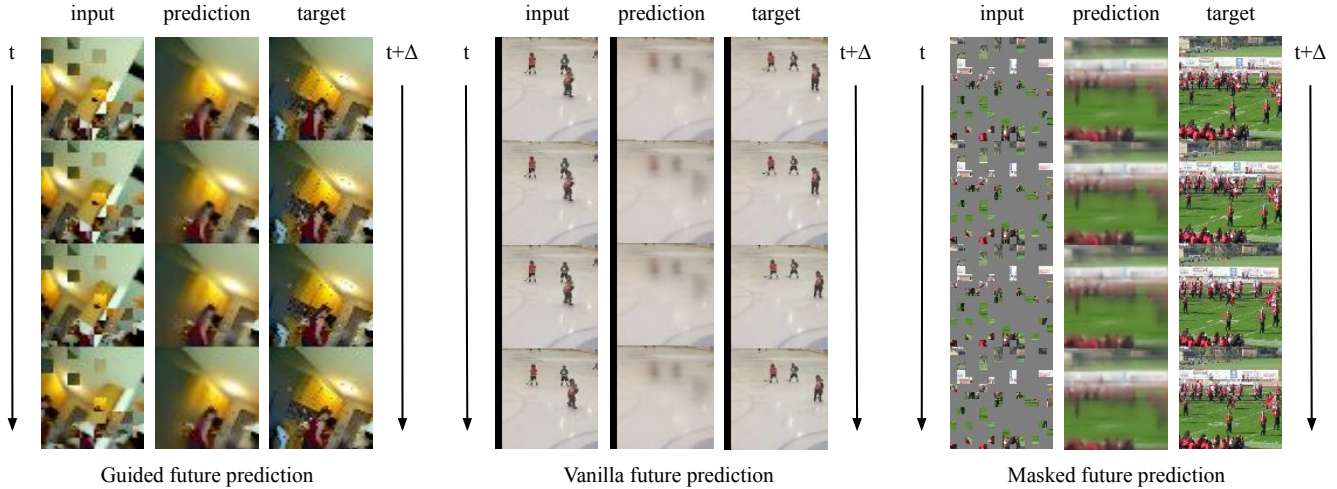


Figure 3. Video pretraining tasks we consider, sorted from easiest to hardest, left to right – guided future prediction, vanilla future prediction, and masked future prediction. Each column shows 4 consecutive frames vertically. For each method we show **left-to-right**: input frames, predictions from the model, target frames. We use a displacement (Δ) of 16 frames (0.64s) between input and target clips.

Encoding where the model must predict the future clip based on a partial view of the current clip.

Guided future prediction is related to Siamese MAE [2, 14, 47], but simpler, requiring a single forward pass over one model. Masked future prediction is related to video MAE-type approaches [10, 42] but with a strict separation between disjoint input and output clips, whereas in traditional video MAE there is a single clip that gets uniformly corrupted as part of denoising auto-encoding.

Hyperparameters. There are two hyperparameters: the fraction of guiding patches / masked patches and the shape of these patches (we use a constant square shape in all experiments). Vanilla future prediction is a special case of the other two in the case where there is no masked nor guiding patches. A general way to increase the difficulty of any of the variants is to increase the displacement (Δ) between the input and the target clips.

5. Results

Our goal is to achieve similar (or better) learning efficiency from a single continuous video stream as we expect from the standard deep learning setting – using sequences of batches of well shuffled examples. Here is a summary of the main results of our experiments on learning from a continuous video stream:

- Momentum, widely used in optimizers such as Adam, hurts performance in single-stream learning.
- Less frequent weight updates (e.g. every 2.5 seconds), helps generalization while sacrificing some adaptation.
- Pretraining the models on IID data before single-stream learning is quite impactful. While popular ImageNet-based pretraining helps, we found future-prediction based video pretraining to be vastly superior.

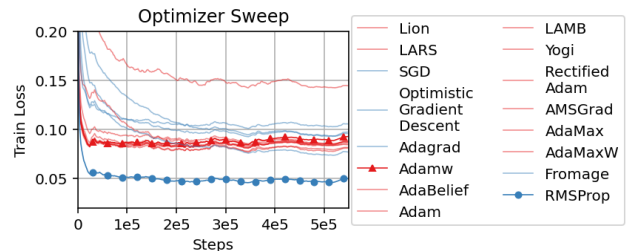


Figure 4. A sweep over commonly used optimizers. Those without momentum are shown in blue and aid the models adaptability considerably compared to the more commonly used Adam variants, which are shown in red.

We ran all in-stream learning for 24 hours of video, and measured out-of-stream performance from regularly saved checkpoints on 3h30 of video. Resolution was always 224x224.

5.1. Optimization

Starting from the standard optimizer used in nearly all deep learning setups, Adam, we observed that it does not train well with default parameters in the continuous video stream setting compared to the IID setting; see Fig. 2, right. To understand the differences between the two settings, we analyzed the temporal correlation between consecutive gradients. We found that the norm and variance of the gradients do not reveal strong differences. However, their orientation reveals strong correlations between pairs of consecutive gradients in the continuous case; see Fig. 2, left, where we plot the distribution of the cosine similarity between consecutive gradients in the continuous vs IID case.

To investigate the optimization further we performed a large sweep over commonly used optimizers, shown in Fig. 4. Results are averaged over 8 settings: 2 tasks

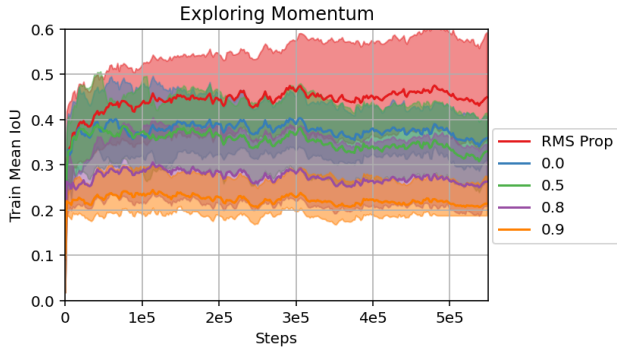


Figure 5. Reducing momentum with the AdamW optimizer helps to recover some of the performance of RMSProp.

(Ego4D-stream and ScanNet-stream segmentation prediction), 2 models (ViT and UNet) and 2 displacements (1 step and 4 steps). RMS Prop significantly outperformed the more commonly used Adam variants.

Momentum hurts. To scrutinize the difference between AdamW and RMSProp, we looked at the impact of *momentum*. We found that lowering the momentum of the AdamW optimizer recovered some of the performance of RMSProp (as shown in Fig. 5). One possible explanation is that momentum exacerbates the problem of correlated consecutive gradients (that differ from the underlying gradient of the loss function over the whole stream) and makes the weights accelerate too much in the wrong direction. We stick with RMSProp for the rest of the paper, as it is more memory efficient due to not using the average of past gradients.

Infrequent weight updates help generalization, hurt adaptation. We also observed an interesting effect associated to the frequency of weight updates: doing it less frequently tends to benefit generalization at some cost to adaptation (since the model cannot adapt as frequently or as strongly to any particular time step). We found that updating weights every 16 frames (or 0.64s) provided a decent trade-off between adaptation and generalization across models, tasks and datasets as shown in Tab. 2

Constant learning rate helps adaptation. We tested a range of common learning rate schedules including constant, linear, cosine and exponential decay, “1cycle” and cosine decay with restarts. The main finding was that decaying learning rates over the course of training, in particular cosine decay with exponent 2.0, helps generalization but significantly hurts adaptation as shown in Fig. 7. We therefore used a constant learning rate, with a linear warmup period of 1k steps, in our experiments.

Replay buffer batches can be small. We focus most of our exploration on the case without replay buffers, but did check what happens when we add a replay buffer to our very best setting - which slows the experiment down proportionally

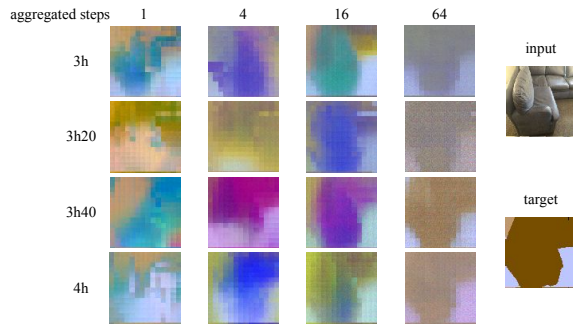


Figure 6. ScanNet-stream segmentation task *generalization* results on the exact same out-of-stream clip after different number of hours of exposure to a same stream for 4 model runs. The models differ only by the length of the interval between weight updates / number of steps of gradient aggregation (horizontal axis). Models with less frequent updates tend to generalize better (rightmost column), whereas models with more frequent updates tend to have strong priors about which objects are currently in the scene, leading to hallucinations (leftmost 3 columns in this case).

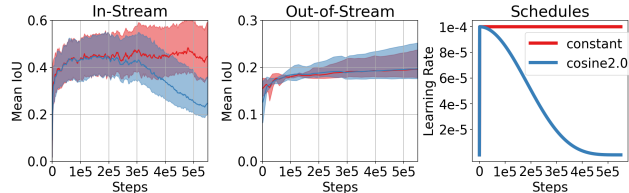


Figure 7. In-stream and out-of-stream results for a learning rate schedule using cosine decay with exponent 2.0 vs a constant learning rate (both with a linear warmup phase of 1k steps). The decaying learning rate is able to generalise more quickly but at a significant cost to adaptation.

to the batch size employed. Over 11 hours of wall-clock training on the ScanNet-stream segmentation task with prediction displacement of 1 time step, an experiment with a replay buffer containing 10,000 samples (circular writing, random reading) and batch size 4 reached mean IoU only 2% superior to a model trained without replay buffer. Batch size 16 did not improve over batch size 4 and we did not use replay buffers in any other experiments.

5.2. Pretraining

For pretraining we did not learn from a single continuous stream, but instead formed big well-shuffled batches of 1024 clips using 8x8 slices of TPU-v5-lite, used AdamW and updated weights every step, for 150k steps. We pre-trained on the Kinetics-700-2020 dataset [39], which is composed of 10s clips. We found that initializing with the ImageNet-MAE checkpoint led to much quicker optimization, so used that in all cases. More details can be found in the Appendix.

We pretrained ViT-L models on the future prediction tasks from sec. 4 and monitored their representation learn-

Stream	dataset	model	n steps per update			
			1	4	16	64
In	Ego4D (↓)	UNet	.036	.038	.037	.039
		ViT	.035	.037	.039	.047
	Segm (↑)	UNet	.420	.292	.211	.195
		ViT	.457	.395	.302	.232
Out-of	Ego4D (↓)	UNet	.095	.051	.047	.042
		ViT	.076	.062	.046	.044
	Segm (↑)	UNet	.176	.179	.205	.183
		ViT	.251	.272	.280	.274

Table 2. Results on video streams from two datasets, two models, different numbers of steps per gradient update (horizontal), and evaluated in-stream and out-of-stream, showing the impact of accumulating gradients over multiple steps. Results are averaged over 1 and 4 displacement steps. For Ego4D-stream we report average L2 pixelwise distance (lower is better), for ScanNet-stream Segm we report mean per-frame IoU (higher is better). Bold marks the best result for each model-stream pair.

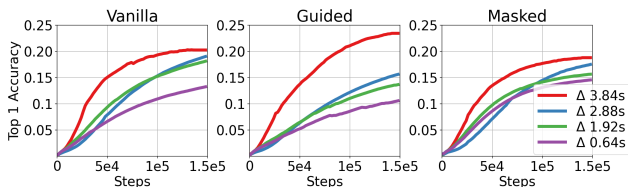


Figure 8. Linear top-1 accuracy in Kinetics during pretraining for different forms of future prediction tasks: Vanilla, Guided, Masked, and different displacements. The longer the displacement the better the accuracy is. Note that this is an online evaluation where the model only sees 4 frames (that for "Masked" and "Guided" are distorted) hence performance is far below models evaluated on the whole 10s of video (250 frames), often with plenty of test time augmentation, but the relative performance between the various curves is informative.

ing performance during training by adding a linear head to the model just before the decoder (we use a stack of 4 self-attention layers on top of the ViT-L encoder). For this monitoring, we employed a standard cross-entropy loss between logits and ground truth labels in conjunction with a stop-gradient so the supervision would not influence the backbone weights.

We experimented with different input-target clip temporal displacements and observed the interesting effect that the longer the displacement the better the classification performance is as shown in Fig. 8, even for masked future prediction, which reduces to the popular Masked Auto-encoding [16] when displacement is 0. Our experiments suggest that 0 may be a sub-optimal choice. For the longest displacement the best method for top-1 accuracy emerged as Guided Future Prediction. The top result shown in the plot also used fewer guiding patches, 5% instead of 10% – we used this best model for the rest of our evaluations. We show example video predictions for the various models online at <https://sites.google.com/view/one-stream-video>.

Table 3 has results on in/out-of-stream evaluation, showing large benefits over ImageNet classification pretraining and using no pretraining (training from scratch). ImageNet-MAE also does considerably better than classification-based pretraining. The same ViT-L architecture was used in all cases.

5.3. IID vs Continuous Learning

We compare our best setup, which we call, for no particularly technical reason, "Baby Learning" (BL), to a standard deep learning setup (STDL). STDL uses AdamW with standard parameters (learning rate $1e-4$, momentum $b1$ as 0.9), with weight updates after each batch, the same ViT-L model but with the popular ImageNet MAE checkpoint. We implement the IID setting by sampling a sequence of random time steps (and associated target time steps) from random videos of the same base dataset. Note that while most approaches employ large batches, SGD with mini-batch size 1 is well known to work well [25] and is even said to generalize better than using large batches. We match the number of frames that each method sees – namely for batch size > 1 , the total number of learning steps is reduced proportionally.

Dummy baseline. For reference, we also include a dummy blind baseline, proposed in previous online learning works [15], which exploits the temporal correlations in the target stream without seeing the input frames at all: this "model" outputs the mean of the previously seen pixels at the same locations for depth and pixel, and the most frequent colors for segmentation, for the previous target. It produces random performance for out-of-stream evaluation where consecutive targets are unrelated².

Results on all tasks are shown in Table 4. We show visual results of a subset of these tasks in Fig. 9. It is visible that our approach BL matches STDL with batch size 1 out-of-stream while outperforming it in-stream. In fig. 11 in the Appendix we also report results for STDL on a continuous stream, which does not work at all, and for BL on an IID stream which does best – this shows that there are still many more improvements possible for learning from continuous streams.

6. Conclusion

We have presented a different perspective on continual learning by defining prediction tasks on a single very long stream of video and by proposing evaluations that measure both adaptation and generalization. We show that there is a permanent tension between the two aspects, but that it is possible to improve both by specializing the optimizer (lowering momentum, increasing the number of gradient

²Another baseline that may be of interest is one that copies the previous input frames instead of target frames – this could do well for both train and eval for pixels, but poorly for segmentation and depth

Pretraining Checkpoint	Ego4D (\downarrow)	ScanNet Depth (\downarrow)	ScanNet Segm (\uparrow)	ScanNet (\downarrow)
None	.074 / .105	1.969 / 2.163	.177 / .188	.083 / .083
ViT-L-I1K-CLS	.043 / .048	1.821 / 2.040	.288 / .234	.040 / .042
ViT-L-I21K-CLS	.042 / .048	1.735 / 2.013	.244 / .192	.039 / .040
ViT-L-I1K-MAE	.040 / .044	1.806 / 2.045	.360 / .320	.037 / .038
Guided Future Prediction	.036 / .043	1.622 / 1.990	.390 / .313	.032 / .034

Table 3. Table comparing performance of models pretrained on ImageNet-1K and 21K using MAE or classification, and models pretrained on our video tasks in Kinetics. The results are averaged over two sets of experiments, with displacements of 1 and 4 time steps and are shown in the format in-stream / out-of-stream.

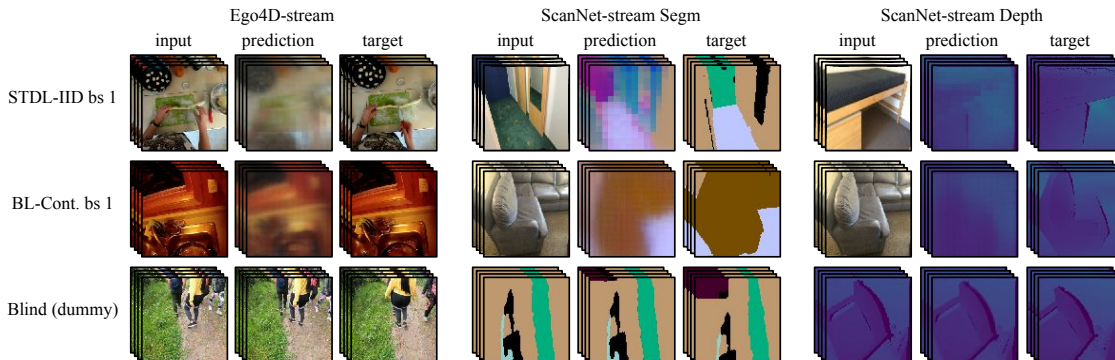


Figure 9. Future prediction results for video streams from Ego4D-stream, ScanNet-stream Segm, and ScanNet-stream Depth.

displacement (Δ)	Ego4D (\downarrow)		ScanNet (\downarrow)		ScanNet Segm (\uparrow)		ScanNet Depth (\downarrow)	
	t+1	t+4	t+1	t+4	t+1	t+4	t+1	t+4
STDL (IID) bs 16	.023/.019	.060/.055	.014/.013	.042/.061	.495/.398	.382/.295	1.798/2.01	1.838/2.038
STDL (IID) bs 1	.019/ .018	.057/ .056	.010 /.013	.051 /.060	.376/.302	.276/.227	1.722/ 2.012	1.759/ 2.034
BL (Cont.) bs 1	.018 /.021	.055 /.066	.011/ .012	.055/.061	.463 / .312	.328 /.241	1.595 /2.038	1.655 /2.097
Blind (dummy)	.038 / -	.086 / -	.032 / -	.109 / -	.547 / -	.307 / -	1.256 / -	1.649 / -

Table 4. Future prediction results for video streams from various datasets for two different temporal displacements (horizontal). We show results for the standard deep learning approach (STDL) on IID data with batch size 16 and 1, and for our approach (BL) when using a continuous video stream (Cont.). Two numbers are reported for every cell, corresponding to in-stream / out-of-stream performance. We highlight which of the two batch size 1 approaches performs best for each number. The Blind (dummy) model exploits correlations in the target stream which is available in-stream and used for learning, hence performs very well in-stream, but has random performance in evaluation where the target stream is not available. This shows the necessity to measure both adaptation and generalization performance.

aggregation steps between weight updates) and by pretraining the model appropriately. Conversely, we show that just using Adam with default parameters performs poorly on single-stream learning, due to the large correlations in the data. This makes the proposed setting different from popular ImageNet-based continual learning setups where batches are employed, and the data is significantly uncorrelated - for example in class-by-class learning, seeing different web images of dogs will not break Adam the way consecutive frames in a video does.

We admit that the motivation for the research direction laid out in this paper may not be immediately crystal clear to everyone, particularly in the context of the current research landscape, that focuses on fitting larger and larger models to the whole internet. We are motivated by a possible future where people have their models in physical devices they carry around and train them via natural interac-

tion - by showing them the world from their perspective - and teach them only the things that they believe the models should be taught, similar to the way we teach children. It is likely that such models, by not having to know everything, can be smaller, more efficient, have fewer internet biases and be more privacy-friendly.

Acknowledgments. We would like to thank Ross Goroshin and Ali Eslami for reviewing the paper and providing helpful comments, and Razvan Pascanu and Simon Osindero for insightful advice and interesting pointers into the literature and background work.

References

- [1] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. A critical analysis of self-supervision, or what we can learn

- from a single image. In *International Conference on Learning Representations*, 2020. 3
- [2] Daniel Bear, Kevin T. Feigels, Honglin Chen, Wanhee Lee, Rahul Venkatesh, Klemen Kotar, Alex Durango, and Daniel L. K. Yamins. Unifying (machine) vision via counterfactual world modeling. *ArXiv*, abs/2306.01828, 2023. 3, 5
- [3] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. *arXiv*, 2019. 4
- [4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV 2021 - International Conference on Computer Vision*, pages 1–21, Virtual, France, 2021. 3
- [5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 3
- [6] Joao Carreira, Viorica Patraucean, Laurent Mazare, Andrew Zisserman, and Simon Osindero. Massively parallel video networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 649–666, 2018. 2
- [7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017. 4, 1
- [8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3, 1
- [9] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014. 4
- [10] Christoph Feichtenhofer, Yanghao Li, Kaiming He, et al. Masked autoencoders as spatiotemporal learners. *Advances in neural information processing systems*, 35:35946–35958, 2022. 5
- [11] Mariana-Iuliana Georgescu, Eduardo Fonseca, Radu Tudor Ionescu, Mario Lucic, Cordelia Schmid, and Anurag Arnab. Audiovisual masked autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16144–16154, 2023. 3
- [12] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Christian Fuegen, Abrahm Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Mery Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the World in 3,000 Hours of Egocentric Video. In *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*, 2022. 3
- [13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2020. Curran Associates Inc. 3
- [14] Agrim Gupta, Jiajun Wu, Jia Deng, and Li Fei-Fei. Siamese masked autoencoders, 2023. 3, 5
- [15] Hasan Abed Al Kader Hammoud, Ameya Prabhu, Ser-Nam Lim, Philip H. S. Torr, Adel Bibi, and Bernard Ghanem. Rapid adaptation in online continual learning: Are we evaluating it right? In *2023 International Conference on Computer Vision (ICCV)*, 2023. 3, 4, 7
- [16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 7
- [17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020. 3, 1
- [18] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. 1
- [19] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521 – 3526, 2016. 3, 2
- [20] Alexei A. Efros Krishna Kumar Singh, Kayvon Fatahalian. Krishnacam: Using a longitudinal, single-person, egocentric dataset for scene understanding tasks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016. 3
- [21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recog-

- dition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3
- [22] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. 3
- [23] Mateusz Malinowski, Grzegorz Swirszcz, Joao Carreira, and Viorica Patraucean. Sideways: Depth-parallel training of video models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11834–11843, 2020. 2
- [24] Mateusz Malinowski, Dimitrios Vytiniotis, Grzegorz Swirszcz, Viorica Patraucean, and Joao Carreira. Gradient forward-propagation for large-scale temporal video modelling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9249–9259, 2021. 2
- [25] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018. 7
- [26] 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. 3
- [27] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015. 2, 3
- [28] Tian Pan, Yibing Song, Tianyu Yang, Wenhao Jiang, and Wei Liu. Videomoco: Contrastive video representation learning with temporally adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11205–11214, 2021. 3
- [29] pandas development team. pandas.dataframe.ewm. <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.ewm.html>, Accessed: 2023-11-24. Online documentation. 1
- [30] Nikhil Parthasarathy, SM Ali Eslami, Joao Carreira, and Olivier J Henaff. Self-supervised video pretraining yields robust and more human-aligned visual representations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3
- [31] A. J. Piergiovanni, Weicheng Kuo, and Anelia Angelova. Rethinking video vits: Sparse video tubes for joint image and video learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 2214–2224. IEEE, 2023. 3
- [32] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017. 4
- [33] Senthil Purushwalkam, Pedro Morgado, and Abhinav Gupta. The challenges of continuous self-supervised learning. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, page 702–721, Berlin, Heidelberg, 2022. Springer-Verlag. 2
- [34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 3
- [35] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2022. 3
- [36] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society, 2017. 3
- [37] Adria Recasens, Pauline Luc, Jean-Baptiste Alayrac, Luyu Wang, Florian Strub, Corentin Tallec, Mateusz Malinowski, Viorica Patraucean, Florent Altché, Michal Valko, Jean-Bastien Grill, Aaron Oord, and Andrew Zisserman. Broaden your views for self-supervised video learning. pages 1235–1245, 2021. 3
- [38] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2
- [39] Lucas Smaira, João Carreira, Eric Noland, Ellen Clancy, Amy Wu, and Andrew Zisserman. A short note on the kinetics-700-2020 human action dataset. *arXiv preprint arXiv:2010.10864*, 2020. 6
- [40] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852. PMLR, 2015. 4
- [41] Richard Sutton. Maintaining plasticity in deep continual learning, 2022. Accessed 11 16, 2023. 2
- [42] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*, 2022. 3, 5
- [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. *arXiv:2310.08584*, 2023. 3
- [44] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application, 2023. 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. 2
- [46] Yujie Wei, Jiaxin Ye, Zhizhong Huang, Junping Zhang, and Hongming Shan. Online prototype learning for online continual learning. In *ICCV*, 2023. 3

- [47] Philippe Weinzaepfel, Vincent Leroy, Thomas Lucas, Romain Brégier, Yohann Cabon, Vaibhav Arora, Leonid Antsfeld, Boris Chidlovskii, Gabriela Csurka, and Jérôme Revaud. Croco: Self-supervised pre-training for 3d vision tasks by cross-view completion. *Advances in Neural Information Processing Systems*, 35:3502–3516, 2022. [5](#)
- [48] Jay Zhangjie Wu, David Junhao Zhang, Wynne Hsu, Mengmi Zhang, and Mike Zheng Shou. Label-efficient online continual object detection in streaming video, 2023. [2](#)

Learning from One Continuous Video Stream

Supplementary Material

7. Overview

This section provides additional results that we could not fit in the main paper, more details about the experiments, and extra visualizations. It also discusses two negative results we had along the way. See <https://sites.google.com/view/one-stream-video> for videos of future prediction pretraining and of ScanNet-stream training with the best models from table 4, in sequential and IID cases.

7.1. Additional results

0 displacement results. We show results for predicting segmentation and depth with no offset in time (the usual semantic segmentation and depth estimation tasks), comparing the strong *standard deep learning* (STDL) approach to our approach (*baby learning*, BL) on continuous streams and IID data (Fig. 10). As expected, the standard deep learning approach performs very poorly on continuous streams. Our approach improves the performance in this setting and obtains outstanding performance in the IID setting, showing the effectiveness of pre-training via guided future prediction in both settings. Unless specified otherwise, plots are shown with exponential smoothing, with $\alpha = 1e - 3$, using Pandas [29].

Performance over time in video. Fig. 11 shows the average in-stream performance over 1 training run on sequential data against the time spent in the video. This shows that the model performance improves rapidly over the first minute of each video it sees.

Disaggregated results. In the main body of the paper, many of our results are aggregated over multiple models, tasks and displacements so we can be more confident about their generality. Here we include disaggregated results for reference. Fig. 17 and Fig. 18 show the in-stream and out-of-stream evaluation for the 2 different learning rate schedules experimented with in Fig. 7 separately for different datasets, models, and displacements (these were aggregated over in the main paper). Fig. 19 shows the training loss for the different optimizers experimented with in Fig. 4 separately for different datasets, models, and displacements. Fig. 20 and Fig. 21 show the in-stream and out-of-stream evaluation for RMS Prop and AdamW with different levels of momentum as in Fig. 5, but separately across datasets, models, and displacements.

7.2. Experimental details

Pretraining. We used the AdamW optimizer with learning rate $2e-4$, weight decay of 0.05 and batch size 1024,

updating the weights every training step. We did 1000 steps of linear warmup and afterwards kept the learning rate constant for 150k steps. We experimented with different temporal displacements for prediction, from 0.64s to 3.84s observing continuous improvement for all 3 objectives (vanilla, guided and masked future prediction). Guidance and masking are both implemented by splitting the input frames into a regular grid of non-overlapping patches and replacing a fixed percentage of them, respectively by patches from the future or by gray. We replaced 5% and 10% of the patches for guiding (5% did better for longer displacements and overall) and 50% or 75% of the patches for masking (50% did better for longer displacements and overall). 32×32 patches did best in both cases, compared to 16×16 .

Models. Our ViT-L experiments used a standard model as described in [8] except that we inflated (replicated 4 times) the first layer when starting from ImageNet-pretrained checkpoints, to be able to deal with the inputs being 4 frames stacked along the channel dimension (unlike in training where ImageNet images had only 3 channels).

Our UNet experiments use a variant of the model described in [17] processing at 4 resolutions, starting with 224×224 . As in that paper, we use group norm throughout. At each resolution we use 8 residual blocks and 64 channels. At the lowest resolution, 28×28 , we apply a self attention block with 4 heads.

Colormaps. Our models output predictions for all tasks in RGB space. This is straightforward for the case of future frame prediction, but for some tasks the outputs are originally defined in a different space. For example, for ScanNet semantic segmentation, outputs are traditionally represented by 40d one hot vectors. We map such spaces to RGB using the typical colormaps used by the datasets in their publications. For segmentation, we use the NYU40 colormap, which was used for visualizing the classes in ScanNet [7]. To evaluate, because predicted RGB values may not match exactly those corresponding to one label, we find the nearest neighbor color in the colormap, based on L2 distance and assign that pixel the corresponding label. For the depth task, we use a Viridis colormap from Matplotlib [18] which is a perceptually uniform sequential colormap with 256 colors and is commonly used to visualize depth in papers. We proceed similarly to the segmentation mapping to obtain the depth value from the rgb prediction. We assume a maximum depth of 8 and scale it to 1 when colormapping, then invert this normalization to find the predicted depth value. The colormaps can be found in Figure 12.

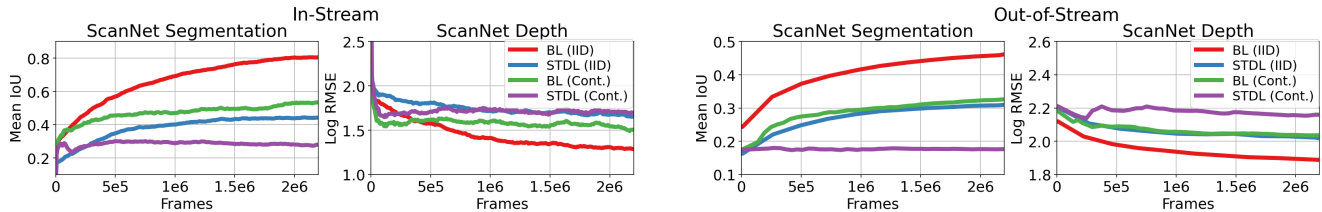


Figure 10. In-stream and out-of-stream performance on segmentation and depth estimation (with no offset in time) comparing a strong standard deep learning (STDL) approach, which uses an ImageNet-MAE checkpoint with batch size 1 and our approach (BL) when using a continuous video stream (Cont.) or IID data. STDL fails completely when fed a continuous stream. Note that here, pretraining from MAE is more closely aligned with the task than the video prediction pretraining objective we use, since displacement is 0 – segmentation and depth align perfectly with the input frames. Yet, results suggest the video prediction objective leads to significant improvements. These results also suggest that there is a need for new continual learning techniques for bridging the gap between learning from IID data and a sequential stream (this becomes apparent when using the same strong pretrained model).

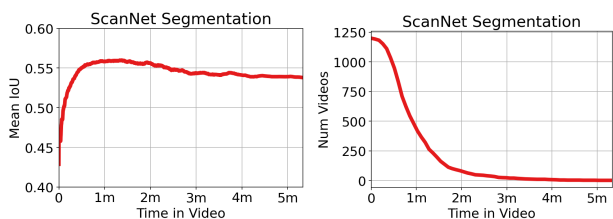


Figure 11. **Left:** In-stream Mean IoU plotted against time through the video averaged over all videos in 1 training run on ScanNet-stream segmentation. **Right:** The number of videos that ScanNet-stream is composed of in training, with at least n minutes.



Figure 12. Colormaps that are used for mapping semantic labels and depth to RGB. Viridis is used for depth estimation task and NYU40 is used for segmentation.

7.3. Additional visualizations

Gradient correlations. As showed in the main paper for the semantic segmentation task, Fig. 2, the cosine similarity between consecutive gradients shows very strong correlations when training from a continuous video stream as opposed to training with IID data. We observed the same trend when training on Ego4D for the task of future prediction, although in this case the loss seems less impacted by the correlations; see Fig. 13.

In-stream / out-of-stream performance plots for table 4. Fig 14 and Fig. 15 show the in-stream and out-of-stream evaluation curves corresponding to the results in Table 4 (Fig. 16 for version without any smoothing).

7.4. Negative results.

Other things we tried but were inconclusive or did not help:

- Elastic weight consolidation [19] is a well known contin-

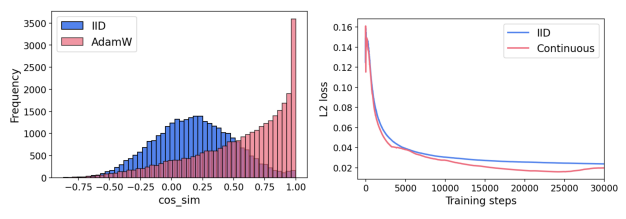


Figure 13. UNet training on Ego4D-stream for the future prediction task. **Left:** The cosine similarity of consecutive gradients is close to a normal distribution when training on IID data, but shows very strong correlations when training on a continuous video stream. **Right:** L2 loss.

ual learning technique that can be used in an online setting by creating a new copy of the model weights periodically, then penalizing certain departures away from these anchor weights afterwards. We implemented the simple L2 version of the method also described in the original paper. While this helped for methods like Adam with default momentum, it did not provide additional benefit once we moved to RMSprop.

- Augmentation – random crops + flips per step. We experimented with data augmentation in an attempt to reduce deviation from IID, but did not observe advantages over a consistent augmentation for the whole video within a video stream. This was surprising and perhaps could be achieved by using more aggressive augmentation. We experimented with this only for semantic segmentation with time displacement of 0 (present prediction), as doing this for future prediction would require feeding in augmentation parameters to the model (otherwise the model could not possibly predict the target pixels, since the augmentation is random).

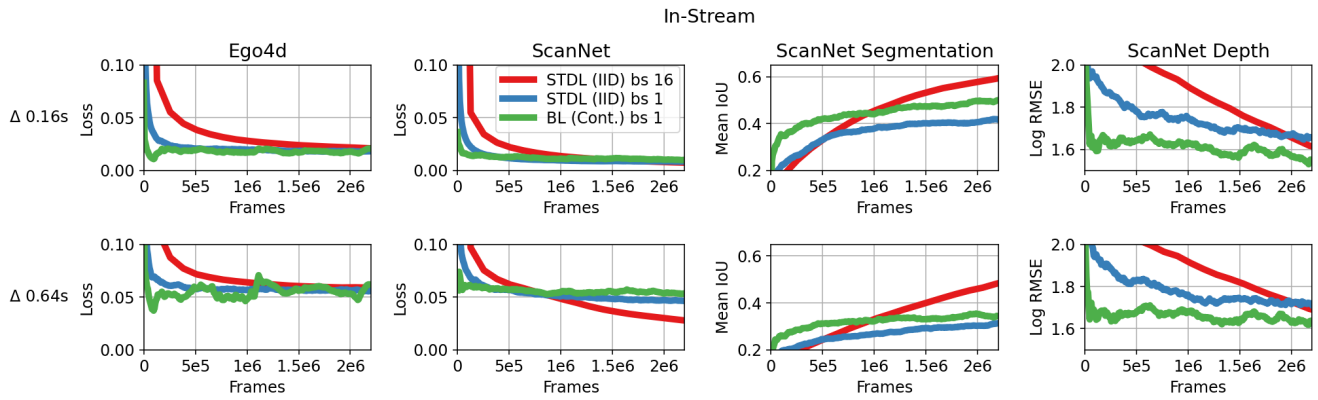


Figure 14. In-stream performance of a strong standard deep learning (STDL) approach with batch size 1 or 16 on IID data, and our approach (BL) when using a continuous video stream (Cont.).

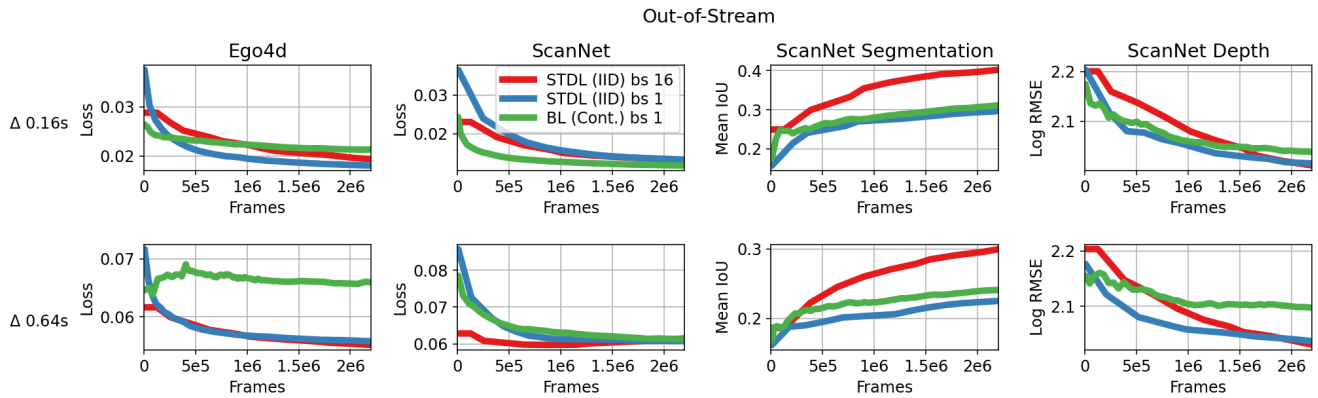


Figure 15. Out-of-stream performance of a strong standard deep learning (STDL) approach with batch size 1 or 16 on IID data, and our approach (BL) when using a continuous video stream (Cont.).

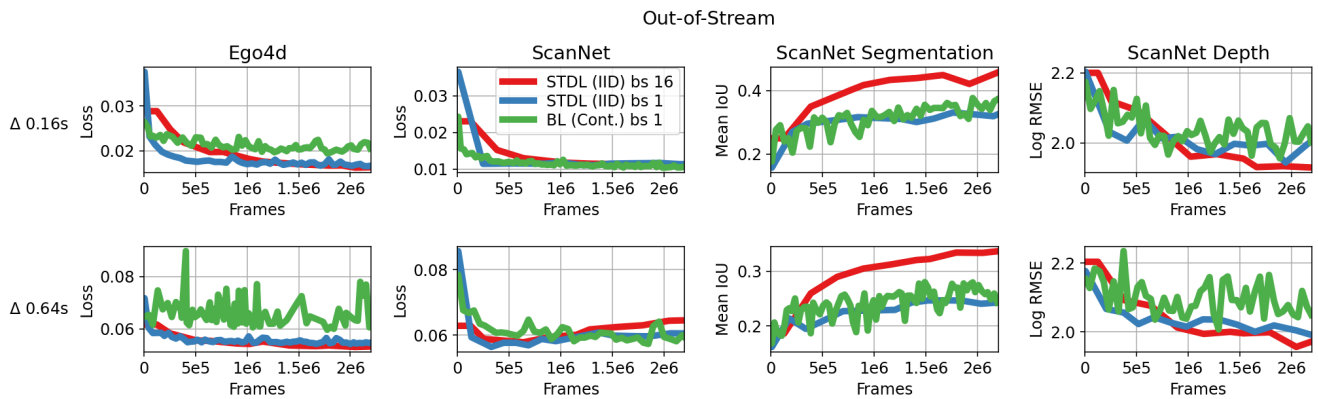


Figure 16. Same as Fig. 15 but with no smoothing.

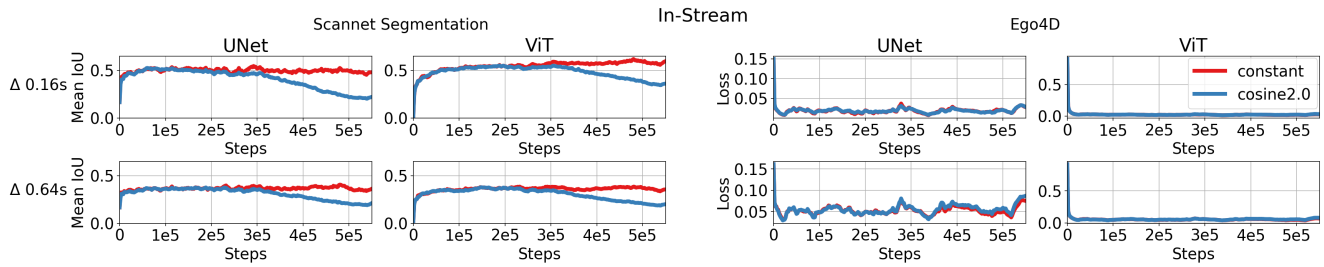


Figure 17. In-stream performance of 2 different learning rate schedules across 2 datasets, 2 displacements and 2 models. Our approach BL on a continuous stream is generally better than baseline STDL on an IID stream, for same batch size 1.

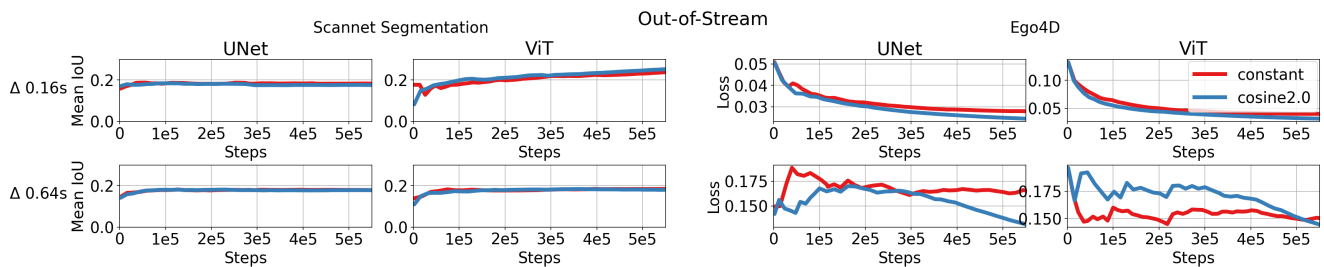


Figure 18. Out-of-stream performance of 2 different learning rate schedules across 2 datasets, 2 displacements and 2 models. Our approach BL on a continuous stream is competitive with STDL on an IID stream for the same batch size 1. It does fail badly for Ego4d with large displacement, but is better on ScanNet segmentation.

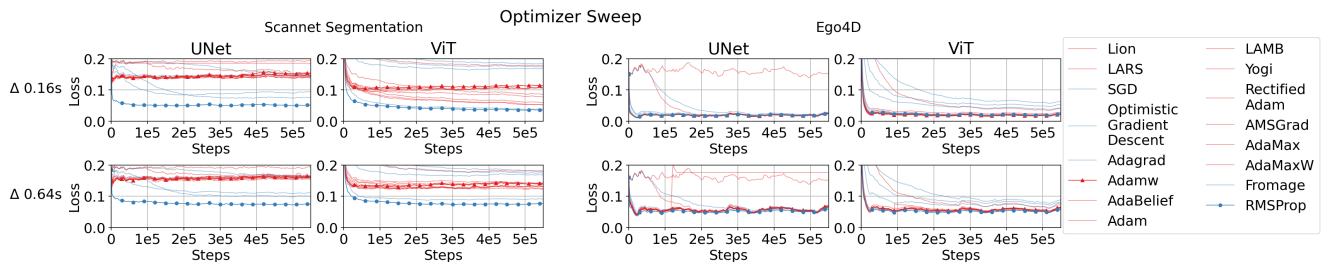


Figure 19. Training loss of different optimizers across 2 datasets, 2 displacements and 2 models.

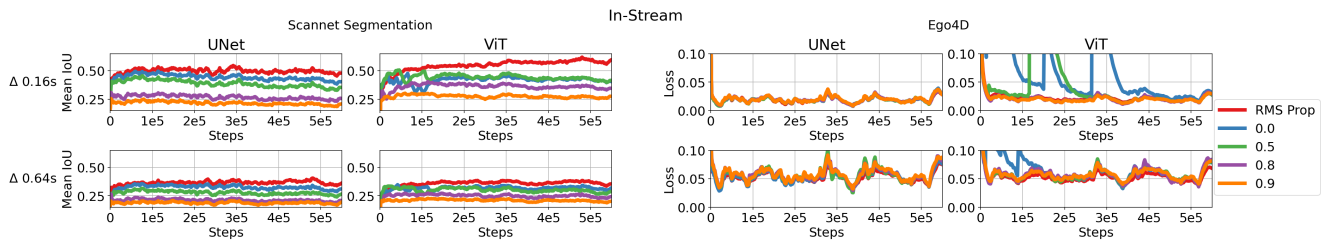


Figure 20. In-stream performance of RMSProp and AdamW with different levels of momentum across 2 datasets, 2 displacements and 2 models.

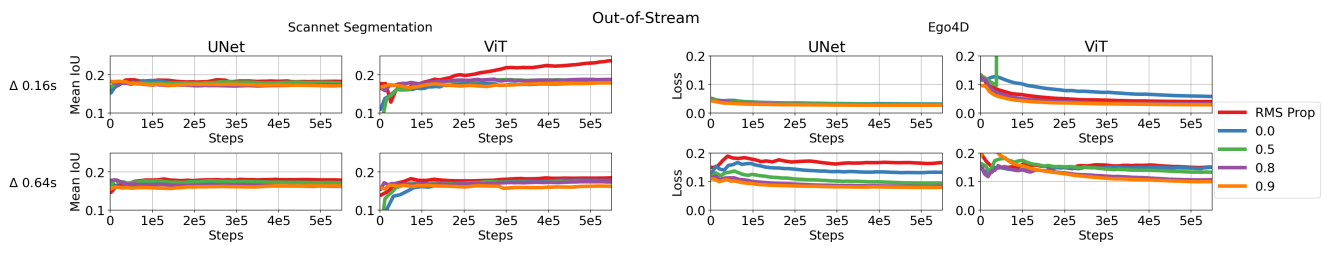


Figure 21. Out-of-stream performance of RMSProp and AdamW with different levels of momentum across 2 datasets, 2 displacements and 2 models.