MEMORY STORYBOARD: LEVERAGING TEMPORAL SEGMENTATION FOR STREAMING SELF-SUPERVISED LEARNING FROM EGOCENTRIC VIDEOS

Yanlai Yang
New York University
yy2694@nyu.edu

Mengye Ren
New York University
mengye@nyu.edu

ABSTRACT

Self-supervised learning holds the promise of learning good representations from real-world continuous uncurated data streams. However, most existing works in visual self-supervised learning focus on static images or artificial data streams. Towards exploring a more realistic learning substrate, we investigate streaming self-supervised learning from long-form real-world egocentric video streams. Inspired by the event segmentation mechanism in human perception and memory, we propose "Memory Storyboard," a novel continual self-supervised learning framework that groups recent past frames into temporal segments for a more effective summarization of the past visual streams for memory replay. To accommodate efficient temporal segmentation, we propose a two-tier memory hierarchy: the recent past is stored in a short-term memory, where the storyboard temporal segments are produced and then transferred to a long-term memory. Experiments on two real-world egocentric video datasets show that contrastive learning objectives on top of storyboard frames result in semantically meaningful representations that outperform those produced by state-of-the-art unsupervised continual learning methods.

1 INTRODUCTION

Humans are capable of learning continuously from a stream of unlabeled and uncurated perceptual inputs, such as video data, without needing to iterate through multiple exposures or epochs. Since early infancy, humans have accumulated knowledge about the world through a continuous flow of raw visual observations. This capability contrasts sharply with the training paradigm of current methods in self-supervised learning (SSL) (Chen et al., 2020; Grill et al., 2020; Chen & He, 2021; Caron et al., 2020; Bardes et al., 2022; He et al., 2022; Assran et al., 2023; He et al., 2020). Despite making significant strides in learning from large unlabeled datasets, these approaches still predominantly rely on static and curated image datasets, such as ImageNet (Deng et al., 2009), and require multiple epochs of training for effective learning. This difference in paradigm raises a compelling question: how can we learn good visual representations in a streaming setting—learning from visual inputs in their original temporal order without cycling back?

Motivated by the differences in mechanisms between human learning and standard SSL, we aim to build learning algorithms that can efficiently learn visual representations and concepts from streaming video. One especially relevant mechanism in the human brain is event segmentation (Newtson et al., 1977; Zacks et al., 2001; Yates et al., 2022), where we spontaneously segment visual streams into hierarchically structured events and identify the event boundaries. Take your recent vacation trip as an example—you probably remember separate events and activities like exploring a city, dining at a local restaurant, or relaxing at the beach. The event segmentation mechanism helps us organize memories, recall specific moments, and summarize lengthened experiences (Zacks et al., 2006; Zacks & Swallow, 2007).

Drawing inspiration from the way we organize our memory in the brain, we introduce *Memory Storyboard*, a novel approach for streaming self-supervised learning. Memory Storyboard features a temporal segmentation module, which groups video frames into semantically meaningful temporal segments, resembling the automatic event segmentation of human cognition. Through our temporal contrastive learning objective, these temporal segments effectively facilitate representation learning in streaming videos. To accommodate efficient temporal segmentation, we propose a two-tier hierarchical memory: temporal segmentation in the short-term memory is used to update the temporal class labels in the long-term memory, and a training batch consists of samples mixed from both memories. A high-level diagram of the algorithm is shown in Figure 1.

We conduct experiments on the SAYCam (Sullivan et al., 2021) and KrishnaCam (Singh et al., 2016) datasets of real-world egocentric videos. Memory Storyboard outperforms state-of-the-art unsupervised continual learning methods

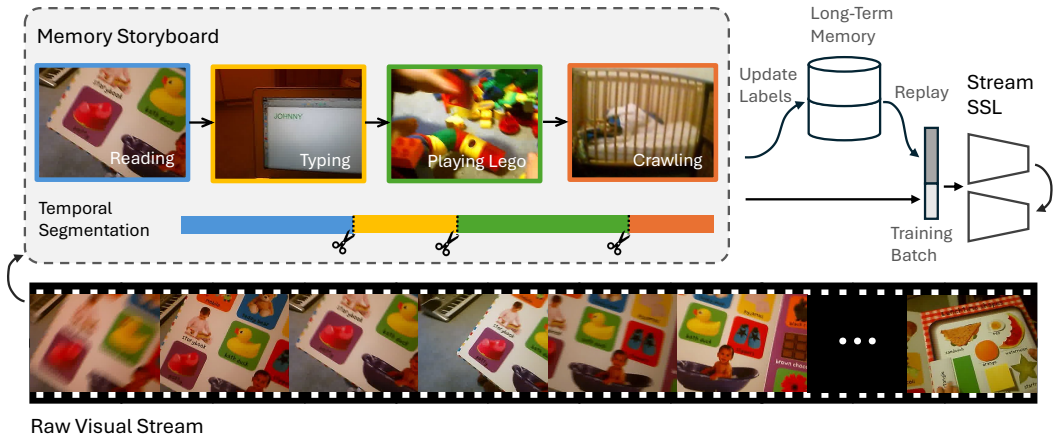

Figure 1: **Memory Storyboard framework for streaming self-supervised learning (SSL) from egocentric videos.** Given a continuous stream of images from an egocentric video, Memory Storyboard effectively learns visual representations by clustering similar frames into temporal segments and updating their labels (text information for illustration purposes only) in the long-term memory buffer for replay. SSL involves contrastive learning at both the frame and temporal segment levels.

on downstream image classification and object detection tasks and significantly reduces the gap between streaming learning and the less flexible IID learning that requires persistent storage of the entire prior video data. We also experiment with different buffer sizes and batch sizes and offer insights into the optimal training batch composition under different memory constraints.

We summarize our contributions as follows:

1) We introduce Memory Storyboard, a novel streaming SSL framework that features temporal segmentation and a two-tier memory hierarchy for efficient learning and temporal abstraction.
2) We demonstrate that Memory Storyboard achieves state-of-the-art performance on downstream ImageNet (Deng et al., 2009) and iNaturalist (Van Horn et al., 2018) classification tasks when trained on real-world egocentric video datasets. Among all the streaming self-supervised learning methods we evaluated, Memory Storyboard is the only one that is competitive with or even outperforms IID training when trained on these datasets.
3) We study the effects of training factors including label merging, subsampling rate, average segment length, memory buffer size, and training batch composition. These studies provide insight for more efficient streaming learning from videos. In particular, we explore the optimal composition ratio of the training batch from short-term vs. long-term memory, under different memory constraints. Larger batches from long-term memory improve performance when we can afford a large memory bank, while smaller batches can help prevent overfitting when we have a small memory bank.

2 Streaming SSL from Egocentric Videos

In streaming self-supervised learning, the goal is to learn useful visual representations from a continuous stream of inputs $(x_1, x_2, \dots)$. Similar to continual learning, we impose a memory budget so that storing the entire video would violate the constraint. Different from standard continual learning, there is no explicit notion of task, and the data distribution shift follows directly from the scene transitions of a video. The learner needs to make changes to the model as it sees new inputs, and finishes learning as soon as it receives the last input of the stream. The streaming setting is similar to Online Continual Learning (Mai et al., 2021; Guo et al., 2022; Wei et al., 2023), but the focus here is primarily on streaming video frames instead of a fixed dataset of static images. We argue that streaming learning from sequential video frames enables better modeling of naturalistic scene transitions in real-world data streams because a stream of image collections often includes artificial class transitions.

Streaming Training Batches. At each training step t, the model fetches a new batch of b images $X_t = x_{tb:(t+1)b}$ from the video stream and updates its parameters upon receiving X_t. At the end of the video, we evaluate the final model checkpoint on various downstream tasks such as object classification and detection, which are fundamental tasks for visual scene understanding as they enable models to recognize and interpret the contents of complex environments.

Standard SSL Fails on Streaming Video. Directly applying the SSL method sequentially on X_t gives very poor performance (Purushwalkam et al., 2022; Ren et al., 2021). This is not only due to catastrophic forgetting (McCloskey & Cohen, 1989) caused by the non-stationary distribution of visual features in the stream, but also due to the high temporal correlation of images in the stream (illustrated in Figure 1). This temporal correlation breaks the IID assumption held by common optimization algorithms like SGD or Adam (Kingma & Ba, 2015). For contrastive learning algorithms like SimCLR (Chen et al., 2020), the similarity across different frames in the same training batch would violate the assumption that each image is different.

Memory Replay. Similar to previous works (Hu et al., 2022; Yu et al., 2023; Purushwalkam et al., 2022), we use a replay buffer M with finite size $|M|$ to mitigate these issues. The model can store some of the fetched images in the replay buffer, and use both samples from the replay buffer and the new frames to form a training batch of size B. By sampling from the replay buffer we de-correlate the frames in the training batch and at the same time reduce the distribution shift between training batches.

Benefits of Streaming SSL over Other Settings. Compared to the traditional self-supervised learning setting, where all the frames are shuffled and uniformly sampled for each batch (we refer to this as "IID learning" in the text below), streaming SSL allows embodied agents to learn good visual representations from natural, uncurated video streams. It also involves less computation delay and less memory storage. For instance, a robot in a new environment can continuously adapt the visual representations from its own egocentric video feed without any human curation.

3 Related Work

In this section, we discuss the most relevant prior works. Please refer to Appendix B for additional related work.

Unsupervised Continual Learning. Unsupervised Continual Learning (UCL) (Rao et al., 2019; Smith et al., 2021; Madaan et al., 2022; Fini et al., 2022; Gomez-Villa et al., 2022; 2024; Cheng et al., 2023; Zhang et al., 2024) aims at learning a good representation through an unlabeled non-stationary data stream. Existing works in UCL often assume that the data stream is composed of a series of episodes and a stationary data distribution within each episode. This is not as naturalistic and human-like as our streaming setting, where the data distribution changes continuously through the data stream, and each image appears in the data stream only once. Meanwhile, we showed that existing UCL methods are also effective in our streaming video setting, and can be used together with the supervised contrastive objective.

Streaming Learning from Videos. While a number of recent papers have studied streaming learning from images (Hayes et al., 2019; Hayes & Kanan, 2020; Hayes et al., 2020; Banerjee et al., 2021) or IID self-supervised learning from video frames (Venkataramanan et al., 2023; Wang et al., 2024), limited works have investigated the problem of streaming learning from a continuous video stream. Roady et al. (2020) introduces a benchmark for streaming classification and novelty detection from videos. Zhuang et al. (2022) benchmarks many self-supervised learning methods in real-time and life-long learning settings in streaming video, assuming infinite replay buffer size which is unrealistic. Most similar to our setup, Purushwalkam et al. (2022) studies the task of continuous representation learning with a SimSiam objective (Chen & He, 2021) and proposes using a minimum-redundancy replay buffer. Their work also belongs to the broader range of works that study replay buffer sampling strategies in continual learning (Aljundi et al., 2019; Wiewel & Yang, 2021; Tiwari et al., 2022; Hacohen & Tuytelaars, 2024). Our work extends these prior works by adopting a two-tier replay buffer and a temporal segmentation component. Also relevant to our work, Carreira et al. (2024) studies online learning from a continuous video stream using a pixel-to-pixel reconstruction loss for representation learning. Their findings on the effect of pre-training and different optimization schemes are orthogonal with the ones in our work. It is worth pointing out that their exploration mainly focuses on settings without data augmentation and replay, limiting the efficacy of their framework.

4 Memory Storyboard

We present Memory Storyboard, an effective method for streaming SSL from egocentric videos. Memory Storyboard includes a temporal segmentation module and a two-tier memory hierarchy. It combines a standard self-supervised contrastive loss with a temporal contrastive objective that leverages the temporal class labels produced by the temporal segmentation module. Figure 2 illustrates the details of our method. The overall data processing and training procedure is summarized in Algorithm 2.

Temporal Segmentation. We describe our temporal segmentation algorithm as follows. Similar to Potapov et al. (2014), we are given a down-sampled video frame sequence of length L, with frames $x_1, x_2, \cdots, x_L$, and a feature extractor f_θ. We aim to find change points $t_1, t_2, \cdots, t_{n-1}$ so that the video is divided into n semantically-consistent segments $[x_1, x_{t_1}], [x_{t_1}, x_{t_2}], \cdots, [x_{t_{n-1}}, x_L]$. We also define $t_0 = 0$ and $t_n = L$. In this work, we determine the number of segments with $n = \frac{L}{T}$, where T refers to the average segment length and is a hyper-parameter.

The optimization objective of our segmentation algorithm is to maximize the average within-class similarity, such that each temporal segment captures a coherent scene, i.e.

$$\max_{t_1, t_2, \cdots, t_{n-1}} \sum_{i=2}^{n} \frac{1}{t_i - t_{i-1}} \sum_{j=t_{i-1}}^{t_i} \sum_{k=j}^{t_i} sim(x_j, x_k). \tag{1}$$

where $sim(x_j, x_k)$ denotes the cosine similarity between the embeddings $f_\theta(x_j)$ and $f_\theta(x_k)$. We compute the approximate solution to this optimization problem with a greedy approach, as detailed in Algorithm 1. We adopt this simple temporal segmentation approach in order to get good segmentation results in the beginning when the encoder network does not provide good representations. We leave it to future work to investigate different temporal segmentation strategies.

Two-tier Memory Hierarchy. Inspired by the Complementary Learning Systems (CLS) theory (McClelland et al., 1995; O'Reilly et al., 2014) of the human brain, we propose a two-tier memory hierarchy to accommodate efficient temporal segmentation. Shown in Figure 2, the system includes a long-term memory M_{long} updated with reservoir sampling (Vitter, 1985), and a short-term memory storyboard M_{short} updated with a first-in-first-out (FIFO) strategy. We store the temporal index and the temporal class of each frame along with the image in the memory. The short-term memory size $|M_{short}|$ is much smaller than the long-term memory size $|M_{long}|$, allowing efficient temporal segmentation of the recent past. The change points produced by the temporal segmentation component on M_{short} are then used to update the temporal class labels in M_{long}.

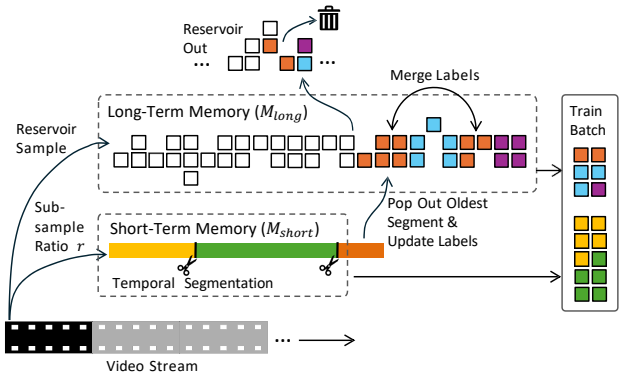

Figure 2: **Details of our two-tier memory in Memory Storyboard.** Long-term memory is updated with reservoir sampling and short-term memory with first-in-first-out (FIFO). Temporal segmentation is applied on the short-term memory, which then updates the labels of corresponding images in the long-term memory.

To increase the horizon of the memory storyboard, we subsample the frames coming from the current stream before adding it to M_{short}. The subsampling also reduces the temporal correlation between the frames in the training batch sampled from M_{short}. Compared to using a single replay buffer as memory, the two-tier memory hierarchy helps avoid overfitting the replay buffer and makes sure that the new frames are seen by the model.

Label Merging. Same objects and scenes often repeat in egocentric video streams. To efficiently share visual concept labels, we introduce here a label merging mechanism. When a new temporal segment is added to M_{long}, we compute the cosine similarity between its average frame embedding and those of existing segments. If the maximum similarity exceeds a threshold δ, the new segment inherits the class label of the most similar segment. This mechanism is activated only after the first C segments, as early-stage embeddings tend to be uniformly high in similarity. Formally, let v_i denote the average embedding of segment i in M_{long}, and v_n for the new segment n. Define $j = \arg\max_i \text{sim}(v_i, v_n)$ and let c_j be the label of segment j. Then,

$$c_n = \begin{cases} c_j & \text{if } \text{sim}(v_j, v_n) > \delta \text{ and } n > C \\ \text{new label} & \text{otherwise.} \end{cases}$$

In practice, choosing a fixed threshold δ that generalizes across methods and datasets is challenging. To address this, rather than fixing δ manually, we define it dynamically based on a quantile threshold $\tau \in (0, 1)$. Specifically, we set δ as the τ-quantile of all off-diagonal values in the similarity matrix.

Temporal Contrastive Loss. To effectively utilize the temporal class labels for representation learning, we adopt the supervised contrastive (SupCon) loss (Khosla et al., 2020), which takes the samples with the same temporal class

label in a batch as positives and contrasts them from the remainder of the batch. Let f_{proj} be a projector network. For a batch of images with size B, we take two random augmentations of each image to get an augmented batch $\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{2B}$, and compute $z_i = f_{proj}(f_\theta(\tilde{x}_j))$ be the projected features of each augmented image $\tilde{x}_i$. Let y_i be the temporal class label of $\tilde{x}_i$ and $P(i) = \{p \in \{1, 2, \ldots, 2B\} \backslash \{i\} : y_p = y_i\}$.

$$\mathcal{L}_{TCL} = \sum_i \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{a \neq i} \exp(z_i \cdot z_a / \tau)}. \tag{2}$$

We refer to this as the temporal contrastive loss. It is conceptually similar to the temporal classification loss proposed in (Orhan et al., 2020). However, in the temporal classification loss, the size of the classification layer needs to be gradually expanded as more data is processed by the model and more temporal classes are formed. Hence, the temporal contrastive loss is more flexible and more suitable for the streaming SSL setting.

Overall Loss Function. In addition to the temporal contrastive loss, we also incorporate a standard self-supervised loss $\mathcal{L}_{SSL}$. In particular, we experimented with the SimCLR loss (Chen et al., 2020; Sohn, 2016) and the SimSiam loss (Chen & He, 2021) because they were shown to work well in lifelong self-supervised learning in prior works (Zhuang et al., 2022; Purushwalkam et al., 2022). The overall loss function is a sum of the temporal contrastive loss and the self-supervised contrastive loss $\mathcal{L} = \mathcal{L}_{TCL} + \mathcal{L}_{SSL}$.

Warm-Start Training. At the beginning of training, the model has only seen a very limited amount of data from the video stream. Even with a memory buffer, there is a likely high temporal correlation between the sampled frames which can cause instability in the training. To alleviate this problem, we warm-start the system by making no model updates on the first M_{long} frames of the stream and just use them to fill the memory. The warm-start phase ensures that the model is trained on de-correlated samples from the buffer starting from the beginning.

Algorithm 1 Temporal Segmentation

```
# n: number of clusters
# feats: features of the frames in the
    sequence
# F: maximization objective (defined by
    Equation 1).
# Returns: detected change points in
    the stream (sorted)

def temporal_segment(n, feats, F):
   S = feats @ feats.T
   L = len(S)
   changepts = []
   for i in range(1, n):
      bestscore = 0
      for changept in range(1, L):
         temp = changepts + [changept]
         score = F(sorted(temp))
         if score > bestscore:
            bestscore = score
            bestchangept = changept
      changepts.append(bestchangept)
   return sorted(changepts)
```

Algorithm 2 Memory Storyboard Streaming SSL

```
# D: streaming data loader
# M_s: short-term memory buffer
# M_l: long-term memory buffer
# B_s, B_l: batch size for M_s, M_l
# T: default segment length
# r: subsampling rate

while True: # Loop until end of stream
   x = D.next()
   x_sub = subsample(x, r)
   M_l.add(x) # Updated with Reservoir
   M_s.add(x_sub) # Updated with FIFO
   if M_s[0].label > tc_label:
      tc_label = M_s[0].label
      n = len(M_s) / T
      feats = normalize(features(M_s))
      changes = temporal_segment(n, feats, F)
      update_labels(M_s, changes)
      update_labels(M_l, changes)
   data = sample(M_l, B_l, M_s, B_s)
   loss = TCL_loss(data) + SSL_loss(data)
   model.update(loss)
```

5 Experiments

5.1 Experiment Setup

Datasets. We use two real-world egocentric video datasets in the experiments: (1) the child S subset of SAYCam dataset (Sullivan et al., 2021), which contains 221 hours of video data collected from a head-mounted camera on the child from age 6-32 months, decoded at 25 fps; (2) the KrishnaCam dataset (Singh et al., 2016), which contains 70 hours of video data spanning nine months of the life of a graduate student, decoded at 10 fps. These two datasets have also been adopted in a number of existing self-supervised learning literature (Orhan et al., 2020; Purushwalkam et al., 2022; Zhuang et al., 2022; Vong et al., 2024).

Training. Following the architectural choices of Osiris (Zhang et al., 2024), we use ResNet-50 (He et al., 2016) as the feature extractor with group normalization (Wu & He, 2018) and the Mish activation function (Misra, 2020). Unless otherwise specified, the default hyperparameter values we use in our experiments are $b = 64$, $B = 512$, $T = 4.5K$ for SAYCam and $T = 1.8K$ for KrishnaCam (both corresponding to 3 minutes of raw video), subsampling rate $r = 8$ for SAYCam and $r = 4$ for KrishnaCam. We train the models with two sets of memory sizes to evaluate their performance across different memory constraints: a larger memory constraint with $|M| = 50K$, $|M_{short}| = 5K$, $|M_{long}| = 45K$, and a smaller memory constraint with $|M| = 10K$, $|M_{short}| = 1K$, $|M_{long}| = 9K$. For context, there are a total of 18.2M frames in the SAYCam training set and 2.5M frames in the KrishnaCam training set. Therefore, even the large memory constraint of 50K frames only stores 0.27% and 2.01% of the total training frames in the memory buffer for SAYCam and KrishnaCam respectively. For the main experiments (Tables 1 and 2), we employ label merging with $\tau = 0.998$ (i.e., the similarity threshold δ is the top 0.002 quantile of the off-diagonal values in the similarity matrix); for all other experiments, label merging is not employed unless explicitly specified otherwise. Each experiment is run on one A100 GPU.

Evaluation. For object classification, we use *mini*-ImageNet classification task for both SAYCam and KrishnaCam models. For each dataset, we also pick another downstream task that evaluates the learned representations of the training data itself. Evaluation tasks are summarized below.

- ***mini*-ImageNet Classification:** Following a similar evaluation protocol as Zhuang et al. (2022), we evaluate the learned representations on downstream classification of a subsampled ImageNet (Deng et al., 2009) dataset (*mini*-INet). We extract the features of the model and train a support vector machine (SVM) to measure its classification performance. The *mini*-ImageNet dataset contains 20K training images and 5K test images across 100 classes.
- **ImageNet-1K and iNaturalist Classification:** Similar to the evaluation protocol used in Purushwalkam et al. (2022), we further evaluate the classification performance with a linear classifier on the larger ImageNet-1K (Deng et al., 2009) dataset (INet) with 1.28M training images and 50K test images across 1K classes, and the iNaturalist-2018 (Van Horn et al., 2018) dataset (iNat) with 437K training images and 24K test images across 8142 classes.
- **Labeled-S Classification:** For SAYCam models, we evaluate the classification performance on the Labeled-S dataset, following Orhan et al. (2020). The Labeled-S dataset is a labeled subset of the SAYCam frames, containing a total of 5786 images across 26 classes after 10x subsampling of frames. We randomly use 50% as training data and 50% as test data.
- **OAK Object Detection:** For KrishnaCam models, we evaluate the object detection performance on the Objects Around Krishna (OAK) dataset (Wang et al., 2021), which includes bounding box annotations of 105 object categories on a subset of the KrishnaCam frames. We fine-tune the model on the entire training set of OAK for 10 epochs before evaluating on the OAK validation set, and report the AP50 metric.

Baselines. We compare Memory Storyboard with a number of competitive SSL methods for image and video representation learning, and different memory buffer strategies:

- **SimCLR:** In prior studies, Zhuang et al. (2022) showed that SimCLR (Chen et al., 2020) is the strongest self-supervised learning method under streaming video setting, outperforming other SSL methods such as BYOL (Grill et al., 2020) and Barlow Twins (Zbontar et al., 2021).
- **SimSiam:** In prior work, Purushwalkam et al. (2022) showed that SimSiam (Chen & He, 2021) is able to learn good representations from egocentric video data.
- **Osiris:** Osiris (Zhang et al., 2024) is a state-of-the-art unsupervised continual learning method that is developed towards static image sequences.
- **TC:** Temporal classification (TC) (Orhan et al., 2020) is a simple self-supervised learning method that is shown to work well on the SAYCam dataset under IID setting. It also uses temporal segments as a source of self-supervision; however, it does not actively group the frames together but instead relies on fixed intervals.
- **Reservoir Sampling:** We mainly use reservoir sampling (Vitter, 1985) as a default baseline approach for updating the memory buffer, which uniformly samples from all the seen images in the memory.
- **MinRed Buffer:** The minimum redundancy (MinRed) buffer (Purushwalkam et al., 2022) is a streaming self-supervised learning algorithm that alleviates temporal correlations in continuous video streams by storing minimally redundant samples in the replay buffer.
- **Two-tier Buffer:** As in MemStoryboard, we use a long-term memory updated with reservoir sampling and short-term memory updated with first-in-first-out (FIFO), but we do not apply the temporal contrastive loss or the temporal segmentation module.

Method	*mini*-INet	INet	iNat	Labeled-S	*mini*-INet	INet	iNat	Labeled-S
IID SimCLR (Chen et al., 2020)	44.04	30.44	8.69	59.50	44.04	30.44	8.69	59.50
IID SimSiam (Chen & He, 2021)	29.02	20.92	4.91	42.71	29.02	20.92	4.91	42.71
SimCLR No Replay	5.76	2.22	0.07	19.13	5.76	2.22	0.07	19.13
SimSiam No Replay	6.44	1.47	0.04	22.03	6.44	1.47	0.04	22.03
	Replay - 10k				*Replay - 50k*			
Osiris (Zhang et al., 2024)	31.16	19.48	4.68	45.81	36.90	23.16	5.85	50.88
TC (Orhan et al., 2020)	33.92	19.03	5.84	48.09	36.68	22.72	8.24	52.22
SimCLR (Chen et al., 2020)	33.02	20.13	4.74	49.29	37.96	23.75	6.91	53.67
+MinRed (Purushwalkam et al., 2022)	33.62	20.21	5.12	48.88	38.66	24.10	6.81	54.75
+Two-tier (Ours)	33.80	20.70	5.61	49.05	39.22	24.93	7.07	55.43
+MemStoryboard (Ours)	34.18	22.59	6.34	**51.09**	38.84	26.87	8.17	**56.26**
SimSiam (Chen & He, 2021)	20.90	13.72	2.55	39.12	26.66	14.44	3.79	43.09
+MinRed (Purushwalkam et al., 2022)	22.68	17.85	3.17	39.78	25.58	18.99	4.24	40.37
+Two-tier (Ours)	21.78	16.87	2.76	39.19	28.34	20.24	3.99	42.95
+MemStoryboard (Ours)	**36.86**	**26.70**	**8.46**	49.87	**41.46**	**28.92**	**10.41**	53.78

Table 1: **Results on streaming SSL from SAYCam (Sullivan et al., 2021).** Downstream evaluation on object classification (Accuracy %) for SSL models trained under the streaming setting. For "No Replay" and "IID" the results are the same for different memory buffer sizes. The "IID" methods are not under the streaming setting and are for reference only as a performance "upper bound" with the same number of gradient updates. Unless specified, standard reservoir sampling is used in the replay buffer.

Method	*mini*-INet	INet	iNat	OAK	*mini*-INet	INet	iNat	OAK
IID SimCLR (Chen et al., 2020)	36.90	23.77	5.60	39.54	36.90	23.77	5.60	39.54
IID SimSiam (Chen & He, 2021)	28.58	22.28	4.16	44.86	28.58	22.28	4.16	44.86
SimCLR No Replay	4.84	1.35	0.07	14.01	4.84	1.35	0.07	14.01
SimSiam No Replay	8.88	1.92	0.05	27.34	8.88	1.92	0.05	27.34
	Replay - 10k				*Replay - 50k*			
Osiris (Zhang et al., 2024)	30.10	19.03	3.55	32.25	32.38	20.85	3.96	33.78
TC (Orhan et al., 2020)	32.58	19.19	6.01	32.61	32.94	20.50	6.25	28.56
SimCLR (Chen et al., 2020)	31.46	19.09	4.43	31.92	34.98	22.37	5.19	33.30
+MinRed (Purushwalkam et al., 2022)	31.56	19.93	4.69	34.78	34.84	22.29	5.30	35.65
+Two-tier (Ours)	33.26	20.39	5.04	33.72	35.78	22.42	5.29	35.68
+MemStoryboard (Ours)	**33.30**	22.52	5.65	36.01	**36.08**	25.37	6.28	38.58
SimSiam (Chen & He, 2021)	19.16	12.94	2.85	39.38	21.84	14.13	3.56	41.13
+MinRed (Purushwalkam et al., 2022)	20.90	14.53	3.16	43.74	22.88	17.64	5.12	44.17
+Two-tier (Ours)	20.08	13.76	2.91	43.68	22.14	17.06	3.73	44.41
+MemStoryboard (Ours)	33.22	**23.76**	**6.52**	**45.18**	34.62	**25.52**	**6.78**	**46.17**

Table 2: **Results on streaming SSL from KrishnaCam (Singh et al., 2016).** Downstream evaluation on object classification (Accuracy %) and object detection (AP50 %) for SSL models trained under the streaming setting. The structure of the table is otherwise similar to Table 1.

5.2 Main Results

In Tables 1 and 2, we report the main results on streaming SSL on both SAYCam and KrishnaCam. Firstly, we observe that all SSL methods work poorly in the streaming setting without replay, and larger memory leads to better performance. In terms of memory buffer strategies, our two-tier memory hierarchy and MinRed (Purushwalkam et al., 2022) outperform reservoir sampling.

Memory Storyboard achieves superior performance in all readout tasks compared to other streaming SSL models. For SimCLR-based methods, Memory Storyboard considerably narrows the gap between streaming learning and IID training. Memory Storyboard also significantly outperforms all baseline methods with a considerable gap on both the ImageNet classification and the challenging OAK object detection benchmark. For SimSiam-based methods, Memory Storyboard not only outperforms all streaming learning baselines by a considerable margin but also beats IID SimSiam training on all readout tasks.

Memory Storyboard with SimSiam achieves the overall best performance across different training datasets and evaluation metrics. We hypothesize that Memory Storyboard works better with SimSiam (Chen & He, 2021) than Sim-

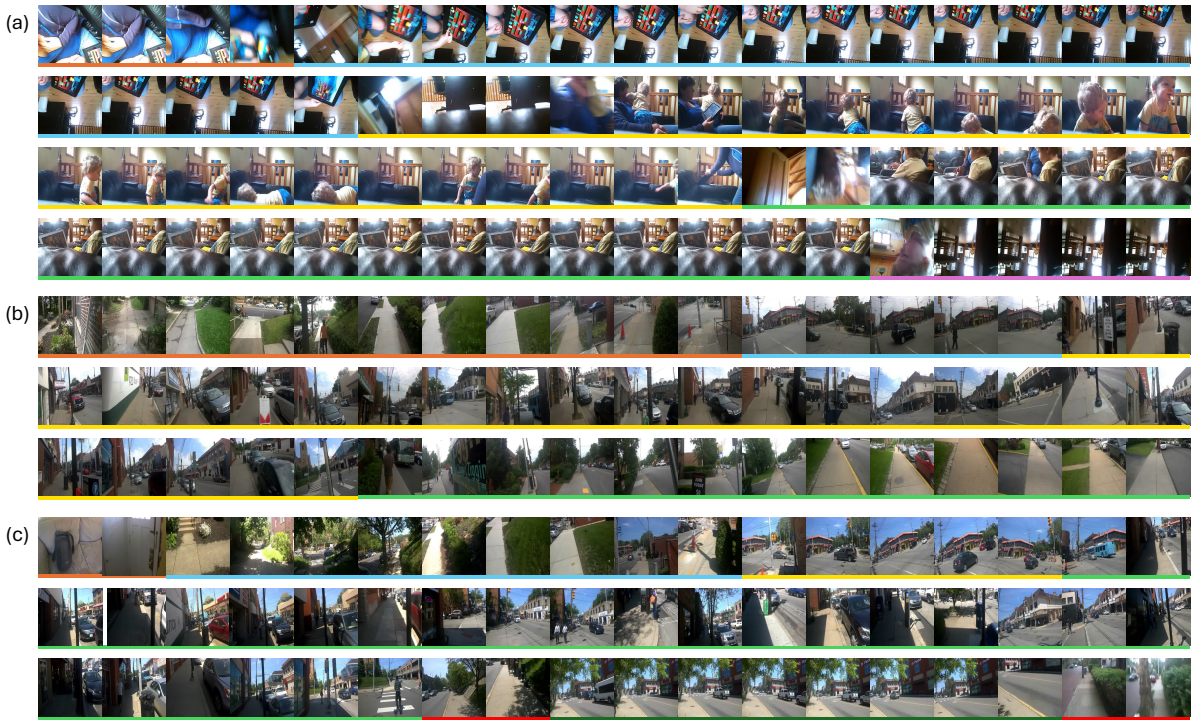

Figure 3: **Visualization of the temporal segments produced by Memory Storyboard on (a) SAYCam (b)(c) KrishnaCam at the end of training.** The images are sampled at 10 seconds per frame. Each color bar corresponds to a temporal class (the first and the last class might be incomplete). Temporal segments produced at the beginning of training are provided in the appendix for comparison.

CLR (Chen et al., 2020) in our experiments because the SimCLR loss can be conflicting with the temporal contrastive objective. SimCLR treats some highly correlated images in the same batch as negative samples during training, despite the fact that the other images in the batch sampled from the short-term buffer might be very similar to the current image since they are temporally close to each other. This issue is exacerbated in the SAYCam experiments due to the high frequency (25 fps) of the SAYCam video stream. By incorporating the temporal contrastive loss in Memory Storyboard, we successfully address this issue by utilizing only images in other temporal classes as negative samples.

Overall, the results demonstrate that Memory Storyboard is effective at learning good representations from a streaming video source, and the learned representations can be successfully transferred to downstream vision tasks on the training dataset itself or an external dataset.

Qualitative Results. We visualize the temporal segments produced by Memory Storyboard at the end of training in Figure 3. The results demonstrate that our temporal segmentation module can produce semantically meaningful temporal segments, showing its strong temporal abstraction capability. We emphasize that the representations are entirely developed during the streaming SSL training as the networks are trained from scratch. We provide additional qualitative results in Appendix C.

5.3 Ablation Experiments and Other Training Factors

In this section, we study how varying different training factors affect the performance of Memory Storyboard, including label merging, subsampling rate, and average segment length. We use SimCLR (Chen et al., 2020) as the base SSL method for training, and a long-term memory size of 50K unless otherwise specified. Please refer to Appendix C for additional ablation experiments.

Label Merging. We train MemStoryboard with and without the label merging mechanism and present the results in Tables 3 and 4. We observe that incorporating label merging consistently improves performance on the ImageNet and iNaturalist classification tasks, which are out-of-domain classification settings. This suggests that grouping labels for representation learning is helpful in recognizing new classes at test time. While the improvements are generally modest

Base SSL Method	Label Merging	*mini*-INet	INet	iNat	Labeled-S	*mini*-INet	INet	iNat	Labeled-S
		Replay - 10k				*Replay - 50k*			
SimCLR	✗	35.02	20.72	5.65	**51.33**	39.58	24.78	7.77	**56.29**
SimCLR	✓	34.18	22.59	6.34	51.09	38.84	26.87	8.17	56.26
SimSiam	✗	36.72	22.99	6.66	49.12	41.32	26.37	9.85	53.29
SimSiam	✓	**36.86**	**26.70**	**8.46**	49.87	**41.46**	**28.92**	**10.41**	53.78

Table 3: Ablation on label merging for MemStoryboard trained on SAYCam.

Base SSL Method	Label Merging	*mini*-INet	INet	iNat	OAK	*mini*-INet	INet	iNat	OAK
		Replay - 10k				*Replay - 50k*			
SimCLR	✗	33.72	20.13	5.64	35.77	**36.36**	22.75	6.10	38.67
SimCLR	✓	33.30	22.52	5.65	36.01	36.08	25.37	6.28	38.58
SimSiam	✗	33.78	21.38	6.51	**45.33**	35.20	22.75	6.71	**46.64**
SimSiam	✓	**33.22**	**23.76**	**6.52**	45.18	34.62	**25.52**	**6.78**	46.17

Table 4: Ablation on label merging for MemStoryboard trained on KrishnaCam.

in absolute terms, they are robust across different datasets and buffer sizes. Performance on the other benchmarks (*mini*-ImageNet, Labeled-S, and OAK) remains largely stable with or without label merging.

Subsampling Rate. We train Memory Storyboard with different subsampling rates when adding data fetched from the current stream to the short-term memory. Results are shown in Table 5. A subsampling ratio of 8 works best for SAYCam, while a ratio of 4 works best for KrishnaCam. Since the two datasets are decoded at different frequencies (25 fps for SAYCam and 10 fps for KrishnaCam), the effective frequency of frames entering the short-term buffer is 3.13 and 2.50 fps respectively. The result suggests that an effective frequency of around 3 fps is preferable although the optimal subsample ratio is dependent on the nature of the video stream. Intuitively, when the subsampling ratio is too small, the images entering the short-term buffer may have too much temporal correlation and hence would hurt the performance; when the subsampling ratio is too big, the model skips too many frames without training on them and the temporal clustering may also become less precise.

Subsample Ratio	SAYCam		KrishnaCam	
	mini-INet	Labeled-S	*mini*-INet	OAK AP50
1×	36.70	55.29	35.54	38.55
2×	37.18	55.43	35.60	37.38
4×	38.38	55.84	**36.36**	38.67
8×	**39.58**	**56.29**	35.48	**38.90**
16×	38.62	55.81	35.88	38.22

Table 5: Effect of subsampling ratio for M_{short} in Memory Storyboard.

T	SAYCam		KrishnaCam	
	mini-INet	Labeled-S	*mini*-INet	OAK AP50
1 min	38.90	55.05	35.86	38.57
2 min	39.16	56.53	36.30	**38.68**
3 min	**39.58**	56.29	**36.36**	38.67
5 min	39.26	**56.64**	36.28	38.07
10 min	38.34	55.36	35.98	37.53

Table 6: Performance of Memory Storyboard using different average temporal segment lengths.

Average Segment Length. We trained Memory Storyboard with different average segment lengths T ranging from 1 minute to 10 minutes on SAYCam and KrishnaCam. The results are shown in Table 6. We demonstrate that the performance of Memory Storyboard is generally robust to average segment length (which determines the number of temporal segments in the segmentation module). We also find that the performance on downstream tasks becomes worse when the average segment length is very long ($T = 10$ min) on both datasets. This observation is different from that of temporal classification (Orhan et al., 2020) which claims longer segments are more helpful.

5.4 Optimal Batch Composition Under Different Memory Constraints

In Memory Storyboard, the training batch is composed of samples from both the long-term memory and the short-term memory (see Figure 2). However, the optimal composition ratio of the training batch, i.e. the optimal percentage of data in the training batch that comes from the short-term memory, is yet to be explored. Sampling more data from the short-term memory means we can digest more data within a fixed number of training steps, but there will be more distribution shift between different training batches. On the other hand, sampling more data from the long-term memory

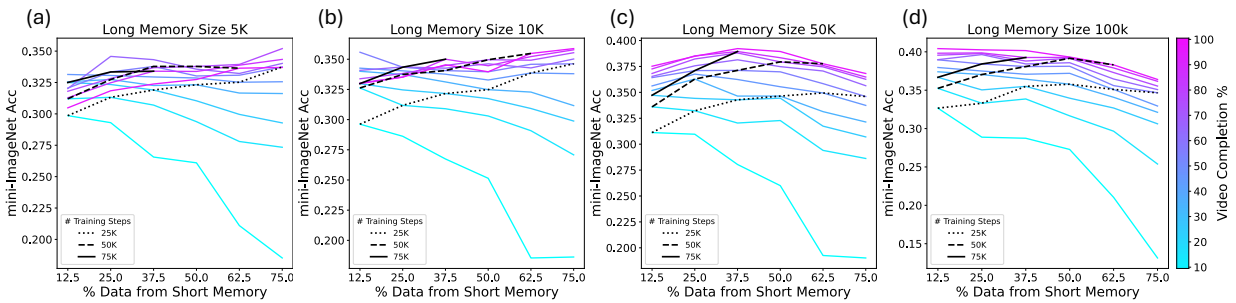

Figure 4: **Memory Storyboard model performance on SAYCam with different long-term memory sizes** (5k, 10k, 50k, and 100k) **and varying training batch compositions** (12.5% – 75.0% from M_{short}) using SVM readout. Each colored line represents the performance of different training batch compositions when **the model has seen the same amount of data** from the stream. Each black line represents the performance of different training batch compositions when the model has taken **the same number of gradient updates**.

buffer may result in overfitting on the long-term memory data. In this section, we experiment with different memory sizes and training composition and demonstrate the optimal batch composition under different memory constraints.

We fix the size of the short-term memory $|M_{short}|$ to be $5K$ and vary the memory constraint for the long-term memory $|M_{long}| = 5K, 10K, 50K, 100K$. For each long-term memory size, we experiment with batch size from data stream $b = 64, 128, 192, 256, 320, 384$ (which corresponds to 12.5% though 75% of the training batch size). We sample b images from the short-term memory and $512 - b$ images from the long-term memory to compose a training batch. We evaluate the model with SVM readout on *mini*-ImageNet after the model has seen every 10% of the entire data stream and plot the results in Figure 4. We discuss the different observations for large memory size and small memory size respectively.

- **Large $|M_{long}|$:** When $|M_{long}|$ is large (Figures 4(c) and 4(d)), overfitting on the memory is unlikely and hence we can sample more data from the long-term memory and the performance still keeps increasing as the model sees more data. Hence, with the same amount of data seen by the model (colored curves), it is better to sample only a small batch from the short-term memory. However, when we control the number of model update steps to the same (black curves), neither focusing on the short-term memory nor focusing on the long-term memory is preferable. In such cases, the optimal batch size from the short-term memory is at roughly 50% of the training batch.
- **Small $|M_{long}|$:** When $|M_{long}|$ is small (Figures 4(a) and 4(b)), the model is prone to overfitting on the memory. As a result, with the same number of model update steps (black curves), taking more images from M_{short} gives better results. With the same amount of data seen by the model (colored curves), getting a higher percentage of data from M_{long} has an advantage in the beginning when there is less memory overfitting. Ultimately, focusing on M_{short} is more beneficial in the late stage.

To summarize, the optimal training batch composition depends on memory and compute constraints. More samples from the long-term memory are preferred when a large memory is available (e.g., 50K images from a 200-hour stream) and model performance is evaluated after seeing a fixed amount of data. More samples from the short-term memory are preferred when memory is limited to prevent overfitting on the long-term memory data. When both memory and compute are sufficient and performance is measured under a fixed compute budget, a balanced batch composition is most effective for real-time learning.

6 Conclusion

The ability to continuously learn from large-scale uncurated streaming video data is crucial for applying self-supervised learning methods in real-world embodied agents. Existing works have limited exploration of this problem, have mainly focused on static datasets, and do not perform well in the streaming video setting. Inspired by the event segmentation mechanism in human cognition, in this work, we propose Memory Storyboard, which leverages temporal segmentation to produce a two-tier memory hierarchy akin to the short-term and long-term memory of humans. Memory Storyboard combines a temporal contrastive objective and a standard self-supervised contrastive objective to facilitate representation learning from scratch through streaming video experiences. Memory Storyboard achieves state-of-the-art performance on downstream classification and object detection tasks when trained on real-world large egocentric video datasets. By studying the effects of subsampling rates, average segment length, normalization, and optimal batch composition under different compute and memory constraints, we also offer valuable insights on the design choices for streaming self-supervised learning.

ACKNOWLEDGEMENTS

We thank James McClelland for a helpful discussion on event segmentation and pointers to relevant literature. We thank Peiqi Liu and Anh Ta for their explorations on training self-supervised learning models on the SAYCam dataset. The work is supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) under grant RS-2024-00469482, funded by the Ministry of Science and ICT (MSIT) of the Republic of Korea in connection with the Global AI Frontier Lab International Collaborative Research. YY is supported by the Meta AI Mentorship Program. The compute is supported through the NYU IT High Performance Computing resources, services, and staff expertise.

REFERENCES

Mohamed Afham, Satya Narayan Shukla, Omid Poursaeed, Pengchuan Zhang, Ashish Shah, and Sernam Lim. Revisiting kernel temporal segmentation as an adaptive tokenizer for long-form video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1189–1194, 2023.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023.

Christopher Baldassano, Janice Chen, Asieh Zadbood, Jonathan W Pillow, Uri Hasson, and Kenneth A Norman. Discovering event structure in continuous narrative perception and memory. *Neuron*, 95(3):709–721, 2017.

Dare A Baldwin, Jodie A Baird, Megan M Saylor, and M Angela Clark. Infants parse dynamic action. *Child development*, 72(3):708–717, 2001.

Soumya Banerjee, Vinay Kumar Verma, Toufiq Parag, Maneesh Singh, and Vinay P Namboodiri. Class incremental online streaming learning. *arXiv preprint arXiv:2110.10741*, 2021.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *The Tenth International Conference on Learning Representations*, 2022.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28751–28761, 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Haoyang Cheng, Haitao Wen, Xiaoliang Zhang, Heqian Qiu, Lanxiao Wang, and Hongliang Li. Contrastive continuity on augmentation stability rehearsal for continual self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5707–5717, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pp. 1422–1430, 2015.

Sarah DuBrow and Lila Davachi. The influence of context boundaries on memory for the sequential order of events. *Journal of Experimental Psychology: General*, 142:1277–1286, 08 2013. doi: 10.1037/a0034024.

Youssef Ezzyat and Lila Davachi. What constitutes an episode in episodic memory? *Psychological science*, 22: 243–52, 02 2011. doi: 10.1177/0956797610393742.

Enrico Fini, Victor G Turrisi Da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9621–9630, 2022.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *6th International Conference on Learning Representations*, 2018.

Alex Gomez-Villa, Bartlomiej Twardowski, Lu Yu, Andrew D Bagdanov, and Joost Van de Weijer. Continually learning self-supervised representations with projected functional regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3867–3877, 2022.

Alex Gomez-Villa, Bartlomiej Twardowski, Kai Wang, and Joost Van de Weijer. Plasticity-optimized complementary networks for unsupervised continual learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1690–1700, 2024.

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. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Yiduo Guo, Bing Liu, and Dongyan Zhao. Online continual learning through mutual information maximization. In *International conference on machine learning*, pp. 8109–8126. PMLR, 2022.

Guy Hacohen and Tinne Tuytelaars. Forgetting order of continual learning: Examples that are learned first are forgotten last. *arXiv preprint arXiv:2406.09935*, 2024.

Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 220–221, 2020.

Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9769–9776. IEEE, 2019.

Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *European conference on computer vision*, pp. 466–483. Springer, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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*, pp. 9729–9738, 2020.

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*, pp. 16000–16009, 2022.

Dapeng Hu, Shipeng Yan, Qizhengqiu Lu, Lanqing Hong, Hailin Hu, Yifan Zhang, Zhenguo Li, Xinchao Wang, and Jiashi Feng. How well does self-supervised pre-training perform with streaming data? In *The Tenth International Conference on Learning Representations*, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33: 18661–18673, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

G. Lassiter and David Slaw. The unitization and memory of events. *Journal of Experimental Psychology: General*, 120:80–82, 03 1991. doi: 10.1037/0096-3445.120.1.80.

Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Representational continuity for unsupervised continual learning. In *The Tenth International Conference on Learning Representations*, 2022.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3589–3599, 2021.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Diganta Misra. Mish: A self regularized non-monotonic activation function. In *31st British Machine Vision Conference*, 2020.

Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6707–6717, 2020.

Darren Newtson, Gretchen A Engquist, and Joyce Bois. The objective basis of behavior units. *Journal of Personality and social psychology*, 35(12):847, 1977.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pp. 69–84. Springer, 2016.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Emin Orhan, Vaibhav Gupta, and Brenden M Lake. Self-supervised learning through the eyes of a child. *Advances in Neural Information Processing Systems*, 33:9960–9971, 2020.

Randall C O'Reilly, Rajan Bhattacharyya, Michael D Howard, and Nicholas Ketz. Complementary learning systems. *Cognitive science*, 38(6):1229–1248, 2014.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544, 2016.

Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pp. 540–555. Springer, 2014.

Senthil Purushwalkam, Pedro Morgado, and Abhinav Gupta. The challenges of continuous self-supervised learning. In *European Conference on Computer Vision*, pp. 702–721. Springer, 2022.

Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. *Advances in neural information processing systems*, 32, 2019.

Mengye Ren, Tyler R. Scott, Michael L. Iuzzolino, Michael C. Mozer, and Richard S. Zemel. Online unsupervised learning of visual representations and categories. *arXiv preprint arXiv:2109.05675*, 2021.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Ryne Roady, Tyler L Hayes, Hitesh Vaidya, and Christopher Kanan. Stream-51: Streaming classification and novelty detection from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 228–229, 2020.

Mrigank Rochan, Linwei Ye, and Yang Wang. Video summarization using fully convolutional sequence networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 347–363, 2018.

Karen Sasmita and Khena Swallow. Measuring event segmentation: An investigation into the stability of event boundary agreement across groups. *Behavior Research Methods*, 55, 04 2022. doi: 10.3758/s13428-022-01832-5.

Megan M Saylor, Dare A Baldwin, Jodie A Baird, and Jennifer LaBounty. Infants' on-line segmentation of dynamic human action. *Journal of Cognition and Development*, 8(1):113–128, 2007.

Marta Silva, Christopher Baldassano, and Lluís Fuentemilla. Rapid memory reactivation at movie event boundaries promotes episodic encoding. *Journal of Neuroscience*, 39(43):8538–8548, 2019.

Krishna Kumar Singh, Kayvon Fatahalian, and Alexei A Efros. Krishnacam: Using a longitudinal, single-person, egocentric dataset for scene understanding tasks. In *2016 IEEE Winter Conference on Applications of Computer Vision*, pp. 1–9. IEEE, 2016.

James Seale Smith, Cameron E. Taylor, Seth Baer, and Constantine Dovrolis. Unsupervised progressive learning and the STAM architecture. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

Jessica Sullivan, Michelle Mei, Andrew Perfors, Erica Wojcik, and Michael C Frank. Saycam: A large, longitudinal audiovisual dataset recorded from the infant's perspective. *Open mind*, 5:20–29, 2021.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 776–794. Springer, 2020.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 99–108, 2022.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8769–8778, 2018.

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 preprint arXiv:2310.08584*, 2023.

Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1): 37–57, 1985.

Wai Keen Vong, Wentao Wang, A Emin Orhan, and Brenden M Lake. Grounded language acquisition through the eyes and ears of a single child. *Science*, 383(6682):504–511, 2024.

Alex N Wang, Christopher Hoang, Yuwen Xiong, Yann LeCun, and Mengye Ren. Poodle: Pooled and dense self-supervised learning from naturalistic videos. *arXiv preprint arXiv:2408.11208*, 2024.

Jianren Wang, Xin Wang, Yue Shang-Guan, and Abhinav Gupta. Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10829–10838, 2021.

Yujie Wei, Jiaxin Ye, Zhizhong Huang, Junping Zhang, and Hongming Shan. Online prototype learning for online continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18764–18774, 2023.

Felix Wiewel and Bin Yang. Entropy-based sample selection for online continual learning. In *2020 28th European signal processing conference (EUSIPCO)*, pp. 1477–1481. IEEE, 2021.

Jay Zhangjie Wu, David Junhao Zhang, Wynne Hsu, Mengmi Zhang, and Mike Zheng Shou. Label-efficient online continual object detection in streaming video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19246–19255, 2023.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Tristan S Yates, Lena J Skalaban, Cameron T Ellis, Angelika J Bracher, Christopher Baldassano, and Nicholas B Turk-Browne. Neural event segmentation of continuous experience in human infants. *Proceedings of the National Academy of Sciences*, 119(43):e2200257119, 2022.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

Xiaofan Yu, Yunhui Guo, Sicun Gao, and Tajana Rosing. Scale: Online self-supervised lifelong learning without prior knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2484–2495, 2023.

Jeffrey M Zacks and Khena M Swallow. Event segmentation. *Current directions in psychological science*, 16(2): 80–84, 2007.

Jeffrey M Zacks, Barbara Tversky, and Gowri Iyer. Perceiving, remembering, and communicating structure in events. *Journal of experimental psychology: General*, 130(1):29, 2001.

Jeffrey M Zacks, Nicole K Speer, Jean M Vettel, and Larry L Jacoby. Event understanding and memory in healthy aging and dementia of the alzheimer type. *Psychology and aging*, 21(3):466, 2006.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021.

Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, pp. 766–782. Springer, 2016.

Yipeng Zhang, Laurent Charlin, Richard Zemel, and Mengye Ren. Integrating present and past in unsupervised continual learning. *arXiv preprint arXiv:2404.19132*, 2024.

Wencheng Zhu, Jiwen Lu, Jiahao Li, and Jie Zhou. Dsnet: A flexible detect-to-summarize network for video summarization. *IEEE Transactions on Image Processing*, 30:948–962, 2020.

Chengxu Zhuang, Ziyu Xiang, Yoon Bai, Xiaoxuan Jia, Nicholas Turk-Browne, Kenneth Norman, James J DiCarlo, and Dan Yamins. How well do unsupervised learning algorithms model human real-time and life-long learning? *Advances in neural information processing systems*, 35:22628–22642, 2022.

A Experiment Details

Model Architecture. On top of the ResNet backbone, we use a two-layer MLP with 2048 hidden units, 128 output units, and ReLU activation function as the projector. In Memory Storyboard, we create two separate projectors for $\mathcal{L}_{TCL}$ and $\mathcal{L}_{SSL}$.

Training. For all experiments in Tables 1 and 2, we used a total batch size of 512 (64 from M_{short} and 448 from M_{long} by default). The input resolution of the images to the model is 112. We apply a standard data augmentation pipeline for SSL methods following Zhuang et al. (2022), which include random resized crop, random horizontal flip, random color jitter, random grayscale, random Gaussian filter, and color-normalization with ImageNet (Deng et al., 2009). For the SimCLR (Chen et al., 2020), Osiris (Zhang et al., 2024), and TC (Orhan et al., 2020) experiments, we used the Adam (Kingma & Ba, 2015) optimizer with a constant learning rate of 0.001, and a projector with 2 MLP layers of size 2048 and 128 respectively. For the SimSiam (Chen & He, 2021) experiments, we used the SGD optimizer with learning rate 0.05, momentum 0.9, and weight decay 1e-4, and a projector with 3 MLP layers of size 2048.

Evaluation. For *mini*-ImageNet and Labeled-S evaluations, the streaming SSL models are evaluated every 5% of the entire dataset. That is, we store 20 model checkpoints throughout the streaming training and evaluate them on *mini*-ImageNet and Labeled-S with SVM readout. The *best* results among these checkpoints are reported. Similar to Zhuang et al. (2022), for SVM readout, we report the best performance among learning rate values {1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 1e1, 1e2}.

For ImageNet-1K and iNaturalist-2018 evaluations, we evaluate the final model after streaming SSL training on the entire dataset. Following Purushwalkam et al. (2022), we train a linear classifier on top of the normalized learned representations and report the classification accuracy. We used a batch size of 1024. For ImageNet-1K, we used the LARS (You et al., 2017) optimizer with learning rate 3.0, momentum 0.9, and cosine learning rate schedule for 10 epochs. For iNaturalist-2018, we used the LARS (You et al., 2017) optimizer with learning rate 12.0, momentum 0.9, and cosine learning rate schedule for 20 epochs.

For OAK evaluations, we use Faster R-CNN (Ren et al., 2015), a popular two-stage object detector. We initialize the ResNet-50 (He et al., 2016) backbone with the backbone of the final checkpoint of the streaming SSL model, and fine-tune the entire model on OAK with IID training for 10 epochs, following the training configurations of (Wu et al., 2023).

B Additional Related Work

Self-Supervised Learning. A large number of self-supervised representation learning methods in computer vision follows the contrastive learning framework (Oord et al., 2018; Misra & Maaten, 2020; Tian et al., 2020; He et al., 2020; Chen et al., 2020; Chen & He, 2021) which maximizes the agreement of representations of two augmented views of the same image and minimizes that of different images. Extending this idea, the supervised contrastive (SupCon) method (Khosla et al., 2020) uses the labels as an extra supervision signal to get multiple positive crops for each anchor image. Other recent self-supervised learning works include pretext tasks (Doersch et al., 2015; Noroozi & Favaro, 2016; Gidaris et al., 2018; Pathak et al., 2016), feature space clustering (Caron et al., 2018; 2020; Ren et al., 2021), distillation with asymmetric architectures (Grill et al., 2020; Chen & He, 2021), redundancy reduction (Zbontar et al., 2021; Bardes et al., 2022), and masked autoencoding (He et al., 2022). Most relevant of these to our work, Orhan et al. (2020) proposes the temporal classification objective, which outperforms contrastive learning objectives on the SAYCam dataset (Sullivan et al., 2021). Our work enhances the temporal classification method by using a more flexible supervised contrastive objective, and leveraging temporal segmentation (Potapov et al., 2014; Afham et al., 2023), which have been used extensively in video summarization (Zhu et al., 2020; Zhang et al., 2016; Rochan et al., 2018).

Temporal Segmentation in Human Cognition. Prior research in psychology and cognitive sciences has shown that humans, including infants, are able to identify boundaries between action segments (Newtson et al., 1977; Zacks et al., 2001; Baldwin et al., 2001; Saylor et al., 2007; Baldassano et al., 2017; Yates et al., 2022). Evidence in neuro-imaging further shows that event segmentation is an automatic component in human perception (Zacks & Swallow, 2007). Temporal event segmentation has proven to be critical for memory formation and retrieval (Lassiter & Slaw, 1991; Ezzyat & Davachi, 2011; DuBrow & Davachi, 2013; Silva et al., 2019; Sasmita & Swallow, 2022). The temporal segmentation component in our proposed framework is motivated by how humans interpret videos as segments with coherent semantics. We demonstrate that temporal segmentation can improve the learned visual representation.

C Additional Results

C.1 Performance of Different Normalization Layers

We experimented with a variation of Memory Storyboard as well as three baseline methods (SimCLR (Chen et al., 2020), Osiris (Zhang et al., 2024), and Temporal Classification (Orhan et al., 2020)) where the group normalization layers in the ResNet backbone are replaced with batch normalization (Ioffe & Szegedy, 2015) layers. The models are trained on SAYCam and evaluated on the downstream *mini*-ImageNet classification task with an SVM. The resulting accuracies are shown in Table 7. We observe that GroupNorm significantly outperforms BatchNorm for all the models examined. This result is aligned with the conclusion in (Zhang et al., 2024) that BatchNorm is less compatible with unsupervised continual learning, and extends the conclusion to streaming SSL.

	SimCLR	Osiris	TC	MemStoryboard
Batch Norm	33.62	33.32	33.16	33.68
Group Norm	**37.96**	**36.90**	**36.68**	**39.58**

Table 7: Group norm is better at dealing with temporal non-stationarity for streaming SSL.

C.2 Separating Short-term Memory Batch and Long-term Memory Batch

Inspired by the design of separating the loss on the new data and the replay data in Osiris (Zhang et al., 2024), we investigate the optimal strategy of applying the temporal contrastive loss on the training batch. We consider applying the temporal contrastive loss only on data from short-term memory, only on data from long-term memory, separately on data from short-term and long-term memory and average the losses, and on the entire training batch (concatenated data from short-term and long-term memory). We report the results in Table 8. For experiments in the main paper, we apply the temporal contrastive loss only on data from long-term memory.

The results here demonstrate that applying the temporal contrastive loss only on data from long-term memory or on the entire training batch achieves the best performance. Applying the temporal contrastive loss only on data from short-term memory achieves inferior performance due to the limited number of temporal classes in the short-term buffer.

	SAYCam		**KrishnaCam**	
	mini-ImageNet	Labeled-S	*mini*-ImageNet	OAK mAP
Short Only	38.54	52.95	34.98	19.53
Long Only	**39.58**	**56.29**	36.36	21.29
Concatenate	38.34	55.43	36.08	21.20
Separate	39.42	54.95	**36.70**	**21.40**

Table 8: Performance of Memory Storyboard when the temporal contrastive loss is applied on different parts of the training batch.

C.3 Memory Storyboard in the IID Setting

To provide more context for the performance of Memory Storyboard in the Streaming Learning setting, we investigate the performance of Memory Storyboard in the IID setting. We take a SimCLR IID pre-trained model and use it to generate temporal segmentations on the entire dataset. Then we assign pseudo-labels to each frame according to the temporal segmentation results and train Memory Storyboard in the IID setting, taking the same number of gradient steps as the streaming setting. We observe that IID Memory Storyboard outperforms IID SimCLR and IID Simsiam, demonstrating the effectiveness of the temporal contrastive loss even in the IID setting.

Method	SAYCam			KrishnaCam		
	mini-INet	INet	iNat	*mini*-INet	INet	iNat
SimCLR MemStoryboard (50K)	39.58	24.78	7.77	36.36	22.75	6.10
SimCLR MemStoryboard (50K)	41.32	26.37	9.85	35.20	22.75	6.71
IID SimCLR	44.04	30.44	8.69	36.90	23.77	5.60
IID SimSiam	29.02	20.92	4.91	28.58	22.28	4.16
IID SimCLR MemStoryboard	44.54	29.98	8.92	37.60	24.43	7.36
IID SimSiam MemStoryboard	42.30	31.02	10.07	36.54	26.60	7.06

Table 9: Performance of Memory Storyboard in the IID Setting.

C.4 Memory Storyboard with the Cross Entropy Loss

One alternative objective for the temporal contrastive loss in Memory Storyboard is the cross entropy (CE) loss. We investigate the performance of Memory Storyboard with the CE loss instead of the Supervised Contrastive (SupCon) loss. Results are summarized in Table 10. We observe that the CE loss outperforms the SupCon loss as the temporal contrastive loss when trained jointly with the SimCLR objective. We opted for the SupCon loss in the main text due to its flexibility. With the CE loss, we either need to know the number of temporal segments beforehand or gradually increase the size of the final classifier layer as the model sees more data, which is not ideal for streaming learning on a never-ending video stream. With the SupCon loss, we can learn from a very large number of temporal segments with the same model size.

Temp. Contrast Loss	SAYCam 10K			SAYCam 50K			KrishnaCam 10K			KrishnaCam 50K		
	mini-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat
SupCon	35.02	20.72	5.65	39.58	24.78	7.77	33.72	20.13	5.64	36.36	22.75	6.10
Cross Entropy	35.58	24.04	6.62	40.06	26.92	8.19	34.44	25.27	5.95	36.26	26.67	6.73

Table 10: Results on Memory Storyboard (using SimCLR as the base SSL method) with the cross entropy-loss instead of the supervised contrastive loss as the temporal contrastive loss.

C.5 Memory Storyboard with only the Temporal Contrastive Loss

To demonstrate the need for both the self-supervised loss and the temporal contrastive loss in our training objective, we experiment with using only the temporal contrastive (SupCon) loss and not the self-supervised loss (only using the self-supervised loss (SimCLR or SimSiam) and not the temporal contrastive loss has been experimented in the "two-tier" buffer baseline in Tables 1 and 2). Results are shown in Table 11. We observe that the performance of only using the SupCon loss is also inferior to the full memory storyboard method, demonstrating the necessity of joint training on both losses for best performance.

Method	SAYCam 10K			SAYCam 50K			KrishnaCam 10K			KrishnaCam 50K		
	mini-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat
SimCLR MemStoryboard	35.02	20.72	5.65	39.58	24.78	7.77	33.72	20.13	5.64	36.36	22.75	6.10
SimSiam MemStoryboard	36.72	22.99	6.66	41.32	26.37	9.85	33.78	21.38	6.51	35.20	22.75	6.71
Supcon Only	34.62	21.04	6.62	39.08	24.92	8.19	31.68	21.29	5.86	34.92	23.07	6.56

Table 11: Results on Memory Storyboard with only the temporal contrastive Loss.

C.6 Memory Storyboard with Multiple Gradient Steps per Batch

Using multiple gradient steps for each batch is a widely used technique in online continual learning Madaan et al. (2022). We investigate the performance of Memory Storyboard when we take multiple gradient steps on each batch. Results are shown in Table 12. We observe that using multiple gradient steps (2 or 4) produces a sizable improvement on the ImageNet readout evaluation on KrishnaCam but not on the other benchmarks. We also observed that the improvement of multiple gradient steps is a lot smaller on SAYCam (sometimes even harming the performance), presumably due to the fact that SAYCam is a much larger training dataset than KrishnaCam and streaming learning without multiple gradient steps is sufficient for the model to capture a wide range of visual concepts.

Method	Grad Steps	SAYCam 10K			SAYCam 50K			KrishnaCam 10K			KrishnaCam 50K		
		mini-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat
SimCLR MemStoryboard	1	35.02	20.72	5.65	39.58	24.78	7.77	33.72	20.13	5.64	36.36	22.75	6.10
SimCLR MemStoryboard	2	34.56	23.47	5.66	38.44	26.49	7.67	33.00	23.86	5.45	35.30	26.36	6.38
SimCLR MemStoryboard	4	33.26	22.08	4.21	35.78	25.64	5.96	32.40	23.19	5.61	35.34	26.11	6.38
SimSiam MemStoryboard	1	36.72	22.99	6.66	41.32	26.37	9.85	33.78	21.38	6.51	35.20	22.75	6.71
SimSiam MemStoryboard	2	37.04	26.89	7.13	40.20	30.29	9.27	33.86	25.22	6.27	35.66	26.21	6.85
SimSiam MemStoryboard	4	35.02	22.88	6.73	36.30	25.21	7.36	33.44	24.34	6.44	34.98	25.73	6.62

Table 12: Results on Memory Storyboard with different number of gradient update steps per batch.

C.7 Memory Storyboard with Class-Balanced Buffer

Inspired by other methods with use smart memory storage policies (Yu et al., 2023; Purushwalkam et al., 2022), we investigate the performance of Memory Storyboard with a class-balanced memory. When we attempt to add a new data point to the long-term memory that is already full, we randomly remove one of the data points from the class with the most samples in the memory. Results are shown in Table 13. We observe that using the class-balanced memory produces mild improvements over the reservoir sampling baseline, though results on specific runs are mixed. We think that the memory storyboard method should work well with many different buffer sampling strategies, and advancements in buffer sampling strategies are orthogonal to the contribution of this work.

Base SSL Method	Bal. Buffer	SAYCam 10K			SAYCam 50K			KrishnaCam 10K			KrishnaCam 50K		
		mini-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat	*mini*-INet	INet	iNat
SimCLR	✗	35.02	20.72	5.65	39.58	24.78	7.77	33.72	20.13	5.64	36.36	22.75	6.10
SimCLR	✓	34.04	22.21	6.41	38.58	26.99	7.58	31.92	21.28	5.70	34.66	23.64	5.78
SimSiam	✗	36.72	22.99	6.66	41.32	26.37	9.85	33.78	21.38	6.51	35.20	22.75	6.71
SimSiam	✓	34.66	23.02	6.33	37.20	24.69	7.37	33.86	22.67	6.32	36.76	25.61	7.09

Table 13: Results on Memory Storyboard with class-balanced buffer.

C.8 Additional Qualitative Results

OAK Object Detection Results. We visualize the object detection results produced by Memory Storyboard when fine-tuned on the OAK dataset (Wang et al., 2021) in Figure 5. We observe that the fine-tuned model can successfully detect objects in cluttered environments. The results show that the representations learned by Memory Storyboard can be effectively transferred to downstream tasks which requires more fine-grained features.

Label Merging Results. We visualize the class labels produced by the label merging mechanism in Memory Storyboard in Figure 7. We observe that Memory Storyboard can successfully group semantically similar scenes together, which helps improve the representation learning performance.

Temporal Segmentation by Randomly Initialized Models. We visualize the temporal segments produced by randomly initialized models in Figure 6. By comparing to Figure 3, we observe that randomly initialized models fail to capture intricate transitions between scenes and cannot create accurate temporal segments, while Memory Storyboard training enables the model to learn better image representations to capture more intricate scene transitions.

D Optimal Batch Composition for SimCLR

We replicate the experiments in Figure 4 on SimCLR models with two-tier memory, and plot the results in Figure 8. We observe that the analysis and the conclusions of section 5.4 still hold: when we have a large memory, we either prefer balanced training batches (with a fixed amount of computation) or a bigger batch from long-term memory (with a fixed amount of data); when we can only afford a small memory, we prefer a smaller batch from long-term memory. We also want to note that the SVM readout results start to go down towards the end of the streaming training in SimCLR experiments more often than Memory Storyboard experiments, suggesting the better scalability of Memory Storyboard to larger-scale streaming training.

These results demonstrate that the analysis and observations in section 5.4 regarding the optimal batch composition for streaming SSL training under different memory and compute constraints are general, and apply to standard SSL methods in addition to Memory Storyboard.

Figure 5: **Visualization of object detection results on the OAK validation set.** The Memory Storyboard model is trained on KrishnaCam and fine-tuned on the OAK training set. Red boxes show the predictions and the green boxes are ground truth bounding boxes.

E More Comprehensive Comparison between Memory Storyboard and SimCLR

With the experiment results in Figure 4 and Figure 8, we provide a more comprehensive comparison between Memory Storyboard and SimCLR performance under different memory constraints and batch compositions in Figure 9. We observe that Memory Storyboard outperforms SimCLR under the same amount of seen data, across a wide range of memory sizes and batch compositions. In particular, we note that Memory Storyboard significantly outperforms SimCLR when we sample more data from M_{short} (towards the right side of the x-axis). This results in higher optimal performance when the memory size is small, where a larger batch from M_{short} is needed to prevent overfitting on the long-term memory for better performance. We argue that, with temporal segmentation and the temporal contrastive loss, Memory Storyboard is able to provide better memory efficiency and also alleviate the temporal correlation issue suffered by SimCLR when we sample a large batch from the short-term memory.

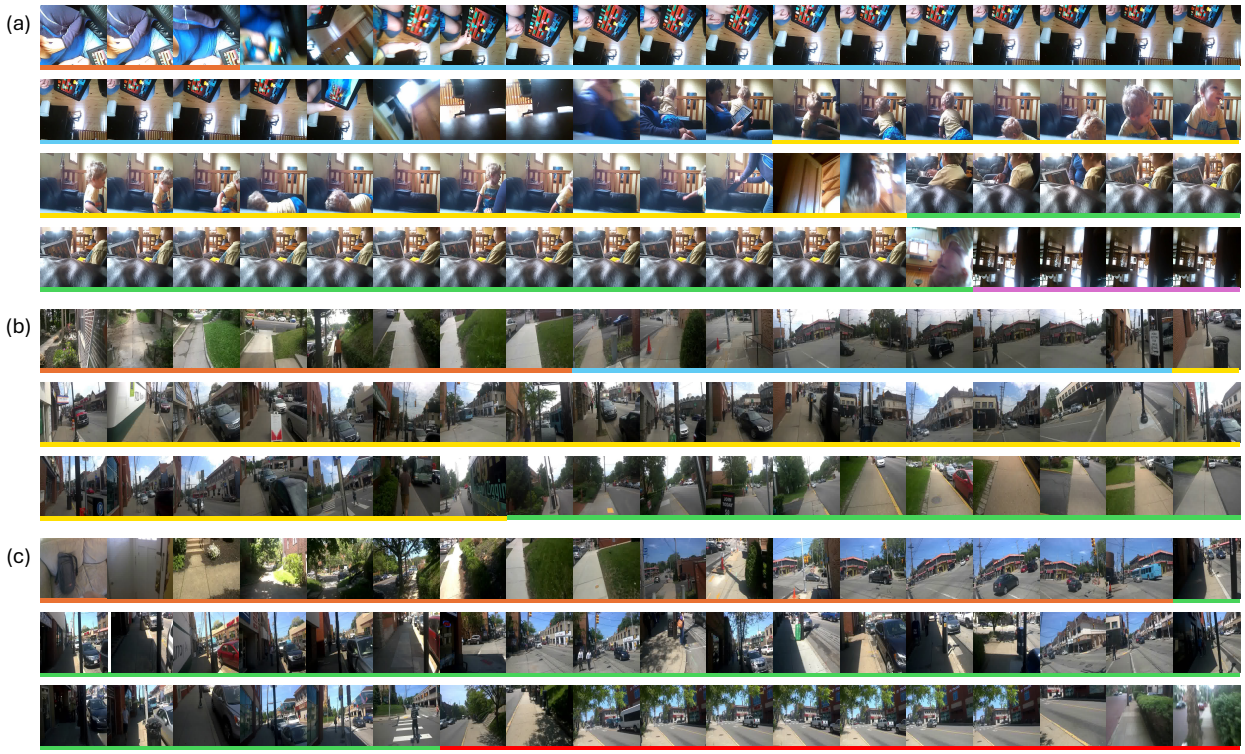

Figure 6: **Visualization of the temporal segments produced by randomly initialized models on (a) SAYCam (b)(c) KrishnaCam.** The images are the same as the ones in Figure 3. We observe that Memory Storyboard training enables to model to capture more intricate transitions between scenes.

Figure 7: **Visualization of label merging by Memory Storyboard on SAYCam.** Each image represents a temporal segment; segments sharing the same color bar have been merged. Memory Storyboard successfully groups semantically similar scenes—e.g., segments marked with the light blue bar are all associated with the dining table.

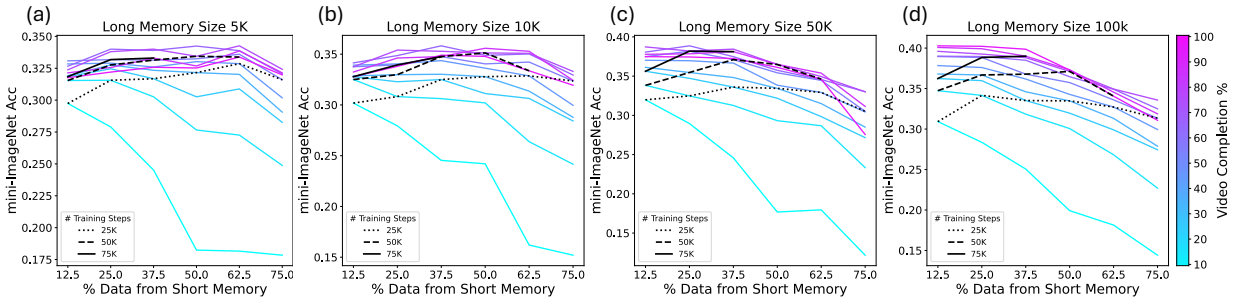

Figure 8: **SimCLR model performance on SAYCam with different long-term memory sizes** (5k, 10k, 50k, and 100k) **and varying training batch compositions** (12.5% – 75.0% from M_{short}) using SVM readout. Each colored line represents the performance of different training batch compositions when **the model has seen the same amount of data** from the stream. Each black line represents the performance of different training batch compositions when the model has taken **the same number of gradient updates**.

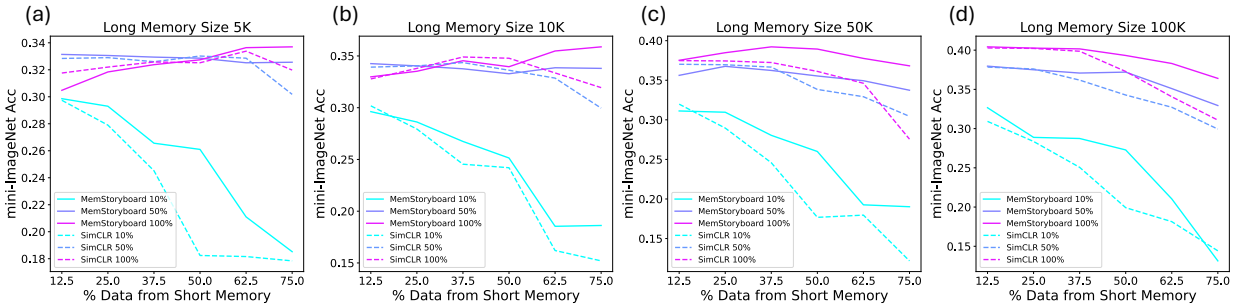

Figure 9: **Comparison of Memory Storyboard (solid lines) and SimCLR (dashed lines) model performance** on SAYCam using SVM readout, controlling the amount of data the model has seen from the stream.