Stream-51: Streaming Classification and Novelty Detection from Videos

Ryne Roady[1,*] Tyler L. Hayes[1,*] Hitesh Vaidya[1] Christopher Kanan[1,2,3]
[1]Rochester Institute of Technology [2]Paige [3]Cornell Tech
{rpr3697,tlh6792,hv8322,kanan}@rit.edu

Abstract

Deep neural networks are popular for visual perception tasks such as image classification and object detection. Once trained and deployed in a real-time environment, these models struggle to identify novel inputs not initially represented in the training distribution. Further, they cannot be easily updated on new information or they will catastrophically forget previously learned knowledge. While there has been much interest in developing models capable of overcoming forgetting, most research has focused on incrementally learning from common image classification datasets broken up into large batches. Online streaming learning is a more realistic paradigm where a model must learn one sample at a time from temporally correlated data streams. Although there are a few datasets designed specifically for this protocol, most have limitations such as few classes or poor image quality. In this work, we introduce Stream-51, a new dataset for streaming classification consisting of temporally correlated images from 51 distinct object categories and additional evaluation classes outside of the training distribution to test novelty recognition. We establish unique evaluation protocols, experimental metrics, and baselines for our dataset in the streaming paradigm[1].

Figure 1: Popular training paradigms for incremental learners. Stream-51 requires agents to learn from real-time video streams of natural scenes.

1. Introduction

Agents operating in real-time environments must be capable of learning from dynamic, open-ended data streams. Further, agents should be able to recognize novel, unknown concepts and learn from new information, while retaining knowledge of previous tasks. Deep neural networks (DNNs) are a dominant approach for perception tasks, but when trained on changing, non independent and identically distributed (iid) data distributions, they suffer from catastrophic forgetting of previous knowledge [30]. Moreover, they struggle to identify unlearned concepts as novel on large-scale datasets [37], which is an important skill for safety critical applications such as self-driving cars.

Recently, much research has focused on overcoming catastrophic forgetting in the *incremental batch learning* paradigm [6, 7, 18, 21, 22, 26, 36, 41, 42], where DNNs are updated incrementally with new information, but the agent is allowed to batch through data and can only be evaluated after it has finished learning the previous batch. This scenario is not realistic for embedded agents that are deployed for long periods of time and must be tested on new information immediately. Online learning in a single pass with severe memory and compute constraints, also called *streaming learning* [2, 5, 10, 13, 14, 20, 35], more closely matches how embedded agents must learn and make inferences in real-time. However, streaming learners still suffer from catastrophic forgetting when

[*]denotes equal contribution.
[1]https://github.com/tyler-hayes/Stream-51

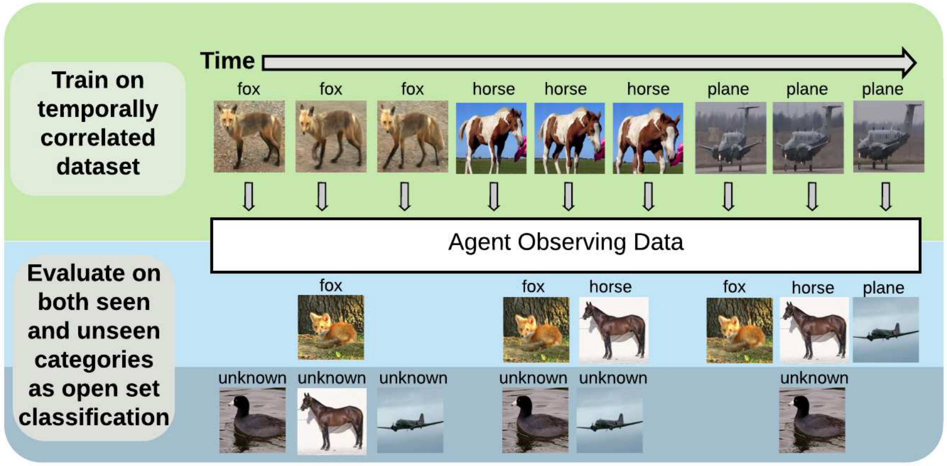

Figure 2: The Stream-51 protocol poses unique challenges, requiring agents to learn from temporally correlated data streams and recognize unlearned concepts as novel. Training data can be ordered either just by instance or by both class and instance. Evaluation data includes a set of novel examples from classes unseen during training.

trained on non-stationary data distributions, which is a result of the stability-plasticity dilemma [1].

While streaming learning is more amenable to real-time learning, part of the limitation in developing models in this paradigm has been the lack of large-scale, many-class datasets with temporal correlations. This lack of diversity makes it difficult to test the generalization capabilities of learners. Further, there are a lack of explicit protocols designed to test the novelty detection capabilities of learners. Here, we introduce the Stream-51 dataset for streaming image classification. Stream-51 consists of temporally correlated videos, which resemble how humans, animals, and other embedded agents receive information in the real-world. Moreover, Stream-51's evaluation protocol tests an agent's ability to learn new concepts and identify novel samples from classes not seen during training.

This paper makes the following contributions:

1. We introduce a large-scale streaming dataset with training instances drawn from 51 unique classes and an evaluation set containing images from the in distribution classes, as well as 43 classes outside of the training distribution for novelty detection.
2. We establish strong classification baselines on our dataset using two different data ordering scenarios known to induce catastrophic forgetting in DNNs.
3. We establish new protocols, baselines, and metrics to test a streaming agent's ability to identify novel inputs, making it easy to compare existing methods.

2. Problem Formulation

In incremental batch learning, an agent is required to learn a dataset $\mathcal{D}$, that is broken up into T distinct batches B_t, each of size N_t. At time t, the learner only has access to B_t, but it may loop over this batch as many times as necessary and can only be tested after the batch has been learned. Conversely, in *streaming learning*, an agent is required to learn examples one at a time ($N_t = 1$) from only a single pass through the labeled dataset. This paradigm is advantageous in developing embedded agents since the agent can be evaluated at any point during training and it cannot infinitely loop over any portion of the dataset. Here, we focus on comparing streaming learners that must correctly classify previously learned classes, while also identifying inputs that are outside of the agent's learned distribution, i.e., novelty detection or open set classification (see Fig. 2).

3. Related Work

Several evaluation paradigms have been proposed for developing agents capable of mitigating catastrophic forgetting, however, many are not applicable for real-time agents that learn from temporally correlated data. Additionally, there are few existing video datasets that are ideal for testing streaming learners and none containing explicit protocols for novelty detection.

3.1. Existing Incremental Learning Evaluations

Two popular evaluation schemes for incremental batch learning are the Permuted MNIST [22, 23] and Split

Table 1: Streaming dataset statistics including information about the videos and their acquisition (acq).

DATASET	CLASSES	IMAGES	VIDEOS	VIDEOS/ CLASS	AVG FRAMES/ VIDEO	ACQ
iCub-1 [12]	10	8,000	40	4	200	hand held
iCub-T [33]	20	400,000	2,000	100	200	hand held
CORe50 [28]	10	165,000	550	55	300	hand held
ToyBox [40]	12	2,300,000	540	45	4,200	hand held
Stream-51	51	150,736	1,136	11-37	132.69	natural/wild

MNIST [7, 22] experiments. In Permuted MNIST, each task consists of a different, but fixed, permutation of the 784 image pixels and the agent must learn to classify the permuted digits given the permutation label. This approach can only be used to evaluate agents with fully-connected layers, the task (permutation) label is required at test time, and this paradigm is equivalent to scrambling up the spatial input space of an agent and requiring it to perform classification, which is unrealistic. In Split MNIST, the MNIST dataset is split into disjoint groups (tasks) of classes and the agent must learn the groups incrementally. While Split MNIST is closer to how animals learn than Permuted MNIST, things that work well on MNIST usually do not scale up to larger datasets [22]. Similar to Split MNIST, other popular evaluation schemes include Split CUB-200 [8, 21, 22], Split CIFAR-100 [6, 7, 18, 21, 36, 41], and Split ImageNet [6, 18, 36, 41]. While these paradigms contain more classes and natural imagery than MNIST, they are still unrealistic for agents operating in real-time because 1) there is no inherent temporal dependence between frames, 2) classes are never revisited once they are learned, and 3) agents are provided large batches of data for each task, where batches are naturally iid. Examples of several popular training paradigms are shown in Fig. 1.

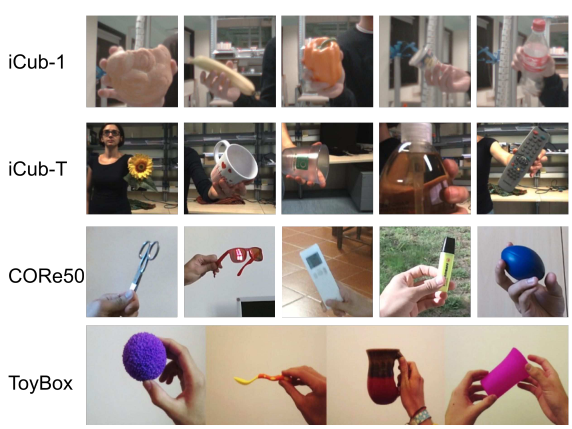

Figure 3: Example images from comparison datasets.

3.2. Existing Streaming Datasets

An ideal streaming classification dataset would consist of temporally correlated videos of objects from a large variety of classes. Dataset statistics for the most well-known streaming datatsets are in Table 1 and example images are in Fig. 3. Some of the earliest streaming datasets were collected from the iCub robot including iCub World 1.0 (iCub-1) [12] and iCub World Transformations (iCub-T) [33]. However, both datasets contain 20 or fewer classes. More recent datasets include CORe50 and ToyBox, but they only have 10 and 12 object categories, respectively. All of the aforementioned datasets are limited to 20 or fewer classes and all were collected by having a person move each object around with their hand. Additionally, the datasets are too small and too easy to adequately test an agent's generalization capabilities. Similarly, there is not enough standardization among the datasets, i.e., researchers use different evaluation paradigms and metrics across datasets making it difficult to compare approaches.

Although there are many video datasets from the object detection [38] and tracking [11, 19] communities, they cannot be used naturally for streaming image classification. Object tracking datasets often contain objects that take up a small portion of the image frame, making it hard to identify objects for classification purposes. Similarly, tracking datasets often have many objects within a frame that are not mutually exclusive, which is necessary for standard classification tasks. In 2015 the ILSVRC Object Detection from Video (VID) dataset [38] was introduced, which contains video sequences of up to 3 unique objects per frame, but it is limited to only 30 total classes. Moreover, none of the aforementioned streaming, object detection, or tracking datasets have evaluation samples that are explicitly outside of the training set classes for testing novelty detection capabilities.

4. Stream-51

Stream-51 is a large-scale image classification dataset with training images drawn from videos to mimic the

Figure 4: Example images from each of the 51 classes in Stream-51.

way real-time agents would experience new objects. It is significantly larger than existing streaming classification datasets with 51 classes drawn from familiar animal and vehicle object classes. The temporal correlation between subsequent frames is difficult for DNNs, which traditionally assume that data are sampled iid during training. Additionally, the Stream-51 test set contains samples from classes not included in the training distribution to test a model's novelty detection capabilities.

4.1. Curation Process

4.1.1 Downloading Object Detection and Tracking Datasets

Stream-51 was curated from a variety of existing, video-based object detection and tracking datasets including: the Generic Object Tracking (GOT-10K) dataset [19], the VID dataset [38], and the Large Single Object Tracking (LaSOT) dataset [11]. The goal in combining snippets from each of these datasets was to maximize the number of independent categories with a sufficient number of unique videos and overall frames per class. The GOT-10K dataset served as the main source of images (46.1% of overall frames). While it has 563 unique classes, many of these classes did not have enough total video sequences per class to curate a robust dataset. For these reasons, we also used the major-class labels which assign each image to one of 115 super classes. Overall, GOT-10K provided data for 34 of the 51 classes. The VID dataset served as the second major source of video frames (27.3% of the total) and provided us with 13 additional unique classes. Finally, we used videos from the LaSOT dataset (26.6% of the total), which supplied 4 unique classes. There were many instances of class overlap among the source datasets, which allowed us to increase the total number of videos for those classes. All videos for overlapping classes were verified to be unique.

4.1.2 Filtering Training Videos

At this point in curation, the raw frames from the source datasets are not useful for training an image classification model. One problem is that many of the full frames contain multiple objects often from multiple classes, which creates too much label noise. A second problem is the resolution of the classes of interest in the full frame videos varies too widely, often with the object of interest containing very few pixels in the full frame. To overcome these limitations, we filtered the images using bounding box information included in the source dataset annotations.

Since typical convolutional neural networks (CNNs) require moderately sized images (e.g., 224×224 for the ResNet architectures [16]), we limited the resolution of the bounding boxes to cover at least an area of 1024 pixels (∼32×32 image). Frames which didn't meet this threshold were removed and the videos were divided into shorter, temporally coherent clips. This bounding box threshold was found to be a good trade off to ensure adequate resolution of the object, but not too limiting to exclude large portions of the underlying videos.

When generating Stream-51 from the underlying object tracking datasets, we also limited the length of individual snippets to no more than 300 frames per video clip and no fewer than 50 frames (sampled at approximately 10 fps). When videos from the underlying datasets were longer than these limits, we broke the longer video up into smaller sub-clips within the limits. If video clips were shorter than the minimum length, we discarded the clips. We first filled every class with the highest resolution unique videos and then supplemented with non-unique clips as needed. We limited each class to have no more than ∼3,000 total frames per class with all clips ranging from 50 to 300 frames.

Determining the final class list for Stream-51 involved first building a larger list of possible independent classes from the datasets above. The list of classes was then selected by reducing semantic overlap using the Wu-Palmer Similarity metric [29], which computes the relatedness of two words using WordNet. Stream-51 statistics are provided in Table 1. Overall, the average clip length in the training dataset is 133 frames and the average resolution is 0.3 megapixels (∼550×550).

4.1.3 Curating the Test Set

Streaming learning requires agents to learn categories from temporally correlated data, however, we also desire to evaluate the generalization capabilities of these agents. For this reason, we curated a distinct set of static images for each class to use for evaluation. This allowed us to maximize the total number of videos available for the training set and to make a larger, more diverse, test set. To curate the test set, we used well-known static image datasets that contained at least 50 unique images for each category in our training set. We used 21 classes from the ImageNet object detection dataset [38], 18 classes from the ImageNet classification dataset [38], and 12 classes from OpenImages V5 [24]. We then added ∼60 unique images from 43 additional categories not represented in the training set to serve as a source for evaluating novelty detection. 23 of these novel classes were from the ImageNet object detection dataset, 11 were from the ImageNet classification dataset, and 9 were from the OpenImages V5 dataset. In total there are 5,100 total images in the evaluation set (50 samples per training class and 2,550 novel samples).

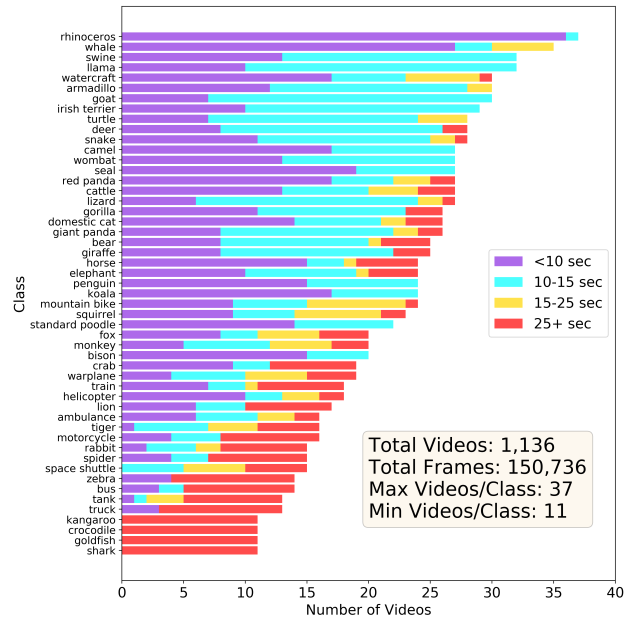

Figure 5: Per class video statistics for Stream-51. Colors denote counts of the number of clips with various lengths in seconds. Best viewed in color.

4.2. What's in Stream-51?

Stream-51 has a wide variety of object categories with 41 animal classes and 10 vehicle classes under various environmental conditions, e.g., indoor scenes and outdoor scenes like desert, water, and sky. Example images from each class in the training set are shown in Fig. 4. Fig. 5 shows the number of unique videos per class in Stream-51 and each video's respective length in seconds.

5. Baseline Experimental Protocol

We train models to predict the category $\hat{y}_{t,\text{classification}}$:

$$\hat{y}_{t,\text{classification}} = \arg\max_{k} F(G(\mathbf{X}_t, k)) \quad , \qquad (1)$$

where $k \in K$ is the class label from K possible labels, $\mathbf{X}_t$ is the input at time t, $G(\cdot)$ consists of the first L layers of the neural network with parameters θ_G, and $F(\cdot)$ consists of the last fully-connected layer of the network with parameters θ_F. We distinguish between two types of streaming learning algorithms: 1) those that only train the top of the network $F(\cdot)$, which can be thought of as a decoder, and 2) those that train the entire network $F(G(\cdot))$, where the function $G(\cdot)$ can be thought of as an encoder consisting of the lower layers of the network. Given an input tensor $\mathbf{X}_t$ at time t, the output of the encoder is given by $\mathbf{z}_t = G(\mathbf{X}_t)$, where $\mathbf{z}_t \in \mathbb{R}^d$ represents the d-dimensional embedding of the input tensor. The class decoder $F(\mathbf{z}_t)$ outputs a K-dimensional vector used to predict $\hat{y}_{t,\mathrm{classification}}$.

In addition to being able to correctly classify inputs, a critical skill for an agent is to recognize when a test input is outside of its learned categories. Traditional closed-set models do not have this capability and instead assign a label of one of the learned categories to novel inputs. The ability of an agent to recognize inputs outside of its training distribution could facilitate automatic class discovery [39, 44] or open world learning [3].

Formally, an agent must learn a classifier $H(\mathbf{X}_t) = F(G(\mathbf{X}_t))$ such that it can be used to distinguish learned inputs from novel inputs, i.e.,

$$\hat{y}_{t,\mathrm{novel}} = \begin{cases} 1 & \text{if } S\left(H(\mathbf{X}_t)\right) \geq \delta \\ 0 & \text{if } S\left(H(\mathbf{X}_t)\right) < \delta \end{cases}, \quad (2)$$

where $S(\cdot)$ is an acceptance score function that uses a threshold δ to determine if an input belongs to the training distribution. We use the confidence thresholding algorithm [17] for computing $S(\cdot)$, which is a simple approach to detecting novel inputs. It determines a threshold for the softmax probabilities output by a model based on correctly classified training inputs. This approach assumes that samples from the known classes seen during training will have much larger maximum class probability than novel inputs. Our protocol is shown in Fig. 2.

5.1. Baseline Models Evaluated

Our effort is on comparing streaming methods, and not CNN architectures, so all methods use the same CNN architecture (ResNet-18 [16]). However, any architecture can be used with Stream-51. We evaluate the following:

- **Fine-Tuning (No Buffer)** – This model trains the CNN one example at a time with a single epoch. Since the model does not have a buffer for replay, it suffers from catastrophic forgetting.
- **SLDA** – Streaming Linear Discriminant Analysis is a popular model in the data mining community and it was recently shown to work well on deep CNN features [15]. SLDA updates a running mean vector per class and a shared running covariance matrix. To make predictions, it assigns to an input the label of the closest Gaussian computed using the means and covariance matrix. SLDA is one way to update the output layer of the CNN (θ_F).
- **ExStream** – The Exemplar Streaming algorithm was proposed for updating the fully-connected layers of a neural network and achieved state-of-the-art performance on the iCub-1 and CORe50 streaming datasets [14]. ExStream uses a form of partial rehearsal to mitigate forgetting by storing a buffer of features for exemplars to replay during later training sessions. The method stores the incoming features and merges the two closest exemplars in its buffer. It then replays all examples stored for a single iteration and uses stochastic gradient descent to update the weights in fully-connected layers (θ_F).
- **Full Rehearsal** – Full rehearsal is a baseline that uses replay mechanisms to mitigate forgetting and was shown to work well in [14]. Full rehearsal stores *all* input examples in a memory buffer and fine-tunes the CNN on all previous examples, which is expensive in terms of memory and compute.
- **Offline** – The offline model serves as an upper bound. Offline uses the traditional procedure for updating a CNN by training on all previous data with multiple epochs through the dataset. It is not trained incrementally and is initialized from scratch.

We evaluate two versions of fine-tuning, full rehearsal, and offline: 1) update only the output layer (θ_F) and 2) update the entire network (θ_F and θ_G). This setup mirrors how neural networks, especially CNNs, are used in practice for many machine learning applications. For it to work successfully, the parameters of $G(\cdot)$ must already have established representations that will enable $F(\cdot)$ to perform well. For image classification, a common approach is for $G(\cdot)$ to be pre-trained on a large image classification dataset (e.g., ImageNet), and then either $F(\cdot)$ is updated alone or both the encoder and decoder are jointly updated. Another approach is to use self-taught learning to train $G(\cdot)$ on another dataset [34].

5.2. Dataset Orderings

Since the temporal structure of the training stream affects a learner's performance, we assess all models on the two most realistic data ordering scenarios given in [14]: instance where videos are temporally ordered by object instances and class instance where videos are organized by class, but organized by temporal video instances within each class. Both orderings are similar to how humans perceive data streams and are known to induce catastrophic forgetting in DNNs.

5.3. Performance Evaluation

We use two metrics: one to capture an agent's overall classification performance and one to capture its ability to detect novel inputs, while still correctly classifying in-distribution samples. Embedded agents operating for long periods of time must have low memory and compute costs, so we also report memory and time requirements.

For overall classification performance, we use the

$\Omega_{\text{Classif.}}$ metric from [14] which is computed as:

$$\Omega_{\text{Classif.}} = \min\left(1, \frac{1}{T}\sum_{t=1}^{T}\frac{\alpha_t}{\alpha_{\text{offline,t}}}\right) \quad , \tag{3}$$

where T is the total number of testing events, α_t is the accuracy of the streaming learner at time t, and $\alpha_{\text{offline},t}$ is the accuracy of an optimized offline model at time t. This metric normalizes a streaming learner's performance to an optimized offline learner and measures how well an agent is able to classify inputs. Normalizing the streaming learner's performance to an offline learner makes the metric easier to interpret across various orderings.

For novelty detection, we propose an incremental variant of the open set classification curve (OSC) metric [9], which has been used for offline open set recognition. The OSC metric computes the correct classification rate among known classes as a function of the false positive rate for distinguishing between seen and novel categories. The resulting correct classification rate is the difference in model accuracy and the false negative rate for novelty detection. The OSC metric is more informative than a traditional ROC curve for detecting novel classes since it accounts for the correct classification of true positive samples. That is, OSC rewards methods that reject incorrectly classified positive samples more than methods that reject correctly classified samples. Formally, we propose an incremental variant of the area under the OSC curve (AUOSC) which normalizes an incremental learner's performance to an optimized offline baseline, i.e.,

$$\Omega_{\text{AUOSC}} = \min\left(1, \frac{1}{T}\sum_{t=1}^{T}\frac{\gamma_t}{\gamma_{\text{offline,t}}}\right) \quad , \tag{4}$$

where T is the total number of testing events, γ_t is the AUOSC score of the incremental learner at time t, and $\gamma_{\text{offline,t}}$ is the AUOSC score of the optimized offline learner at time t. The explicit equation for γ can be found in [9] and computes performance based on novelty detection capabilities, as well as correct classification of in-distribution samples. The Ω_{AUOSC} metric tests two capabilities: 1) the agent's ability to identify inputs that are outside of its training distribution and 2) its ability to correctly classify inputs identified as belonging to its training distribution. Ω_{AUOSC} of 1 indicates that the incremental learner performed as well as the offline learner.

5.4. Experimental Setup

In many applications, it has become common practice to initialize the parameters of a CNN on the large-scale ImageNet classification dataset before training on another dataset. However, many of the categories in the Stream-51 training set and classes in the novelty detection test set overlap with ImageNet, so for our baselines, we initialize $G(\cdot)$ using pre-trained weights on the Places-365 dataset [43]. Places-365 consists of 1.8 million training images of 365 different scene-based categories. We suggest using an initialization dataset without overlap since a classifier pre-trained on any of the Stream-51 classes would already have rich features for those classes and then the true learning and novelty detection capabilities of the streaming learner would not be tested thoroughly.

After initialization, ordered examples from Stream-51 are input into the model one at a time. For the instance ordering, classification performance is computed on all classes in the training set after every 30,000 examples have been learned. For the class instance ordering, classification performance is computed on only the classes trained after every 10 classes have been learned. We report $\Omega_{\text{Classif.}}$ with top-1 accuracy.

For novelty detection, we evaluate the ability of an agent to identify classes on which it has not yet been trained, as well as samples entirely outside of its training distribution. Recognizing unseen classes as novel has been common practice, often under the label of open set recognition [4, 31, 32], which is a difficult task [27]. Lifelong novelty detection is a critical step towards automatic class discovery [39, 44] and open world learning [3].

We perform novelty detection experiments using the class instance ordering of Stream-51. Similar to the classification experiments, we initialize $G(\cdot)$ with pre-trained Places-365 weights. We then stream examples into the network one at a time and evaluate the model after every 10 classes. For the novelty detection experiments, the agent is required to determine if a sample is in-distribution or out-of-distribution from the training set, and if the sample is in-distribution, then the agent must correctly classify it. For the in-distribution test set, we select all images from previously learned classes in the test set. For the out-of-distribution test set, we select all test images of unseen classes and combine them with the 2,550 test images explicitly outside of the training set.

6. Baseline Results

$\Omega_{\text{Classif.}}$ and Ω_{AUOSC} are normalized to an offline learner that achieves 76.9% final accuracy and 0.710 final AUOSC respectively on the class instance ordering of Stream-51. For all models except SLDA, we use stochastic gradient descent with momentum of 0.9, learning rate of 0.01, weight decay of 1e-4, and batch size of 256. SLDA uses shrinkage of 1e-4. ExStream stores 50 clusters per class. Offline and full rehearsal are trained for

Table 2: $\Omega_{\text{Classif.}}$ and Ω_{AUOSC} results on Stream-51. We report the amount of memory required *beyond* the CNN for each model in MB and the run time in seconds as the average over all runs. We denote the plastic (plas.) portion of the network for each model. Results are the average of three runs with different permutations of the Stream-51 orderings.

		INST	CLS INST			
METHOD	PLAS.	$\Omega_{\text{Classif.}}$	$\Omega_{\text{Classif.}}$	Ω_{AUOSC}	MEMORY	TIME
Fine-Tune	θ_F	0.422	0.066	0.051	**0.00**	498
Fine-Tune	θ_F, θ_G	0.030	0.050	0.022	**0.00**	2242
SLDA [15]	θ_F	0.856	0.865	0.661	1.05	**485**
ExStream [14]	θ_F	0.829	0.825	0.721	5.22	2039
Full Rehearsal	θ_F	0.818	0.846	0.777	77.77	9855
Full Rehearsal	θ_F, θ_G	**0.952**	**0.953**	**0.941**	22865	11970
Offline	θ_F	0.806	0.835	0.771	77.77	12363
Offline	θ_F, θ_G	1.000	1.000	1.000	22865	11652

10 epochs on data batches. For all models, we use the bounding box crops and resize the images to 224×224.

6.1. Streaming Classification

Our main results for all orderings of Stream-51 from Places-365 pre-trained weights are summarized in Table 2. It is not surprising that the full rehearsal models perform well since they store all previous data for replay, however, these models are memory and computationally expensive to train. The more lightweight SLDA model is a top performer for both orderings since its independent class means allow it to remain robust to forgetting. ExStream also performed relatively well for both orderings. In general, the models that were fine-tuned without a buffer performed poorly since they did not have any mechanisms to mitigate forgetting.

6.2. Streaming Novelty Detection

A learning curve for AUOSC performance as a function of number of classes trained is in Fig. 6. The full rehearsal model was the top performer for the novelty detection experiment, but it is slow to train and memory intensive. ExStream was a top performer for the novelty experiment, while being more efficient than full rehearsal. Although SLDA was a top performer for streaming classification, it did not perform as well as other methods for detecting novel samples. In general, methods which rely on fixed representations and train only the classification layer of a model performed closest to the offline model, however, even offline performance for detecting novel samples is poor due to the simplistic baseline method used. More sophisticated techniques for detecting novel inputs have been developed in recent years [4, 25, 27], however, adapting these techniques to streaming models remains an area for future research.

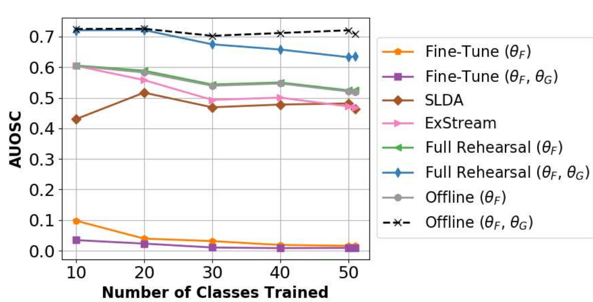

Figure 6: AUOSC Learning Curve.

7. Conclusion

We introduced the Stream-51 dataset for learning from temporally ordered videos collected from natural environments that mimic how humans perceive data. Stream-51 contains more classes and unique videos than existing datasets, making it ideal for developing streaming agents. We also provided baselines and experimental metrics for standard classification and novelty detection tasks. Both of these tasks are needed by agents learning in real-time. The SLDA and ExStream baselines provide a good starting point for Stream-51 demonstrating a compromise between classification performance, novelty detection, memory, and computational requirements.

Acknowledgments. This work was supported in part by NSF award #1909696, the DARPA/MTO Lifelong Learning Machines program [W911NF-18-2-0263], and AFOSR grant [FA9550-18-1-0121]. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies or endorsements of any sponsor.

References

[1] Wickliffe C Abraham and Anthony Robins. Memory retention–the synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 2005. 2

[2] Charu C Aggarwal, Jiawei Han, Jianyong Wang, and Philip S Yu. On demand classification of data streams. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 503–508. ACM, 2004. 1

[3] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, pages 1893–1902, 2015. 6, 7

[4] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *CVPR*, pages 1563–1572, 2016. 7, 8

[5] Albert Bifet and Ricard Gavaldà. Adaptive learning from evolving data streams. In *International Symposium on Intelligent Data Analysis*, pages 249–260. Springer, 2009. 1

[6] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, pages 233–248, 2018. 1, 3

[7] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, pages 532–547, 2018. 1, 3

[8] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *ICLR*, 2019. 3

[9] Akshay Raj Dhamija, Manuel Günther, and Terrance Boult. Reducing network agnostophobia. In *NeurIPS*, pages 9157–9168, 2018. 7

[10] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 71–80. ACM, 2000. 1

[11] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, pages 5374–5383, 2019. 3, 4

[12] Sean Fanello, Carlo Ciliberto, Matteo Santoro, Lorenzo Natale, Giorgio Metta, Lorenzo Rosasco, and Francesca Odone. iCub World: Friendly robots help building good vision data-sets. In *CVPR-W*, pages 700–705, 2013. 3

[13] Mohamed Medhat Gaber, Arkady Zaslavsky, and Shonali Krishnaswamy. A survey of classification methods in data streams. In *Data Streams*, pages 39–59. Springer, 2007. 1

[14] Tyler L Hayes, Nathan D Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *ICRA*, pages 9769–9776, 2019. 1, 6, 7, 8

[15] Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *CVPR-W*, 2020. 6, 8

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 6

[17] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. 6

[18] Saihui Hou, Xinyu Pan, Zilei Wang, Chen Change Loy, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019. 1, 3

[19] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018. 3, 4

[20] Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 97–106. ACM, 2001. 1

[21] Ronald Kemker and Christopher Kanan. FearNet: Brain-inspired model for incremental learning. In *ICLR*, 2018. 1, 3

[22] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *AAAI*, pages 3390–3398, 2018. 1, 2, 3

[23] James Kirkpatrick, Razvan Pascanu, Neil 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*, pages 3521–3526, 2017. 2

[24] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 5

[25] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, pages 7167–7177, 2018. 8

[26] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *ECCV*, pages 614–629. Springer, 2016. 1

[27] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018. 7, 8

[28] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, pages 17–26, 2017. 3

[29] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *NeurIPS*, 2014. 5

[30] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential

learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989. 1

[31] Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual images. In *ECCV*, pages 613–628, 2018. 7

[32] Poojan Oza and Vishal M Patel. Deep cnn-based multi-task learning for open-set recognition. *arXiv preprint arXiv:1903.03161*, 2019. 7

[33] Giulia Pasquale, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Object identification from few examples by improving the invariance of a deep convolutional neural network. In *IROS*, pages 4904–4911. IEEE, 2016. 3

[34] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, pages 759–766. ACM, 2007. 6

[35] Jesse Read, Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. Scalable and efficient multi-label classification for evolving data streams. *Machine Learning*, 88(1-2):243–272, 2012. 1

[36] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 5533–5542. IEEE, 2017. 1, 3

[37] Ryne Roady, Tyler L Hayes, Ronald Kemker, Ayesha Gonzales, and Christopher Kanan. Are out-of-distribution detection methods effective on large-scale datasets? *arXiv preprint arXiv:1910.14034*, 2019. 1

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015. 3, 4, 5

[39] Josef Sivic, Bryan C Russell, Alexei A Efros, Andrew Zisserman, and William T Freeman. Discovering object categories in image collections. In *ICCV*, 2005. 6, 7

[40] Xiaohan Wang, Tengyu Ma, James Ainooson, Seunghwan Cha, Xiaotian Wang, Azhar Molla, and Maithilee Kunda. The toybox dataset of egocentric visual object transformations. *arXiv preprint arXiv:1806.06034*, 2018. 3

[41] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, 2019. 1, 3

[42] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, pages 3987–3995, 2017. 1

[43] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 2017. 7

[44] Jun-Yan Zhu, Jiajun Wu, Yan Xu, Eric Chang, and Zhuowen Tu. Unsupervised object class discovery via saliency-guided multiple class learning. *TPAMI*, 37(4):862–875, 2014. 6, 7