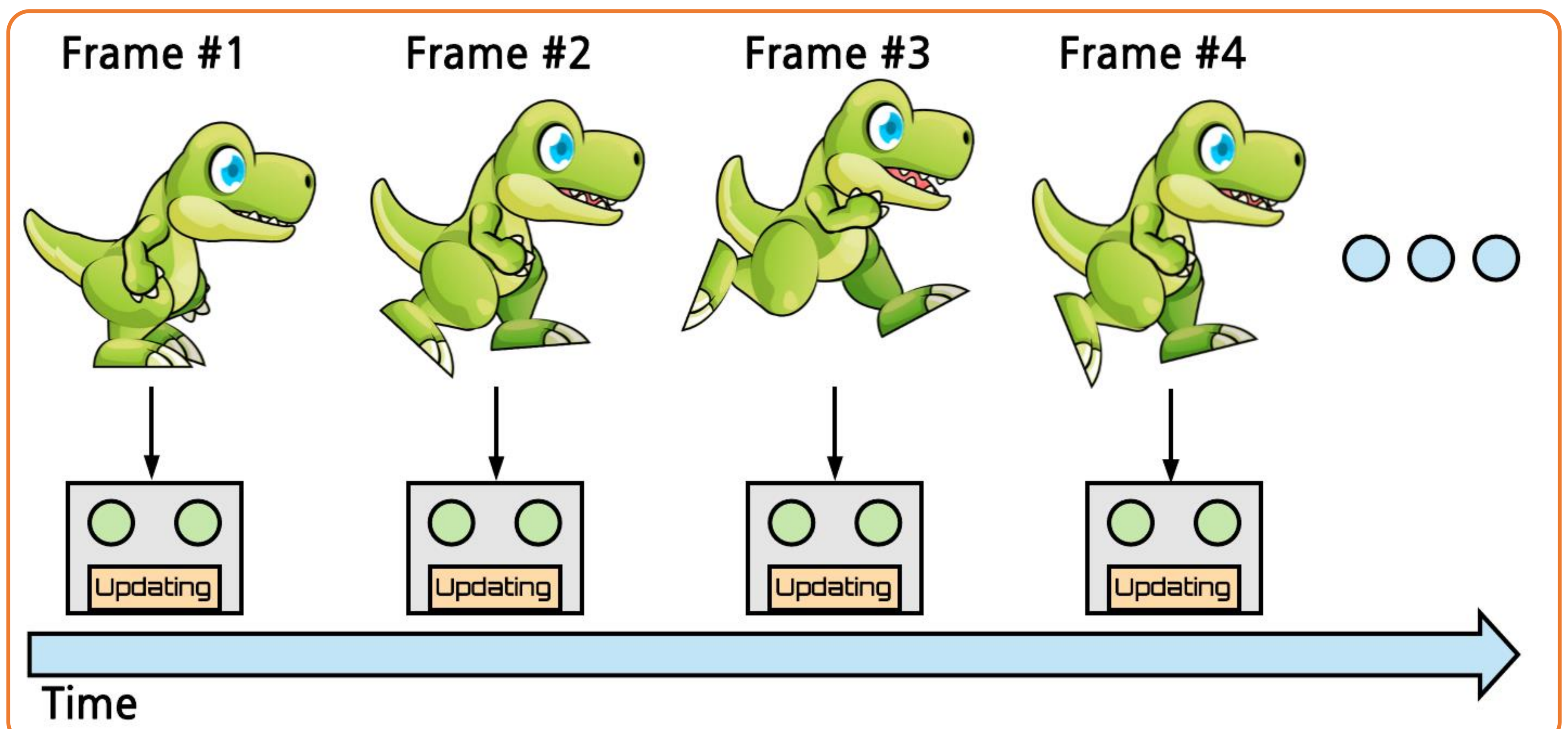
Memory Efficient Experience Replay for Streaming Learning

Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan
Rochester Institute of Technology, Rochester NY
{tlh6792,ndcsma,kanan}@rit.edu

Overview

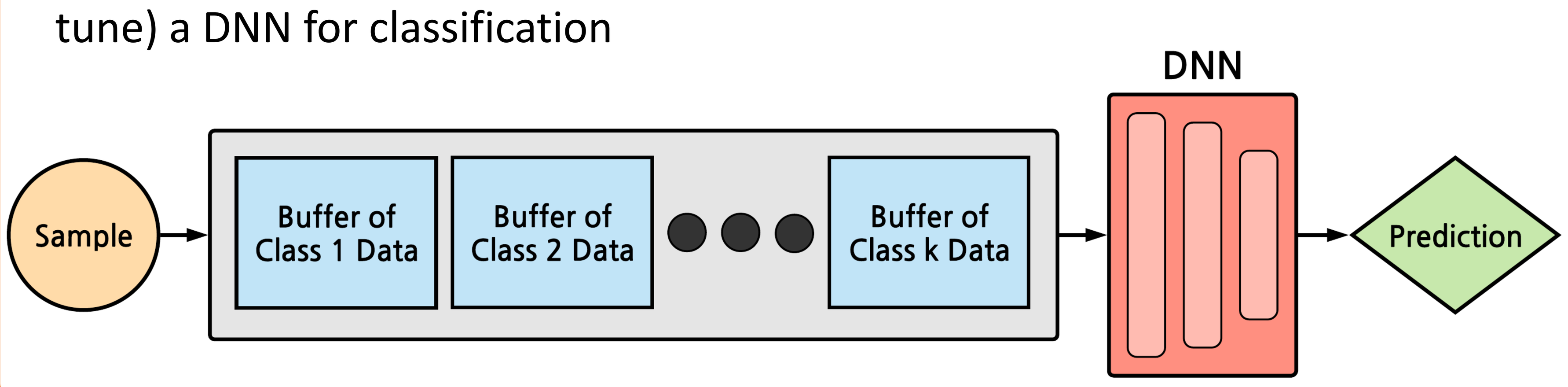
- Agents often operate in changing environments and must quickly learn new things from data streams
- In **streaming learning**, a learner is trained online, in a single pass, from a data stream that may not be independent and identically distributed (iid)
- Deep Neural Networks (DNNs) fail in this paradigm since they require multiple passes through a dataset and non-iid data causes **catastrophic forgetting**
- **Rehearsal** fixes these issues by mixing new examples with all previous data and updating the DNN using this mixture, which is **slow** and **memory intensive**
- We introduce **ExStream**, a memory efficient rehearsal scheme, and compare it to alternatives

Streaming Learning



Memory Efficient Rehearsal

- Full rehearsal mixes **all** older examples with new examples to be learned
- This is not a workable solution for embedded robots deployed for a long time
- We make rehearsal memory efficient, by having K class-specific buffers, each containing at most b prototypes
- The buffers are updated in a streaming fashion and used to update (fine-tune) a DNN for classification



Models

Stream Clustering Buffers:
- *ExStream* – Always store new point and merge two closest clusters
- *Online k-means* – Always merge new point with closest cluster
- *CluStream* – Find closest cluster to new point. If point is within *maximum boundary* of that cluster then merge point in, else create new cluster
- *HPStream* – Find closest cluster to new point using projected distance for high-dimensional data. If point is within *limiting radius* of that cluster then merge point in, else create new cluster

Replacement Buffers:
- *Reservoir Sampling* – Randomly replace existing point with new point
- *Queue* – Replace oldest point with new point

Baselines:
- *No Buffer* – Train DNN sample by sample with single pass through dataset
- *Full Rehearsal* – Store all training data and fine-tune DNN
- *Offline DNN* – Conventional offline DNN trained from scratch on all data

Metrics

- We compute the performance of each method over a set of buffer sizes $\mathcal{B}$
- Performance is normalized to an offline baseline and usually in [0, 1]

Performance for $b \in \mathcal{B}$:

$$\Omega_b = \frac{1}{T}\sum_{t=1}^{T}\frac{\alpha_t}{\alpha_{\text{offline},t}}$$

Performance for $\mathcal{B}$:

$$\mu_{\text{total}} = \frac{1}{|\mathcal{B}|}\sum_{b\in\mathcal{B}}\Omega_b$$

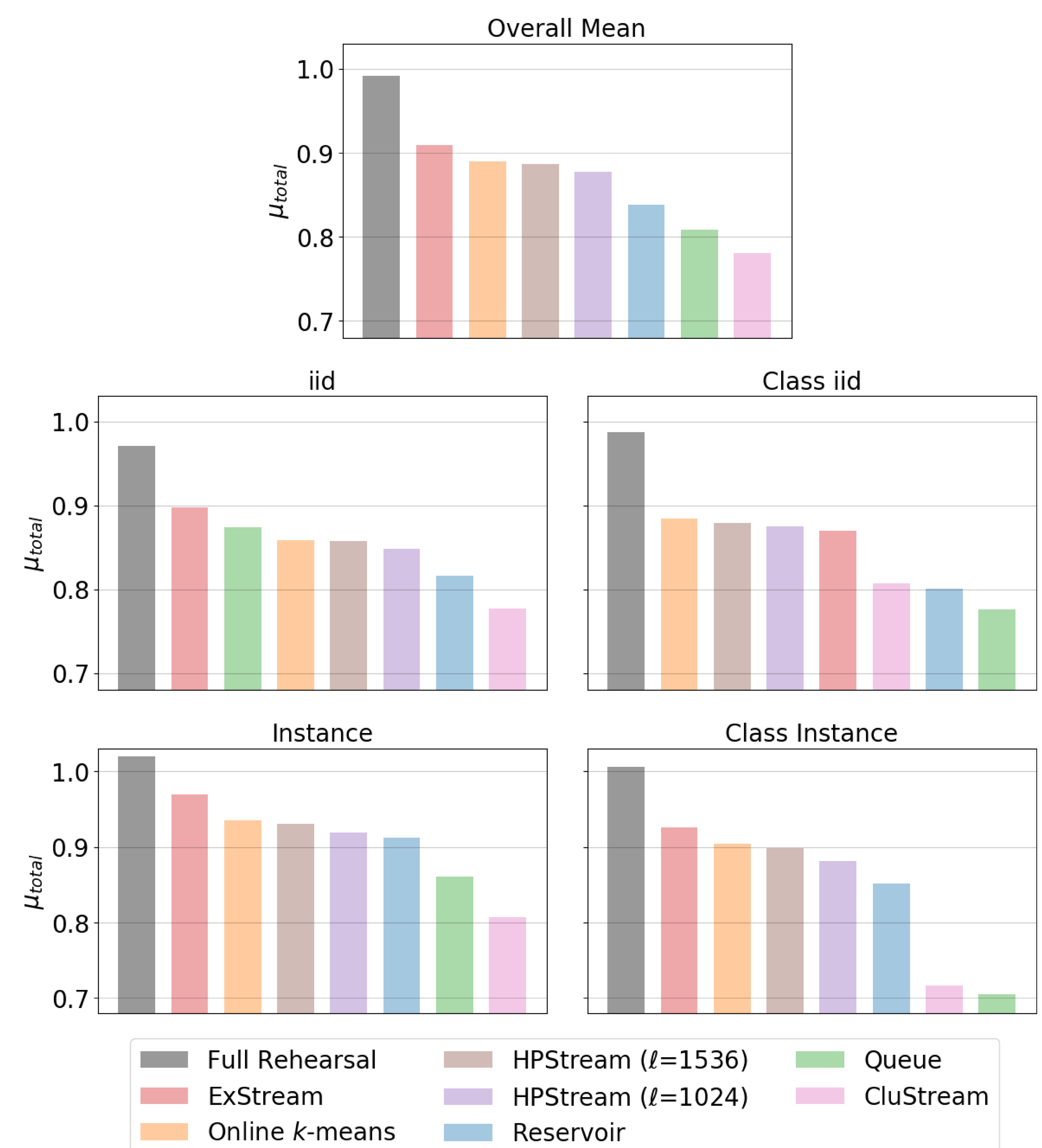
Streaming Learning Paradigms

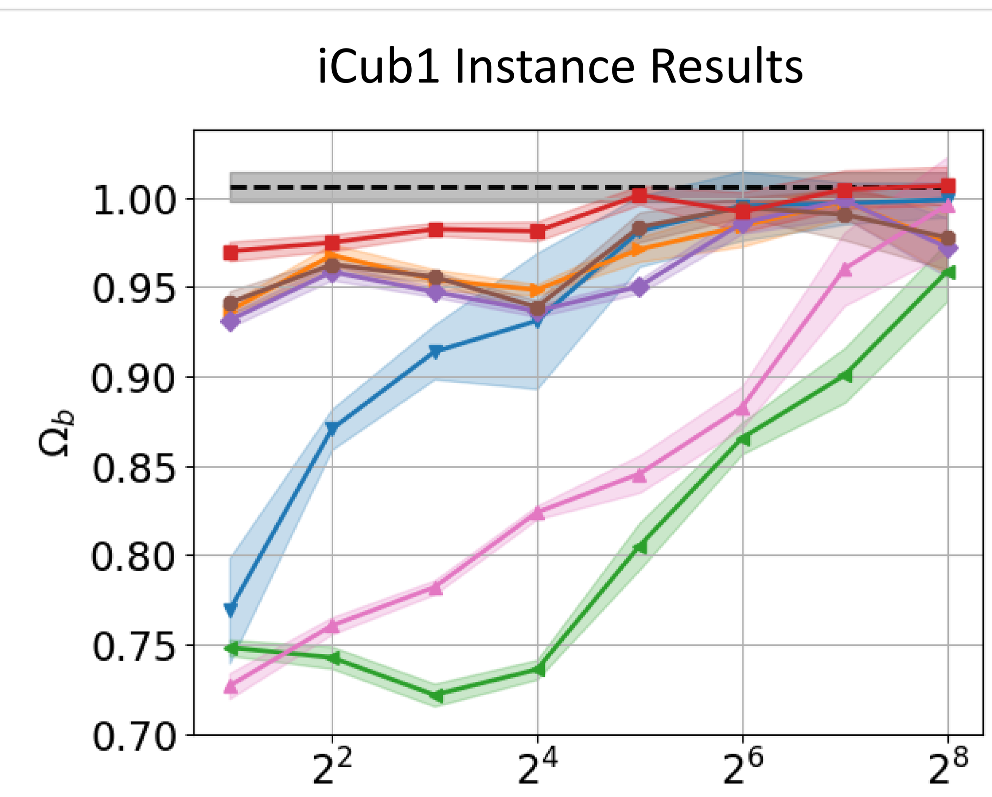
- *iid Ordered* – data stream is randomly shuffled
- *Class iid Ordered* – data stream is organized by class
- *Instance Ordered* – data stream is temporally ordered by object instances
- *Class Instance Ordered* – data stream is temporally ordered by object instances by class

Datasets

	iCub1	CORe50	CUB-200
Type	Streaming	Streaming	Standard Obj. Rec.
Classes	10	10	200
Feature Shape	2,048	2,048	2,048
Train Samples	6,002	5,943	5,994
Test Samples	2,001	2,232	5,794
Train Samples/Class	600-602	591-600	29-30
Test Samples/Class	200-201	221-225	11-30
Buffer Sizes	$\{2^1, 2^2, \cdots, 2^8\}$	$\{2^1, 2^2, \cdots, 2^8\}$	$\{2^1, 2^2, \cdots, 2^4\}$

Experimental Results





Summary

- We demonstrated the **effectiveness of rehearsal** for mitigating catastrophic forgetting during streaming learning with DNNs
- We showed that rehearsal can be done in a **memory efficient** way by introducing the **ExStream algorithm** and demonstrating its efficacy on multiple orderings of high-resolution datasets

Acknowledgements:

We thank DARPA L2M and the US NRL for financially supporting this research.