Streaming Synergistic Forests

Nick Hahn[1*], Haoyin Xu[1], Jayanta Dey[1], Joshua T. Vogelstein[1]

[1] Whiting School of Engineering, Johns Hopkins University, [*] correspondence: nhahn7@jhu.edu

BME Design Day 2022

Summary

- Biological learning leverages streaming data to improve performance on prior tasks as well as future tasks
- Typical transfer learning algorithms experience catastrophic forgetting when learning tasks sequentially
- Current lifelong learning algorithms utilize standard batch mode decision forests
- We introduce preliminary results of a lifelong learning algorithm leveraging streaming data and incremental learning
- Streaming synergistic forests (SynF) are able to utilize streaming data from past tasks to improve performance on future tasks and avoid catastrophic forgetting

Background & Algorithm

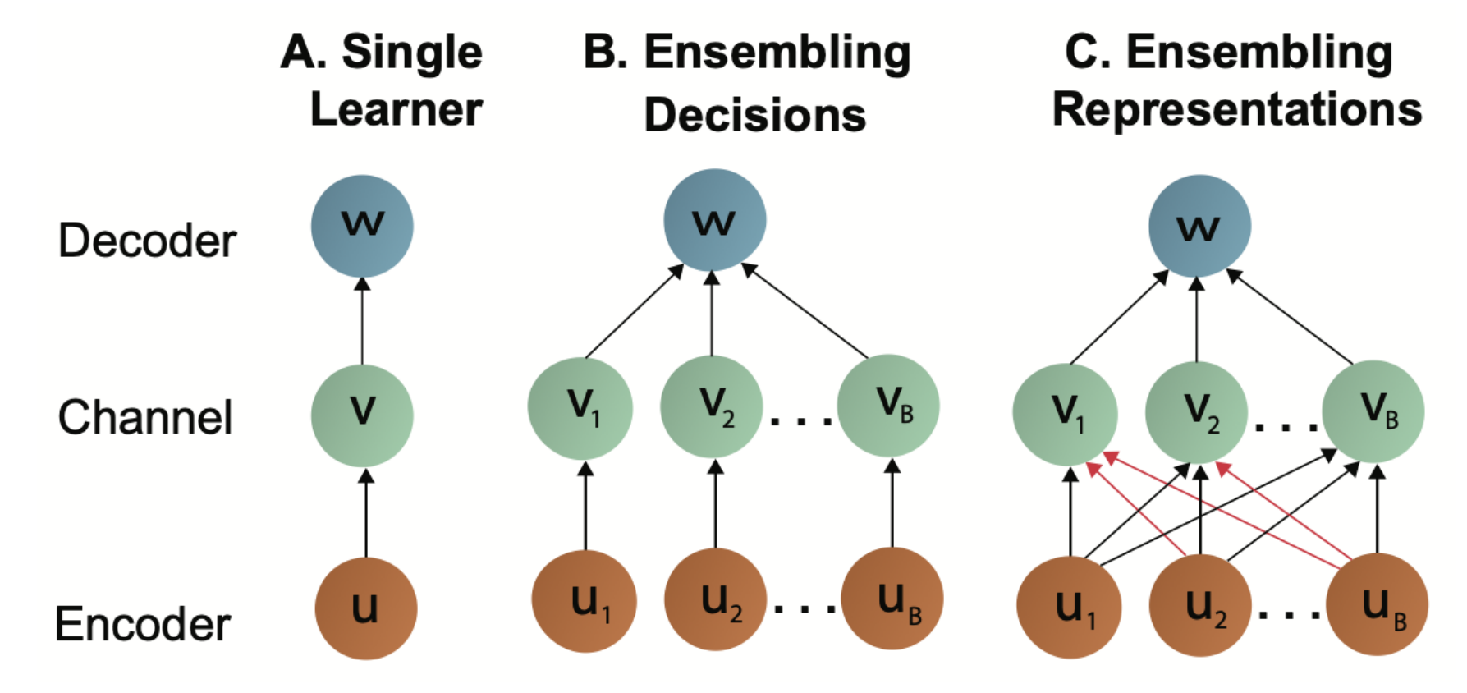

Fig 1 Representation Ensembling for Lifelong Learning. Encoders are representations learned by decision forests, channels output average normalized class votes, and the decoder outputs the argmax prediction. [1]

- Previous work has shown positive forward and backward transfer by ensembling representations learned by independent decision forests [1].
- Current standard implementations of decision forests operate in batch mode. In many real world applications, we are not provided with all data at once and therefore need to incrementally update as data arrives.
- We use a custom fork of `scikit-learn` with added `partial_fit` functionality allowing for incremental updating [2]

Pseudocode

```
Algorithm 1 updating SynF                                    $OOB$ = out-of-bag
Input:
    (1) $\mathcal{D}_u^t = (\mathbf{x}_u^t, \mathbf{y}_u^t)$                     ▷ Data to update task $t$
    (2) $t$                                                  ▷ task id
Output:
    (1) SynF                                                 ▷ updated synergistic forest
 1: function UPDATE_TASK($\mathcal{D}_u^t, t$)
 2:     $\mathcal{D}_n^t \leftarrow \mathcal{D}_n^t + \mathcal{D}_u^t$            ▷ Concatenate existing data and new data
 3:     split into encoder/channel data and decoder data
 4:     for each tree in the forest do
 5:         resample from encoder data
 6:         partial fit tree
 7:         update channel data index
 8:     end for
 9:     for $t' = 0, \ldots, t' = t-1$ do                    ▷ For backwards task ids
10:         fit channel with $OOB$ data
11:         set decoder                                      ▷ updates encoders and channels
12:     end for
13:     return SynF
14: end function
```

Current SOTA Streaming Forests/Trees

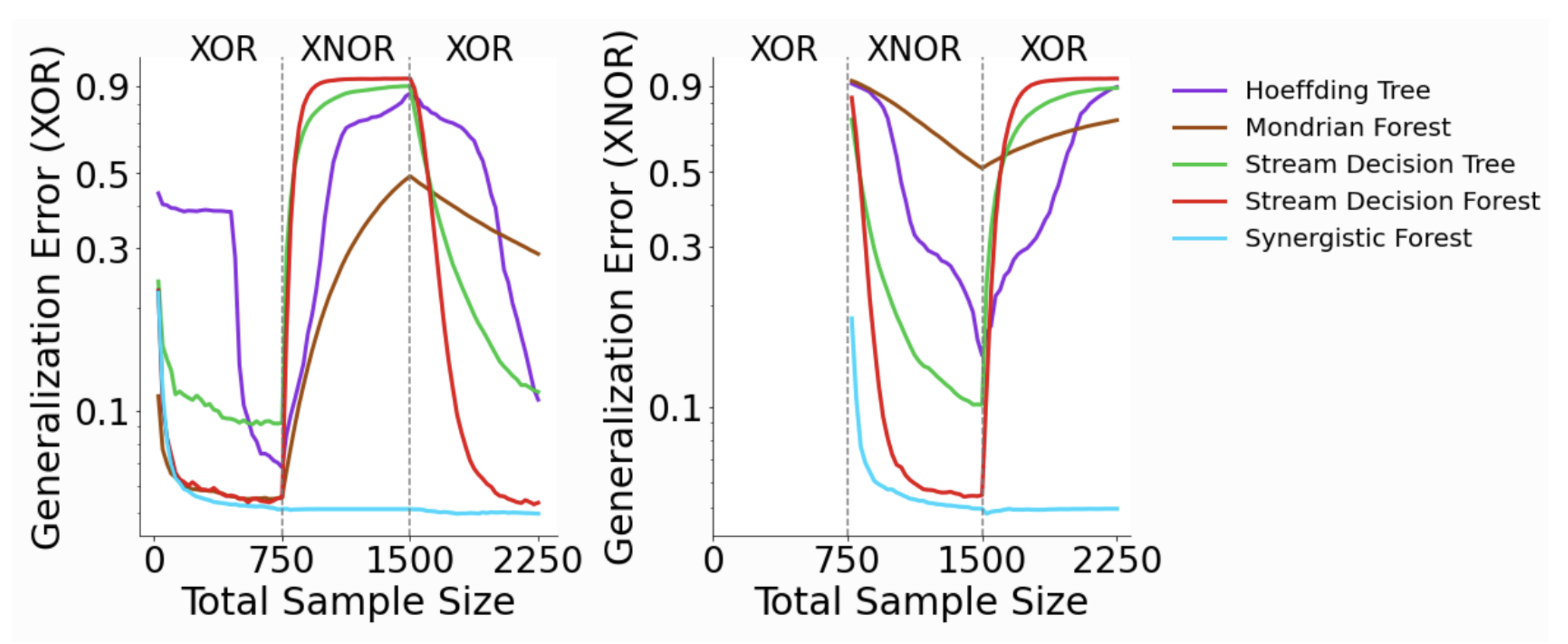

Fig 2: When XNOR data are introduced, SynF is the only classifier which does not deteriorate in performance. SynF is able to leverage past representations whereas all other classifiers must re-learn XNOR from scratch.

Gaussian XOR Experiments

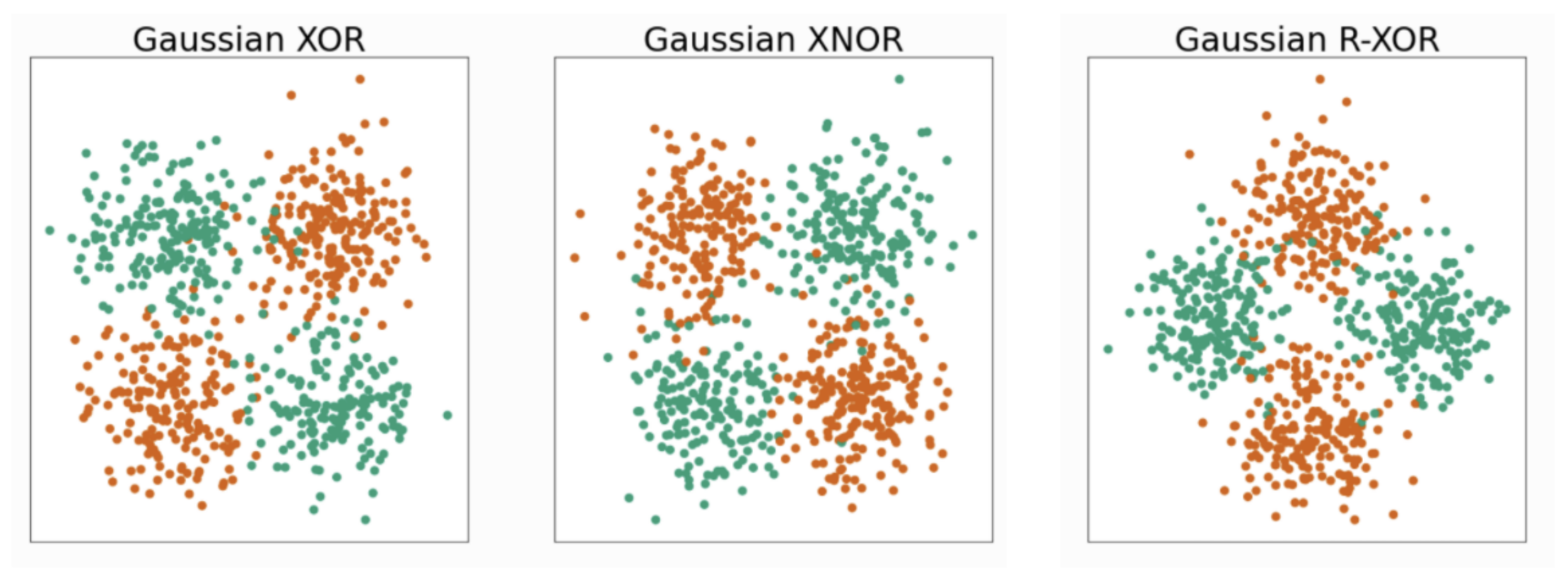

Fig 3: Gaussian XOR, XNOR,and R-XOR simulated data. Gaussian XNOR has the same distribution as Gaussian XOR, but with the class labels rotated 90 degrees. Therefore, XNOR has the same optimal discriminant boundary as XOR. Gaussian R-XOR has the same distribution as Gaussian XOR, but with the class labels rotated 45 degrees.

Gaussian XNOR

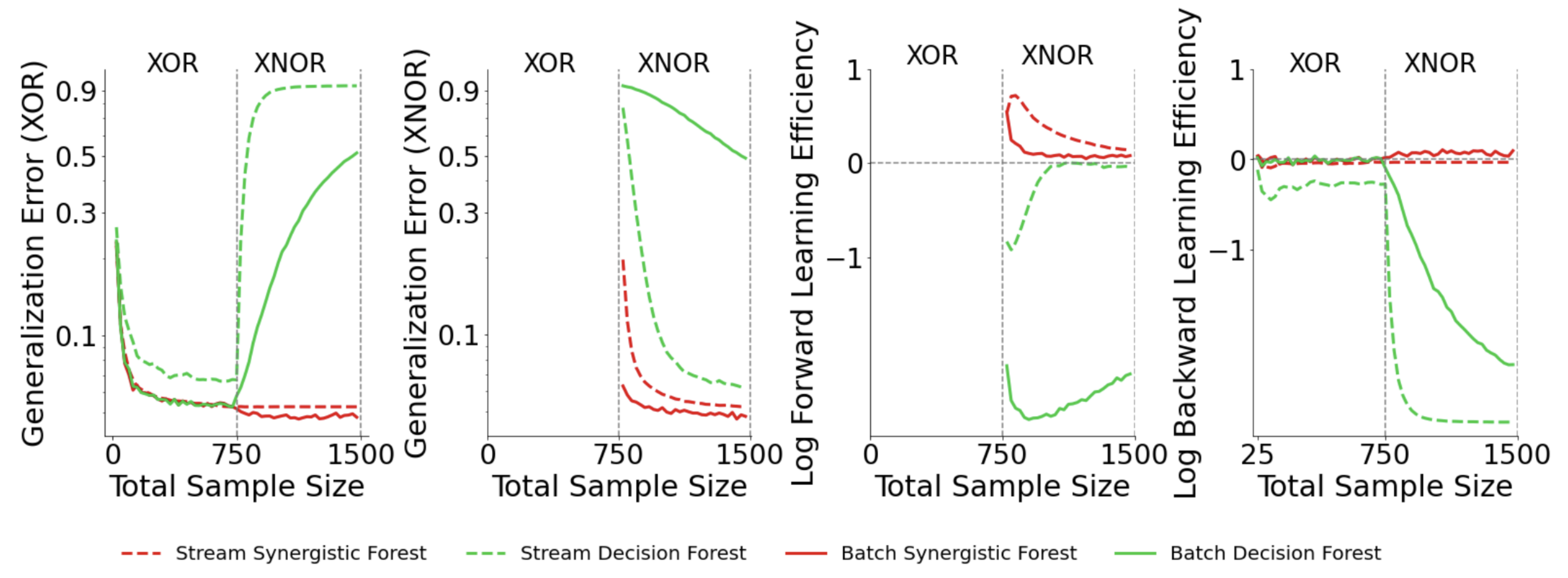

Fig 4: XOR generalization error (left), XNOR generalization error (middle left), log forward learning efficiency (middle right), and log backward learning efficiency (right). Data are introduced 25 samples at a time.

Gaussian RXOR

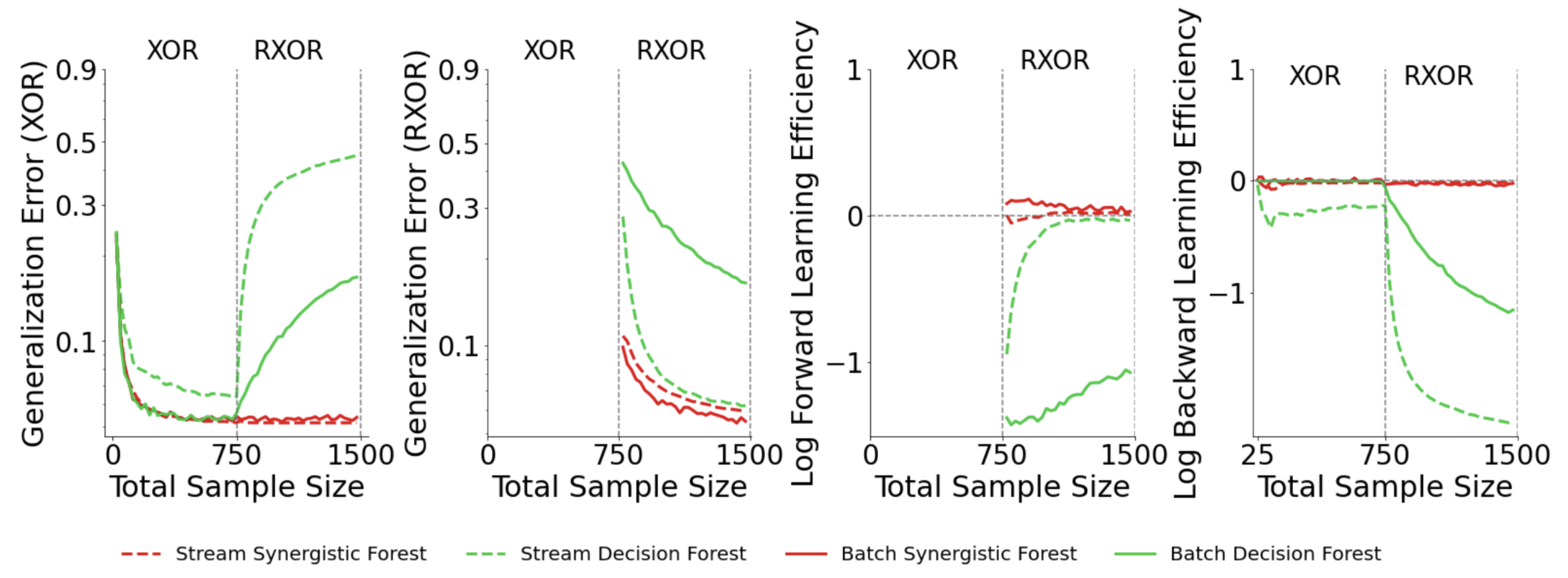

Fig 5: XOR generalization error (left), R-XOR generalization error (middle left), log forward learning efficiency (middle right), and log backward learning efficiency (right). Data are introduced 25 samples at a time.

Evaluation Metrics

The learning ability of streaming synergistic forests is measured with the metric of learning efficiency and its variants [1]. The **learning efficiency** of an algorithm f for a given task t with sample size n is defined as

$$\mathrm{LE}_n^t(f) := \frac{\mathbb{E}[R^t(f(\mathbf{S}_n^t))]}{\mathbb{E}[R^t(f(\mathbf{S}_n))]}$$

Iff $\mathrm{LE}_n^t(f) > 1$, f has learned task t with data $\mathbf{S}_n$

forward learning efficiency is defined as

$$\mathrm{FLE}_n^t(f) := \frac{\mathbb{E}[R^t(f(\mathbf{S}_n^t))]}{\mathbb{E}[R^t(f(\mathbf{S}_n^{\le t}))]}$$

If $\mathrm{FLE}_n^t(f) > 1$, f has leveraged data from past tasks to improve performance on task t (forward transfer)

backward learning efficiency is defined as

$$\mathrm{BLE}_n^t(f) := \frac{\mathbb{E}[R^t(f(\mathbf{S}_n^{\le t}))]}{\mathbb{E}[R^t(f(\mathbf{S}_n))]}$$

If $\mathrm{BLE}_n^t(f) > 1$, f has leveraged data from future tasks to improve performance on previous tasks (backward transfer)

An algorithm f has **synergistically learned** if $\log \mathrm{LE}_n^t(f) > 0$ for all $t \in \mathcal{T}$

Limitations and Future Directions

- Current Streaming Synergistic Forest implementation avoids catastrophic forgetting, but is unable to positively backwards transfer.
- Integrate streaming capabilities for forests and networks into `neurodata/ProgLearn`

Acknowledgements

Thank you to Haoyin Xu, Jayanta Dey, Dr. Joshua Vogelstein and the 2021-2022 Neuro Data Design class for their support and guidance throughout this project.

References

[1] Joshua T. Vogelstein et al. Representation Ensembling for Synergistic Lifelong Learning with Quasilinear Complexity. 2020. doi: 10.48550/ARXIV.2004.12908. url: https://arxiv.org/abs/2004.12908.

[2] Haoyin Xu et al. Simplest Streaming Trees. 2021. doi: 10.48550/ARXIV.2110.08483. url: https://arxiv.org/abs/2110.08483.