Qualifying Exam

Zhepei Wang
Computational Audio Lab, Computer Science, UIUC

11 September 2019

Outline

- Background
- My research
  - Overview
  - "Multi-View Networks for Multi-Channel Audio Classification"
- Paper presentation
  - "Marginal Replay vs Conditional Replay for Continual Learning"

About Zhepei

- Computational Audio Lab
  - Third semester
  - Advised by Prof. Paris Smaragdis
  - Research on applying ML/DL to audio related tasks
- Graduated from Harvey Mudd College
  - B.S., Computer Science
  - Music Information Retrieval (MIR) Lab: song identification

Research Overview

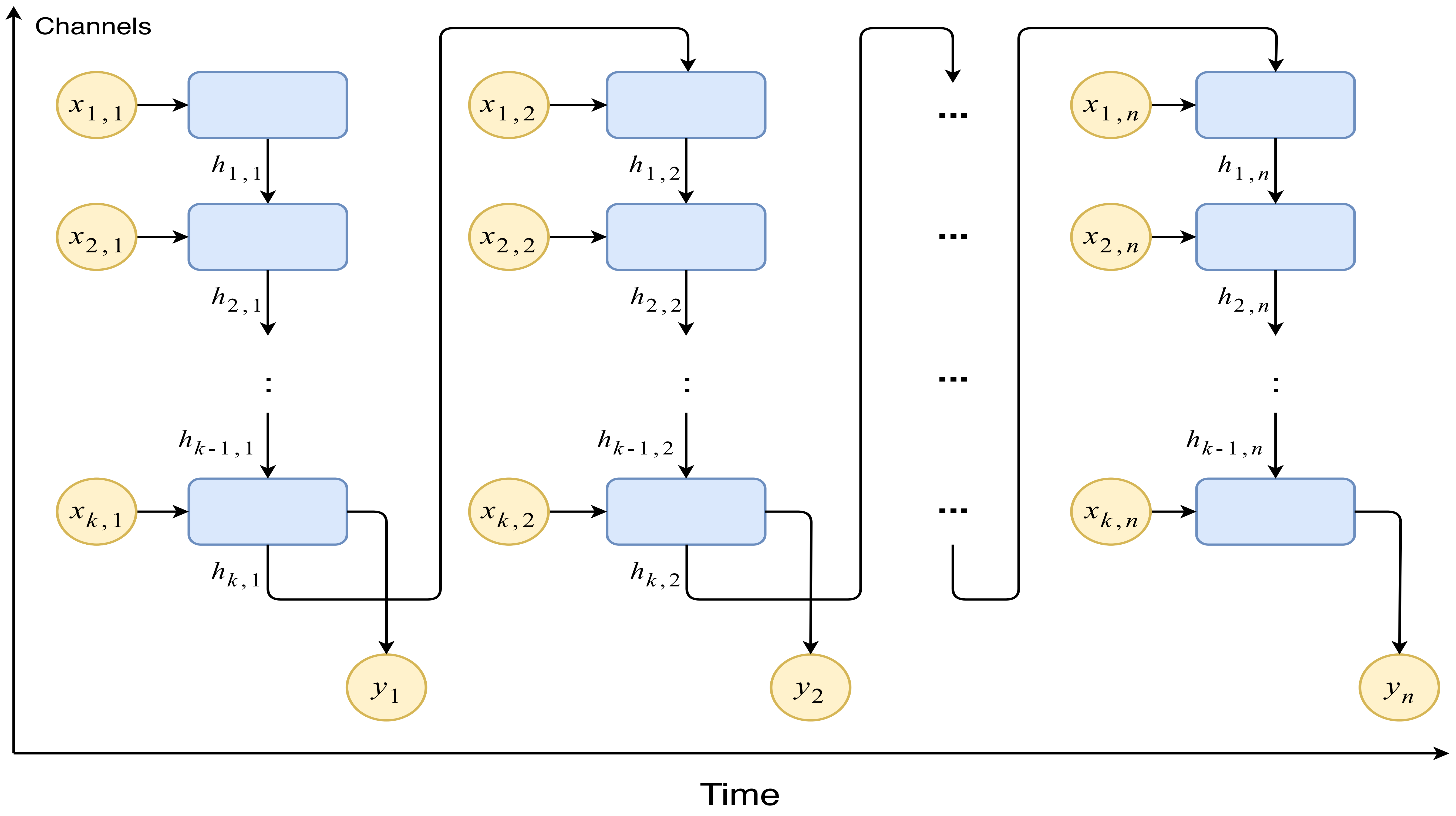
- Audio Classification
  - **Acoustic scene classification (ASC)**
    - Identify the environment in which the signal is produced
    - Given a recording $\mathbf{x} \in \mathbb{R}^T$, predict $y = f_\theta(\mathbf{x}) \in \{0, 1, \ldots, C-1\}$
    - One label per sequence
    - One paper accepted in WASPAA 2019

Research Overview

- Audio classification
  - Acoustic scene classification (ASC)
    - Given a signal $\mathbf{x} \in \mathbb{R}^T$, predict $y = f_\theta(\mathbf{x}) \in \{0,1,\ldots,C-1\}$
    - One label per sequence
  - **Voice activity detection (VAD)**
    - Identify the occurrences of the activity in interest
    - Given a signal $\mathbf{x} \in \mathbb{R}^T$, predict $\mathbf{y} = f_\theta(\mathbf{x}) \in \{0,1,\ldots,C-1\}^T$
    - One label per frame

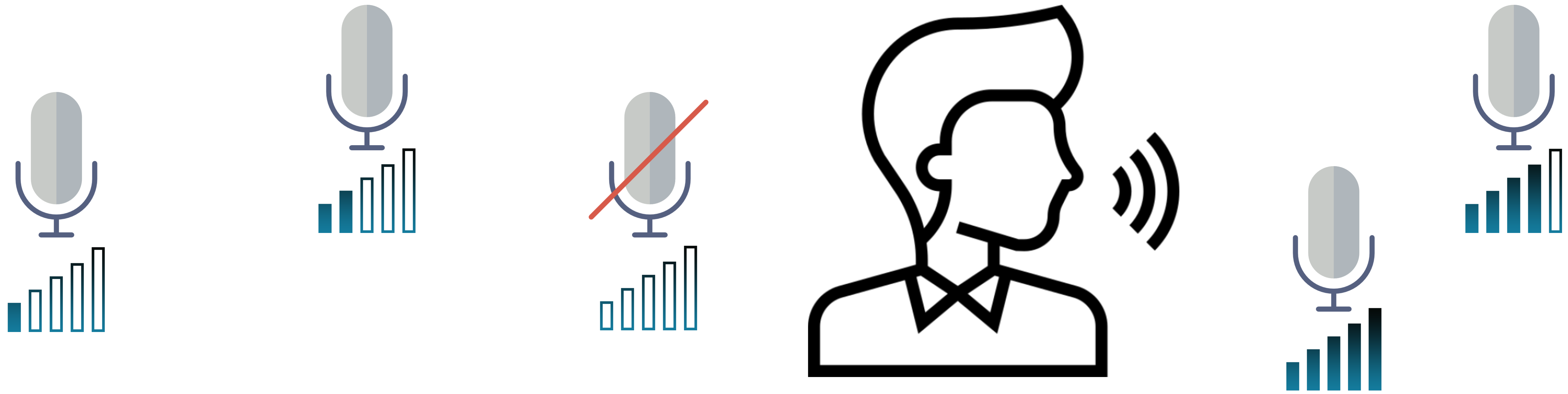
Recent Research: MVN

- “Multi-View Networks for Multi-Channel Audio Classification”, coauthored with Jonah Casebeer
- # channels = # devices (recordings)
- Paper accepted to ICASSP 2019



MVN: Motivation

- Imagine that we're in a conference setting…
- Goal: detect speech recorded from multiple devices
- Varying number of acoustic devices
- Different recording quality



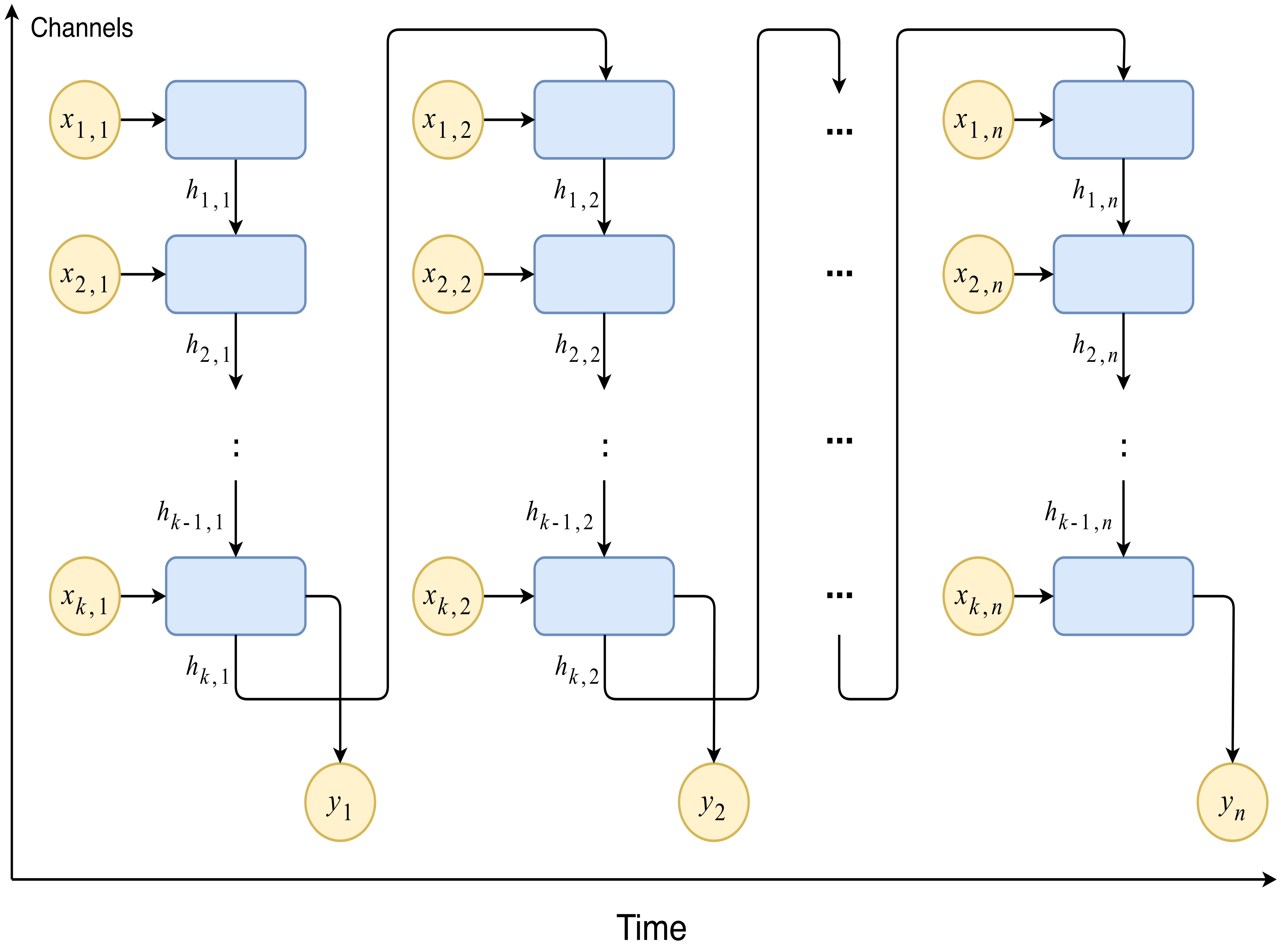
Previous Work

- Beamforming
  - Linear combination of signals from each microphone in the array
  - Able to operate on an arbitrary number of input channels
- Deep neural networks
  - Outperforms beamforming for a fixed number of channels
  - Not adaptive to varying number of channels
    - Trained on K channels, not able to perform well on K' channels, $K \neq K'$
- What we want…
  - A learning based method
  - Handles varying number of input channels

Multi-View Network (MVN): Proposal

- A variant of RNN
- Accepts input of arbitrary number of channels
- Unrolls across both channels and time steps

MVN: Architecture and Recurrence

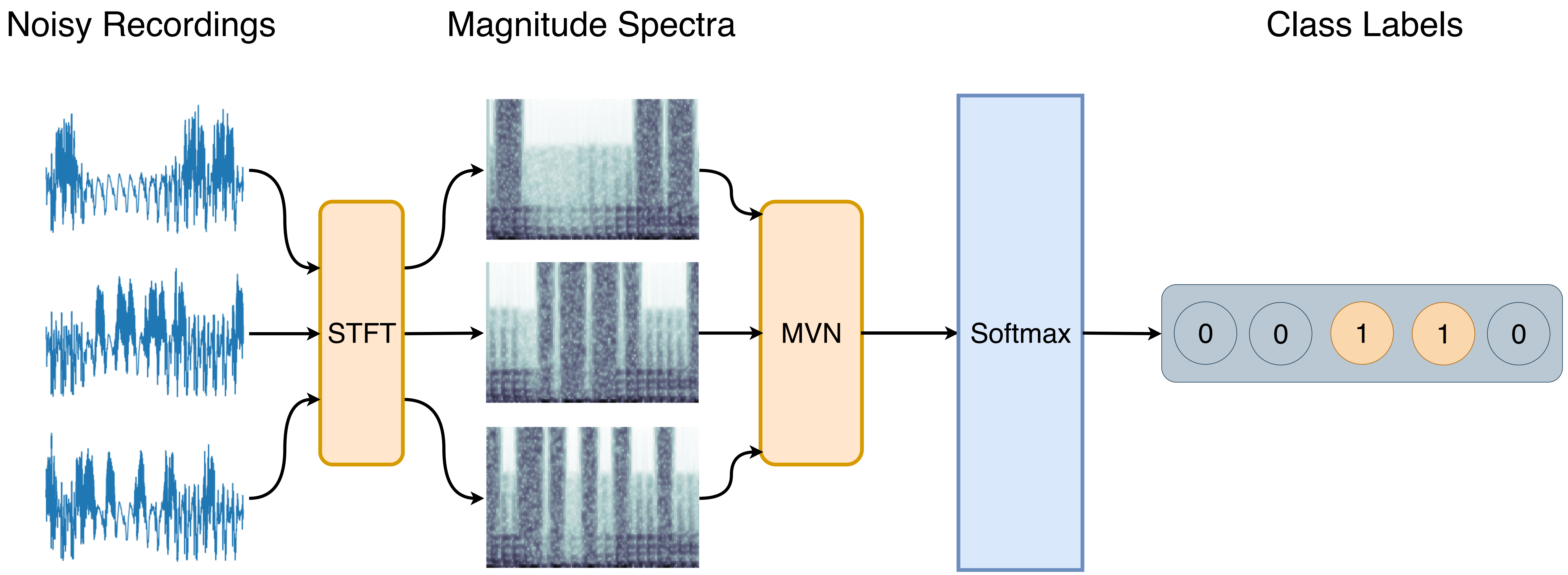


$$\mathbf{h}_{k,t} = \begin{cases} \sigma(\mathbf{W}_x \cdot \mathbf{x}_{k,t} + \mathbf{U}_h \cdot \mathbf{h}_{k,t-1}), k = 1 \\ \sigma(\mathbf{W}_x \cdot \mathbf{x}_{k,t} + \mathbf{U}_h \cdot \mathbf{h}_{k-1,t}), 1 < k \leq K \end{cases}$$

$$\mathbf{y}_t = \sigma(\mathbf{W}_h \cdot \mathbf{h}_{K,t})$$

MVN: Pipeline

- Take short-time Fourier transform (STFT) for each recording
- Unroll across each STFT frame and predict by frame

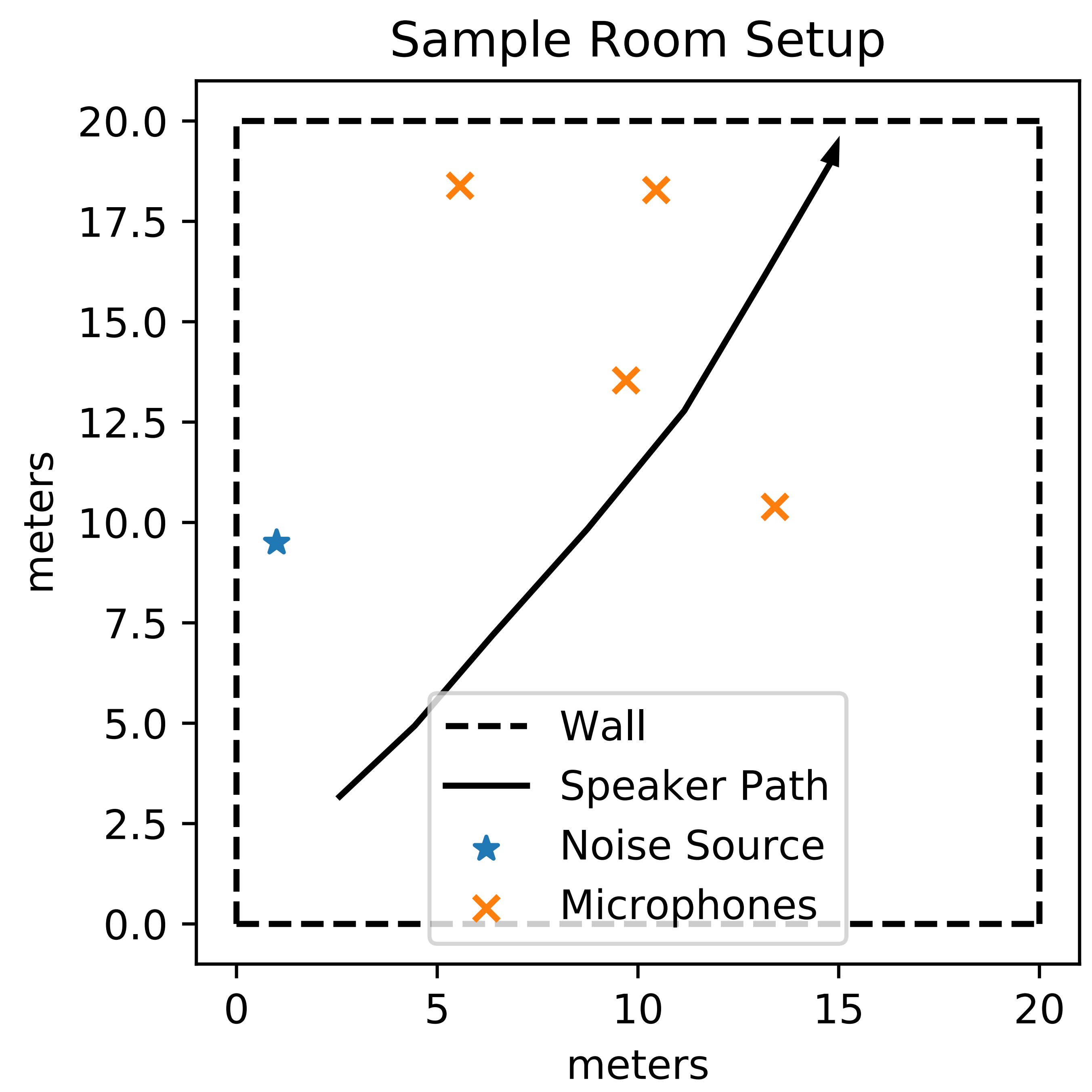


Experiments: Data

- TIMIT (speech) + 13 Urban Noise Classes
- Training set
  - 4-channel 2-second intermittent speech and noise
  - ~50% speech frames
  - SNR linearly spaced between -5 and 5 dB
- Test set
  - 2 - 30 channels

Experiments: Data

- Room simulation
  - 20m x 20m reverberant room
  - Moving point speech source
  - Diffuse noise source
  - Stationary microphones



- Different room geometries between training and test

Experiments: Baseline

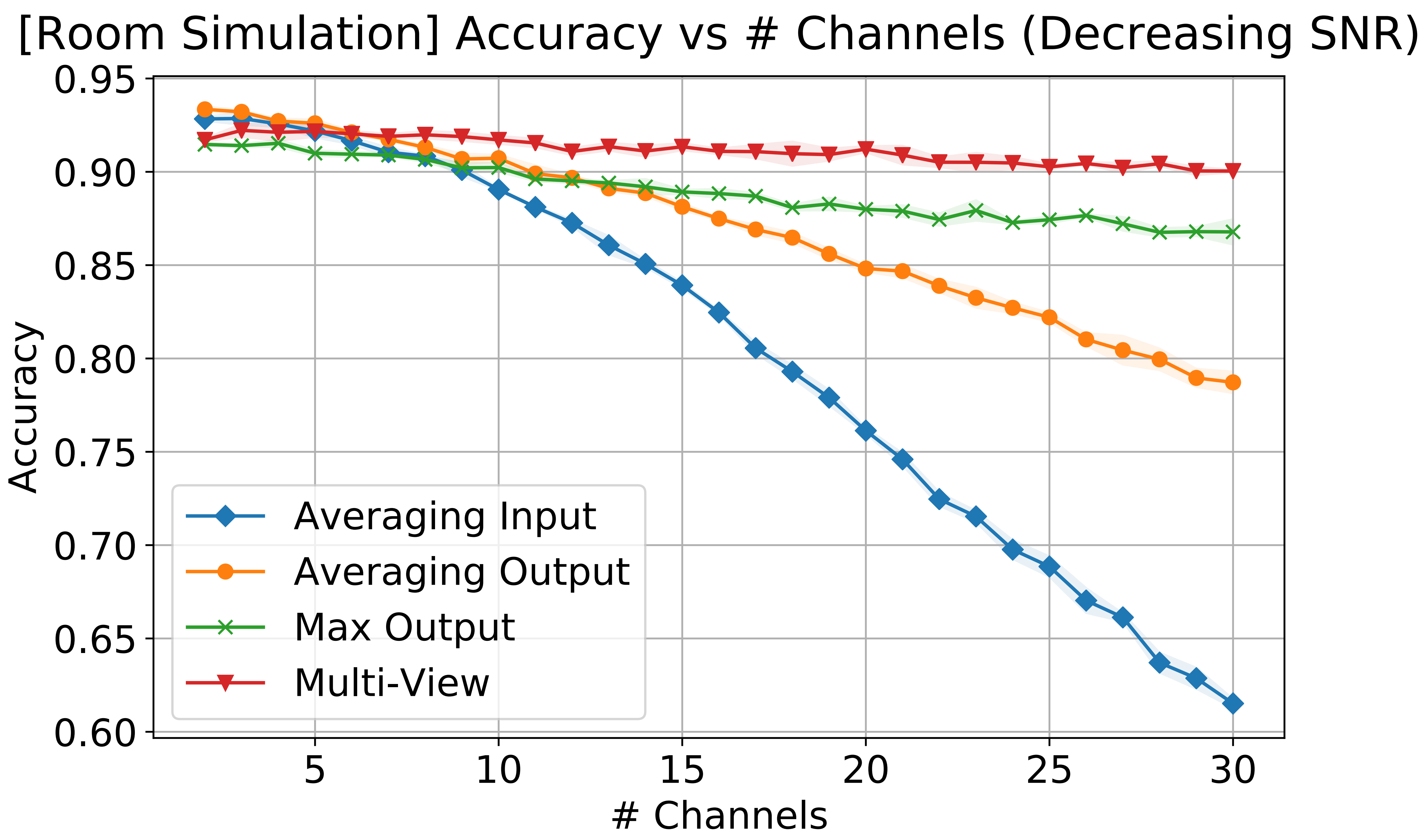
- Considering the following alternatives to MVN:
  - Averaging input
  - Averaging output
  - Max output: taking output channel with highest probability
- Share the same RNN architecture with MVN
- Simple (weighted) averaging scheme

Experiments: Configuration

- STFT: 1024 pt window, 512 pt hop
- Network: single layer, unidirectional GRU with 512 units
- Objective function: cross entropy
- Optimized with Adam

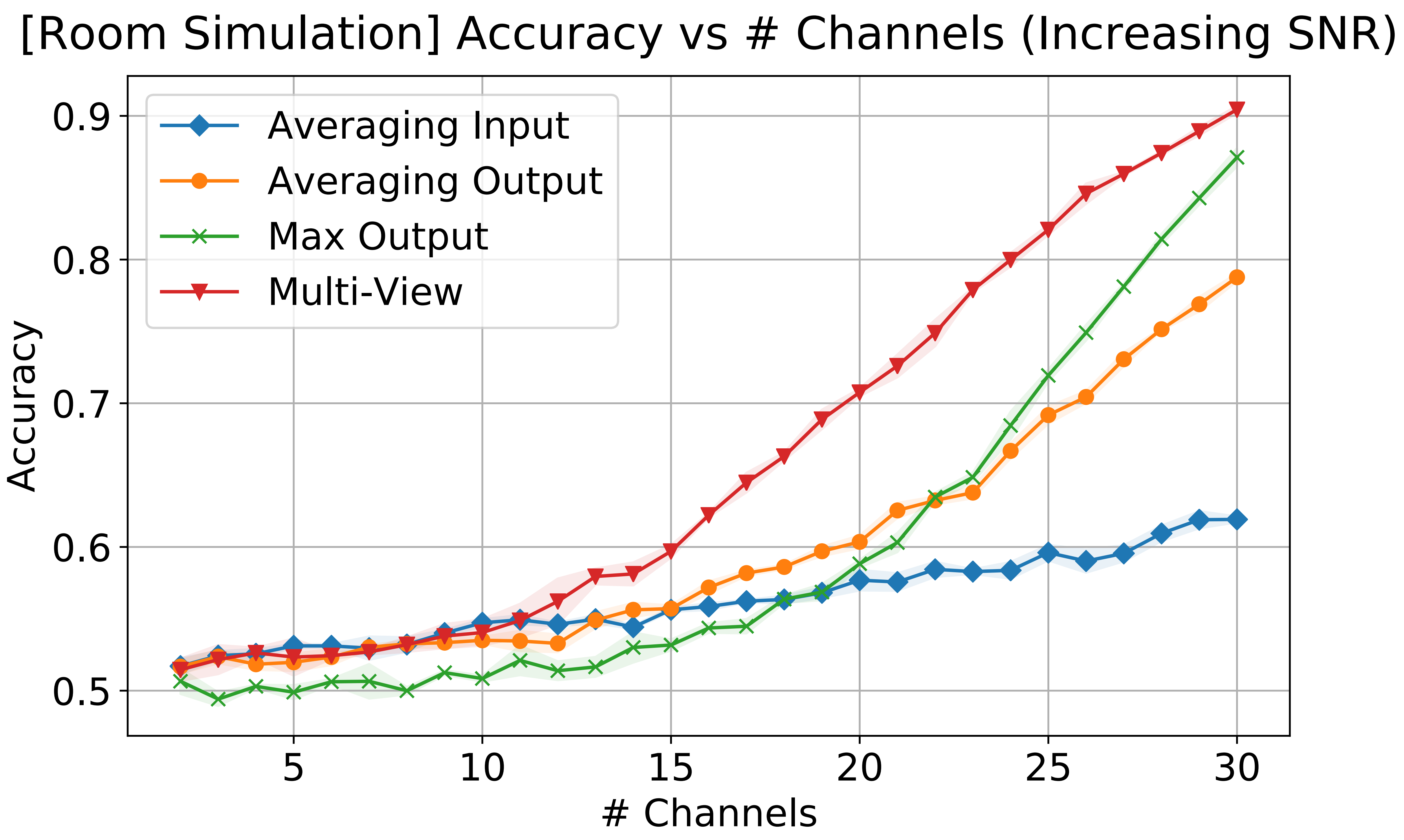
Experimental Results: Decreasing SNR

- Each new channel has a **lower** SNR than previous channels
- SNR decreases from 0 to -29 dB
- MVN less affected by channels with poor signal quality



Experimental Results: Increasing SNR

- Each new channel has a **higher** SNR than previous channels
- SNR increases from -29 to 0dB
- MVN more effective at collecting information from limited clean channels



Takeaways

- Robust performance
  - Arbitrary number of input channels
  - Unseen room geometries
  - SNR varies largely across channels
- Processing is invariant to order of input channels
- Potential extensions
  - More classes, deeper networks, different architectures
  - Source separation: arbitrary number of output channels?

Marginal Replay vs Conditional Replay for Continual Learning

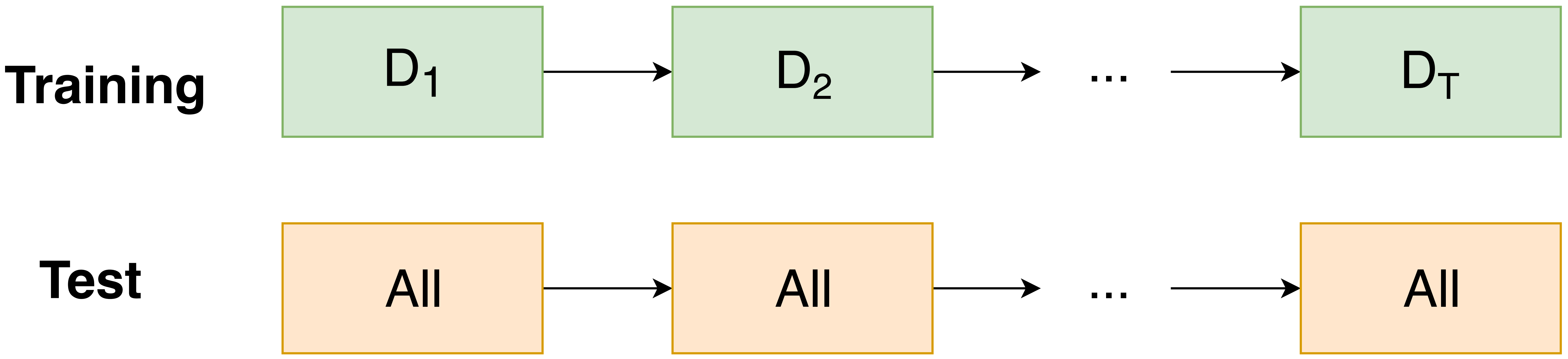
T. Lesort et. al.

Paper Presentation

- What?
  - Continual learning, generative replay, marginal replay, conditional replay
- How (and why)?
  - Generative replay vs regularization
  - Conditional replay vs marginal replay
- Contributions and comments

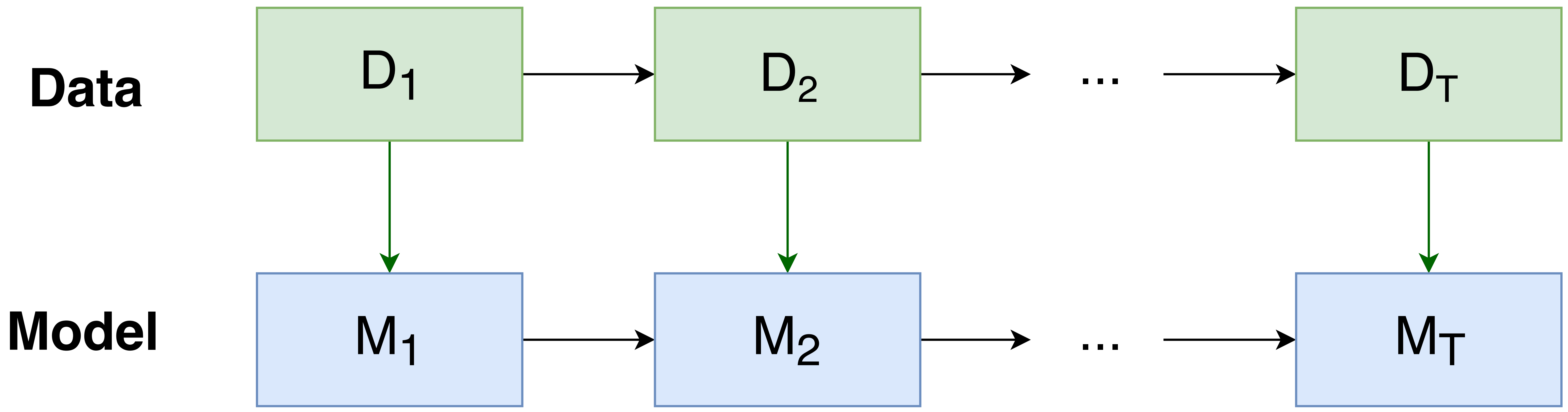
Task

- Continual Learning Task (CLT) (a.k.a incremental/lifelong learning)
  - Given a sequence of tasks and a dataset for each task
  - Want to learn one task at a time
  - Past or future data not accessible
  - Learn from new task while retaining past knowledge
  - Assuming all tasks are classification



Problem

- Catastrophic Forgetting (CF)
  - Brains/models tend to forget previous knowledge
  - DNN algorithms are "greedy"
    - Weights update minimizes the loss for **only the current task**
    - Performance on previous tasks may degrade



CL Approaches

- Regularization
  - Penalize update on weights important to previous tasks
  - Pros: constant time/memory
  - Cons: performance
- Generative replay
  - Use a generator to recover samples from previous tasks
  - Pros: good and robust performance
  - Cons: time and memory complexity

Generative Replay

- Use a generator to reproduce past data
- Generator can be trained in parallel to classifier
- Model-agnostic
- Marginal replay vs conditional replay
  - Whether or not using conditioning vector in generator

Marginal Replay

- Algorithms (for two tasks)

```
# task 1
train C_1, G_1 from D_1 = (X_1, Y_1)
# task 2
generate X_1^rep from G_1
generate Y_1^rep from C_1
train C_2, G_2 from D_2 ∪ D_1^rep
store C_2, G_2 and discard C_1, G_1
```

task 1

train C_1, G_1 from $\mathscr{D}_1 = (\mathscr{X}_1, \mathscr{Y}_1)$

task 2

generate $\mathscr{X}_1^{rep}$ **from** G_1

generate $\mathscr{Y}_1^{rep}$ **from** C_1

train C_2, G_2 from $\mathscr{D}_2 \cup \mathscr{D}_1^{rep}$

store C_2, G_2 and discard C_1, G_1

Marginal Replay

- Algorithms (full)

`train` C_1, G_1 `from` $\mathscr{D}_1 = (\mathscr{X}_1, \mathscr{Y}_1)$

`for t = 2…T`

  `generate` $\mathscr{X}^{rep}_{1:t-1}$ **`from`** G_{t-1}

  `generate` $\mathscr{Y}^{rep}_{1:t-1}$ **`from`** C_{t-1}

  `train` C_t, G_t `from` $\mathscr{D}_t \cup \mathscr{D}^{rep}_{1:t-1}$

  `store` C_t, G_t `and discard` C_{t-1}, G_{t-1}

Conditional Replay

- Algorithms (for two tasks)

`# task 1`

`train` C_1, G_1 `from` $\mathscr{D}_1 = (\mathscr{X}_1, \mathscr{Y}_1)$

`# task 2`

`generate` $\mathscr{X}_1^{rep}$ **`from`** G_1 **`conditioned on`** $\mathscr{Y}_1^{rep}$

`train` C_2, G_2 `from` $\mathscr{D}_2 \cup \mathscr{D}_1^{rep}$

`store` G_2 `and discard` G_1

Conditional Replay

- Algorithms (full)

```
train $C_1, G_1$ from $\mathscr{D}_1 = (\mathscr{X}_1, \mathscr{Y}_1)$
for t = 2…T
    generate $\mathscr{X}^{rep}_{1:t-1}$ from $G_{t-1}$ conditioned on $\mathscr{Y}^{rep}_{1:t-1}$
    train $C_t, G_t$ from $\mathscr{D}_t \cup \mathscr{D}^{rep}_{1:t-1}$
    store $G_t$ and discard $G_{t-1}$
```

University of Illinois at Urbana-Champaign

Experiments: Questions

- Generative replay vs regularization
  - Test accuracy on image classification
  - Time and memory comparison is trivial
- Marginal replay vs conditional replay
  - Test accuracy
  - Time and memory cost on replay generation

Experiments: Setup

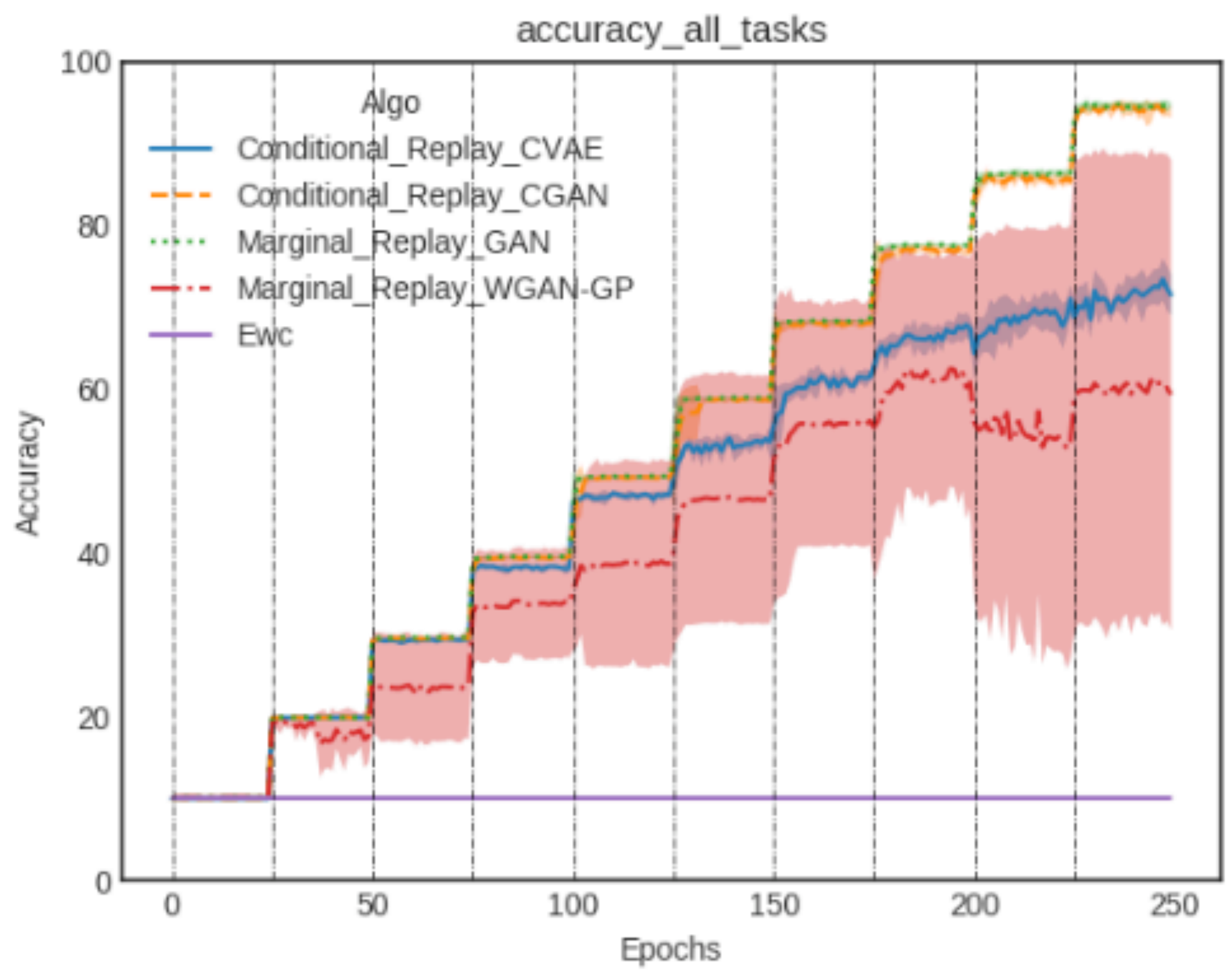
- Dataset: MNIST, FashionMNIST
  - Training/validation: data from current task
  - Test: data from **all tasks**
- Tasks
  - Three different schemes (each contains a sequence of 5 or 10 tasks):
    - Rotations: random rotation angle $\beta \in [0, \pi/2]$
    - Permutations: a random pixel permutation scheme
    - **Disjoint classes**: each task contains samples of only one class
      - No between-class discrimination from the training data

Experiments: Setup

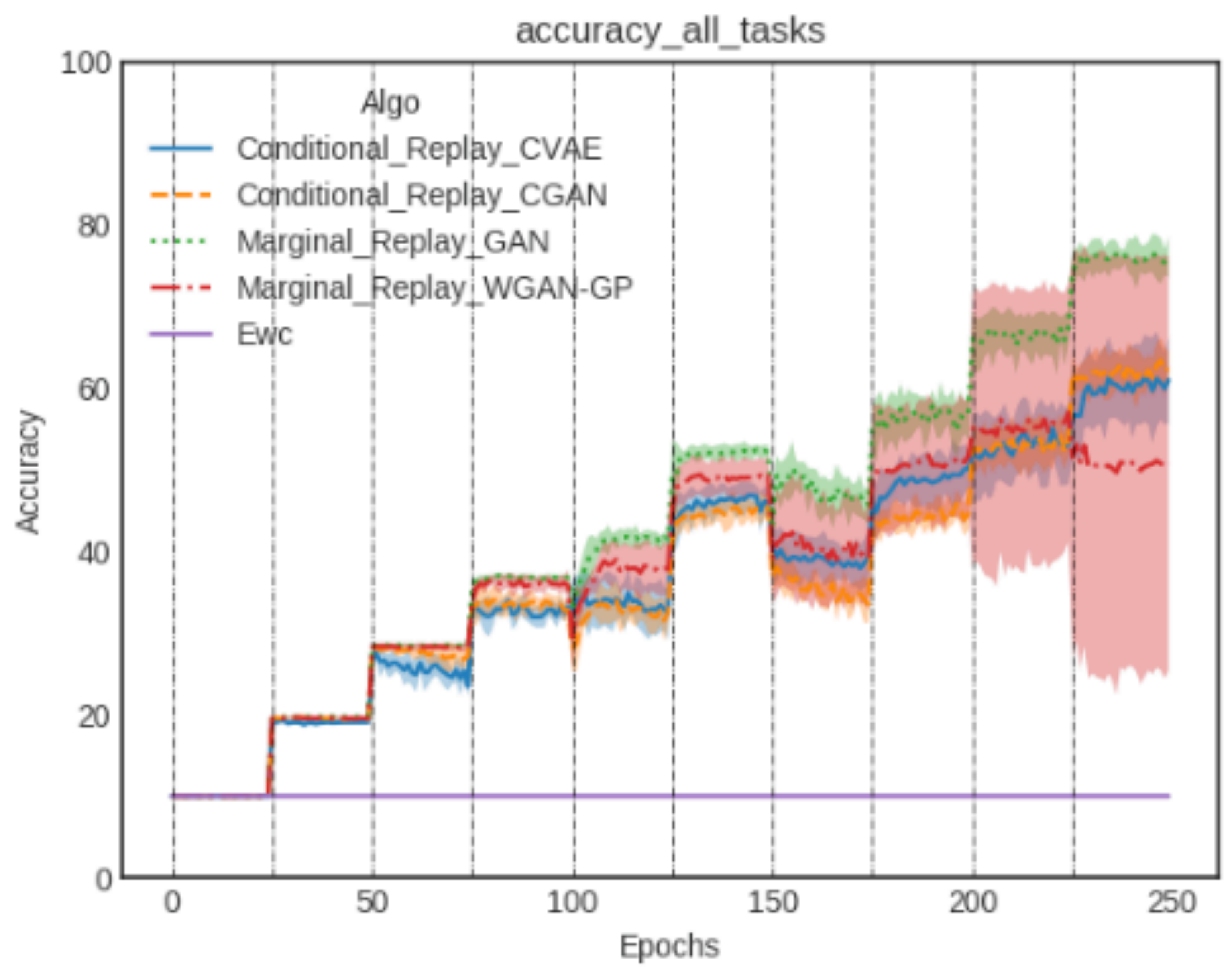
- Algorithms
  - Elastic Weight Constraint (EWC)
  - Marginal replay, conditional replay
- Models
  - Classifier: 2 FC layers with 200 hidden units each, ReLU activated
  - Generator: GAN, WGAN, VAE/ CGAN, CVAE

Observations

- Replay methods outperform EWC across all CLTs
  - EWC completely fails for disjoint classes
- Importance of bringing in past data



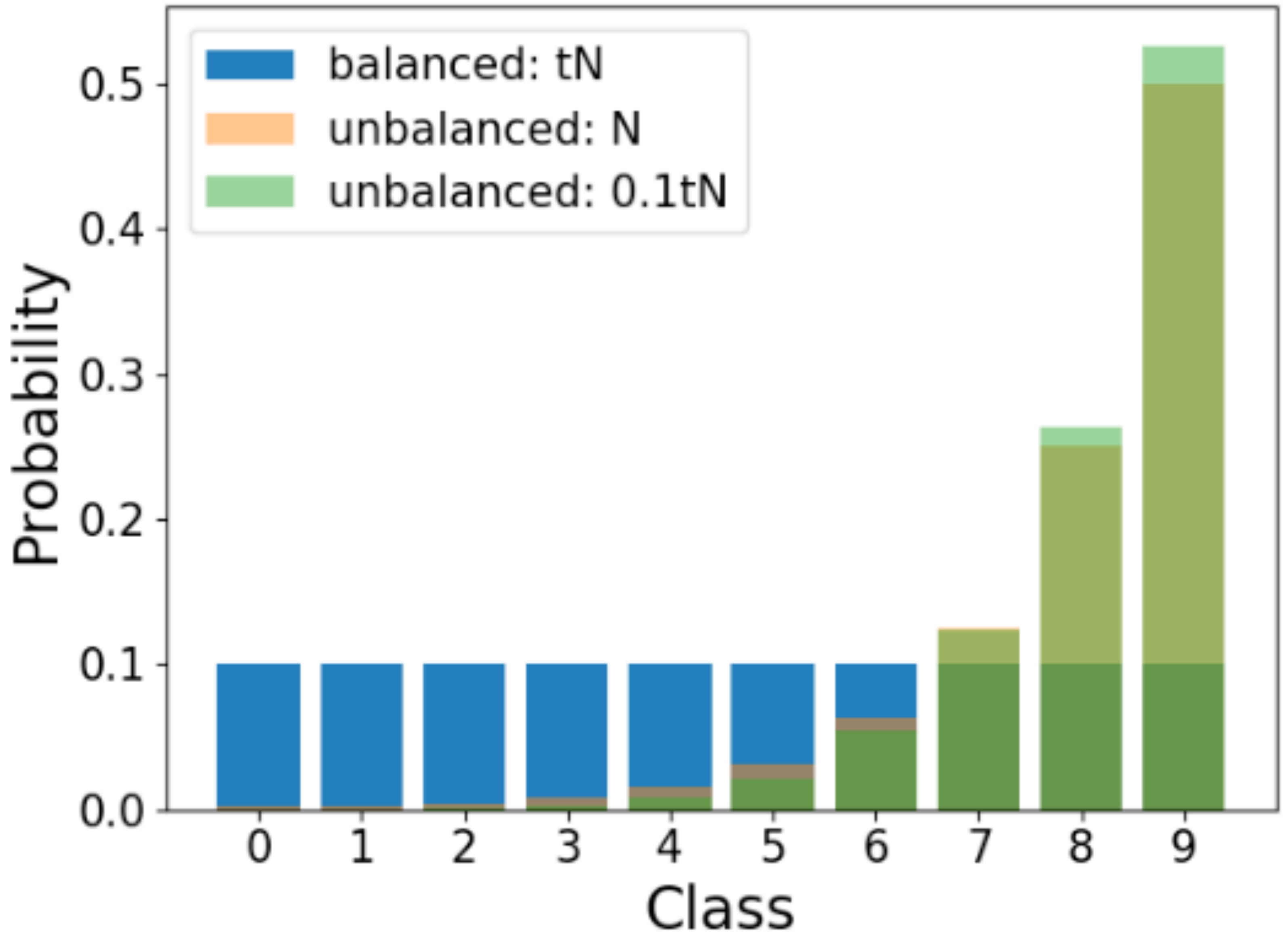
(a) accuracy for MNIST disjoint CLT



(b) accuracy for Fashion MNIST disjoint CLT

Observations

- Marginal replay requires time/memory complexity linear to the number of tasks
  - Unbalanced class distribution
    - Unconditioned generator reproduces the training set distribution



Observations

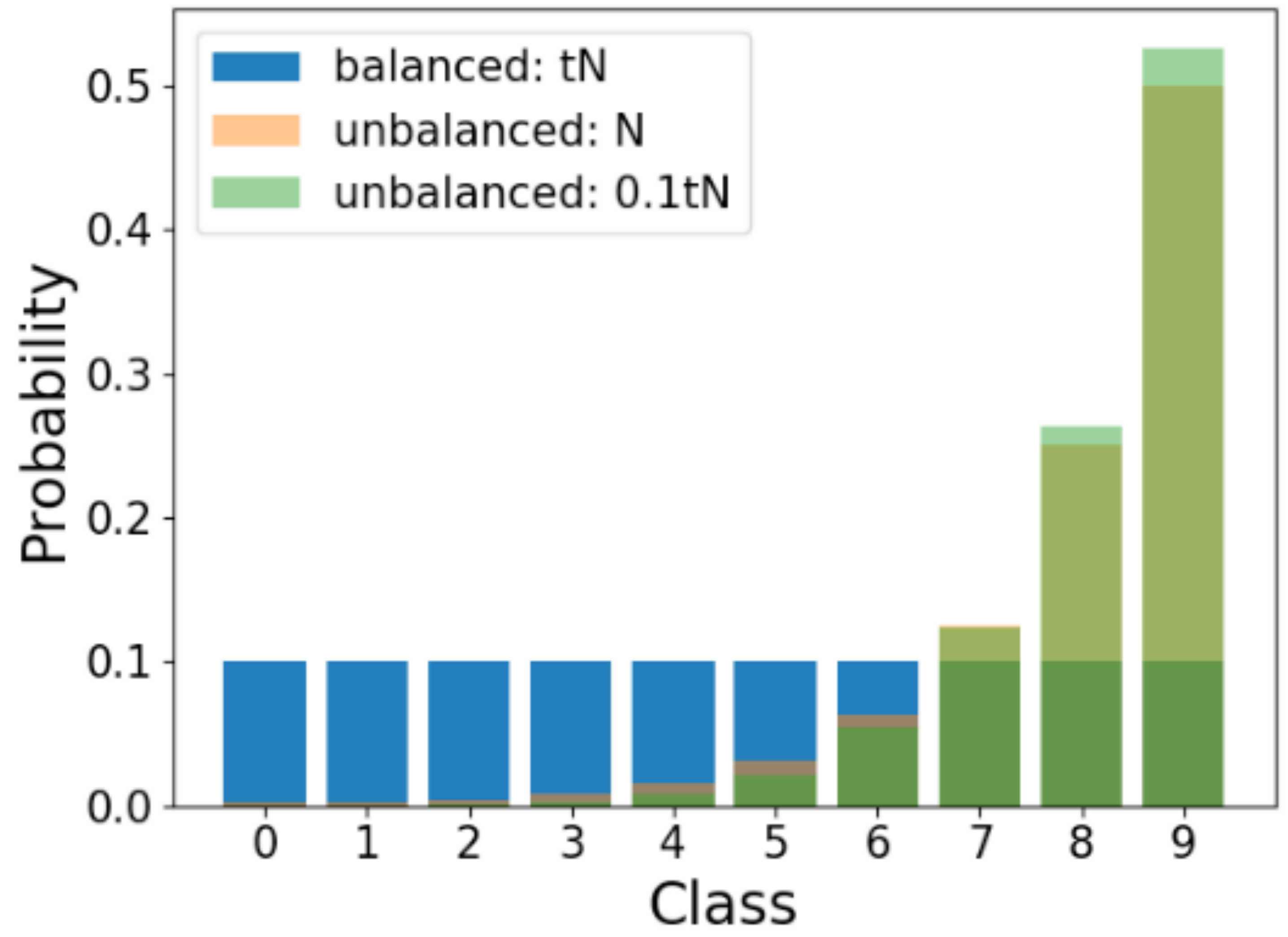
- Suppose t preceding tasks (in disjoint class settings)
- The current task contains with N training samples
- Assuming G_t generates class-balanced samples…
- Case 1: generating tN samples for replay
  - expected number of samples for each previous task: N
  - $\mathscr{D}_{t+1} \cup \mathscr{D}^{rep}_{1:t}$ is class-balanced
  - G_{t+1} likely to generate class-balanced samples

Observations

- Suppose t preceding tasks (in disjoint class settings)
- The current task contains with N training samples
- Assuming G_t generates class-balanced samples...
- Case 2: generating N samples for replay
  - expected number of samples for each previous task: $\dfrac{N}{t}$
  - $\mathcal{D}_{t+1} \cup \mathcal{D}_{1:t}^{rep}$ is not class-balanced
  - G_{t+1} more likely to generate samples from $\mathcal{D}_{t+1}$

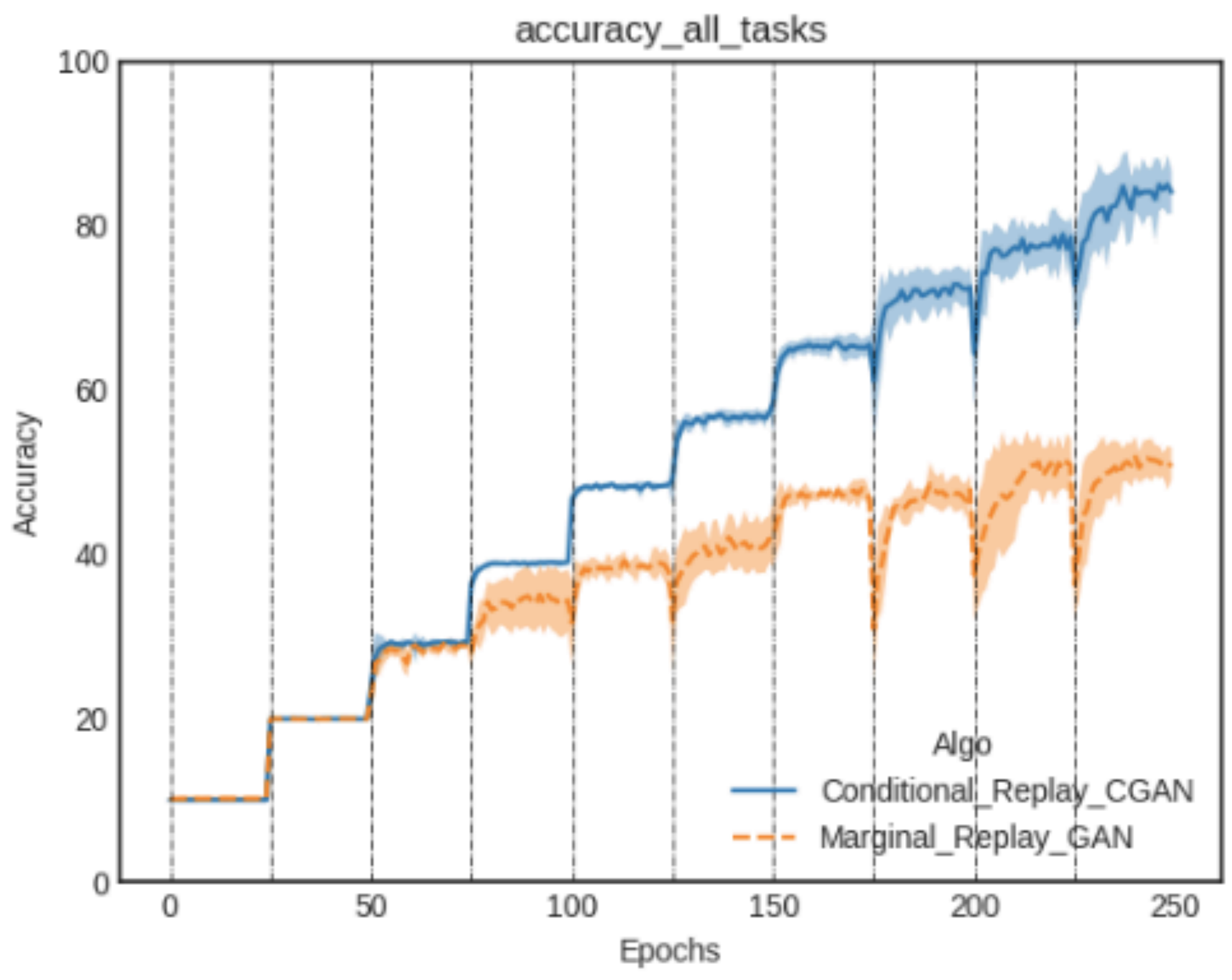
Observations

- Marginal replay requires time/memory complexity linear to the number of tasks
  - Unbalanced class distribution
    - Unconditioned generator reproduces the training set distribution
    - Conditional generator controlled by conditioning vector

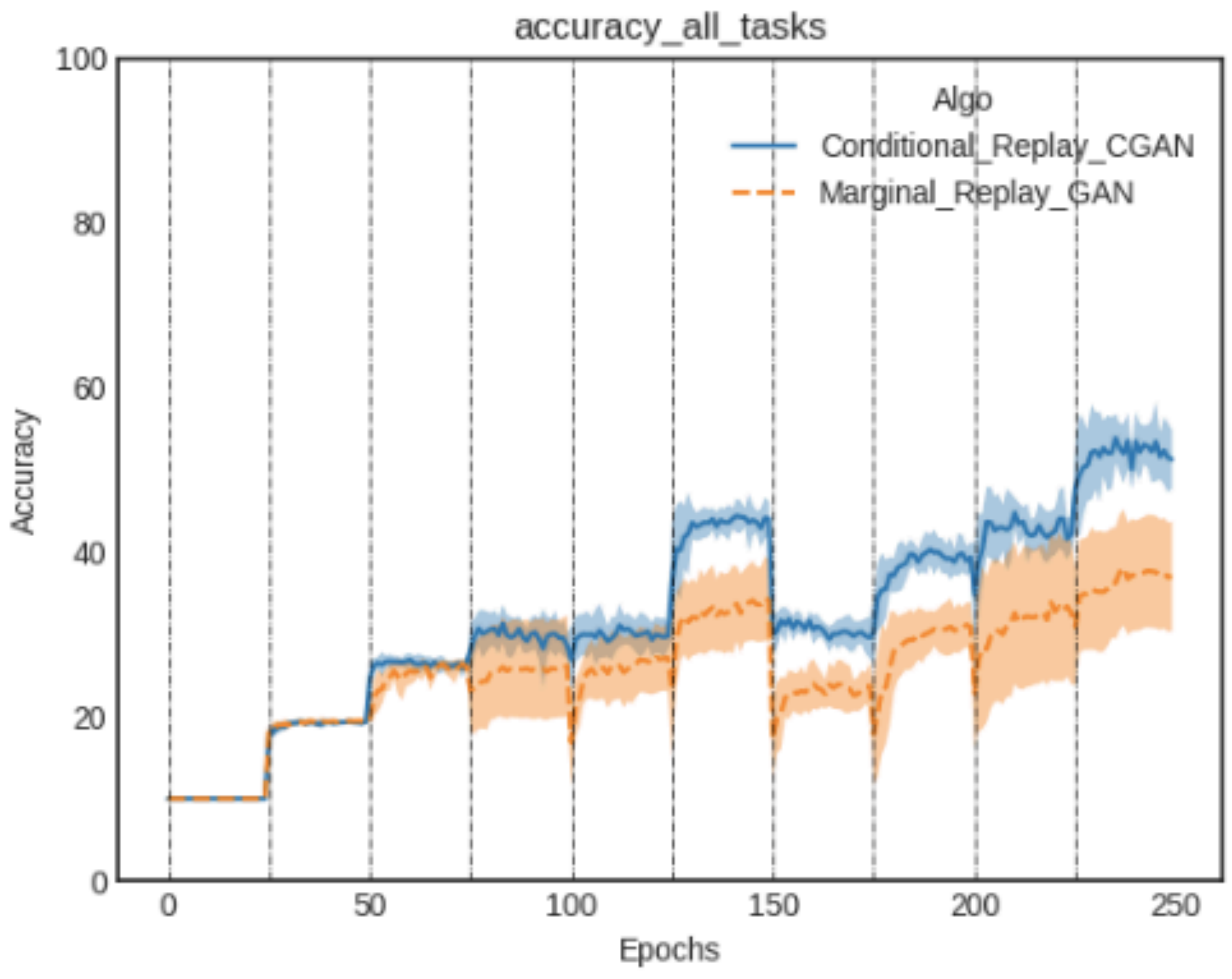


Observations

- With memory constraint, conditional replay is superior



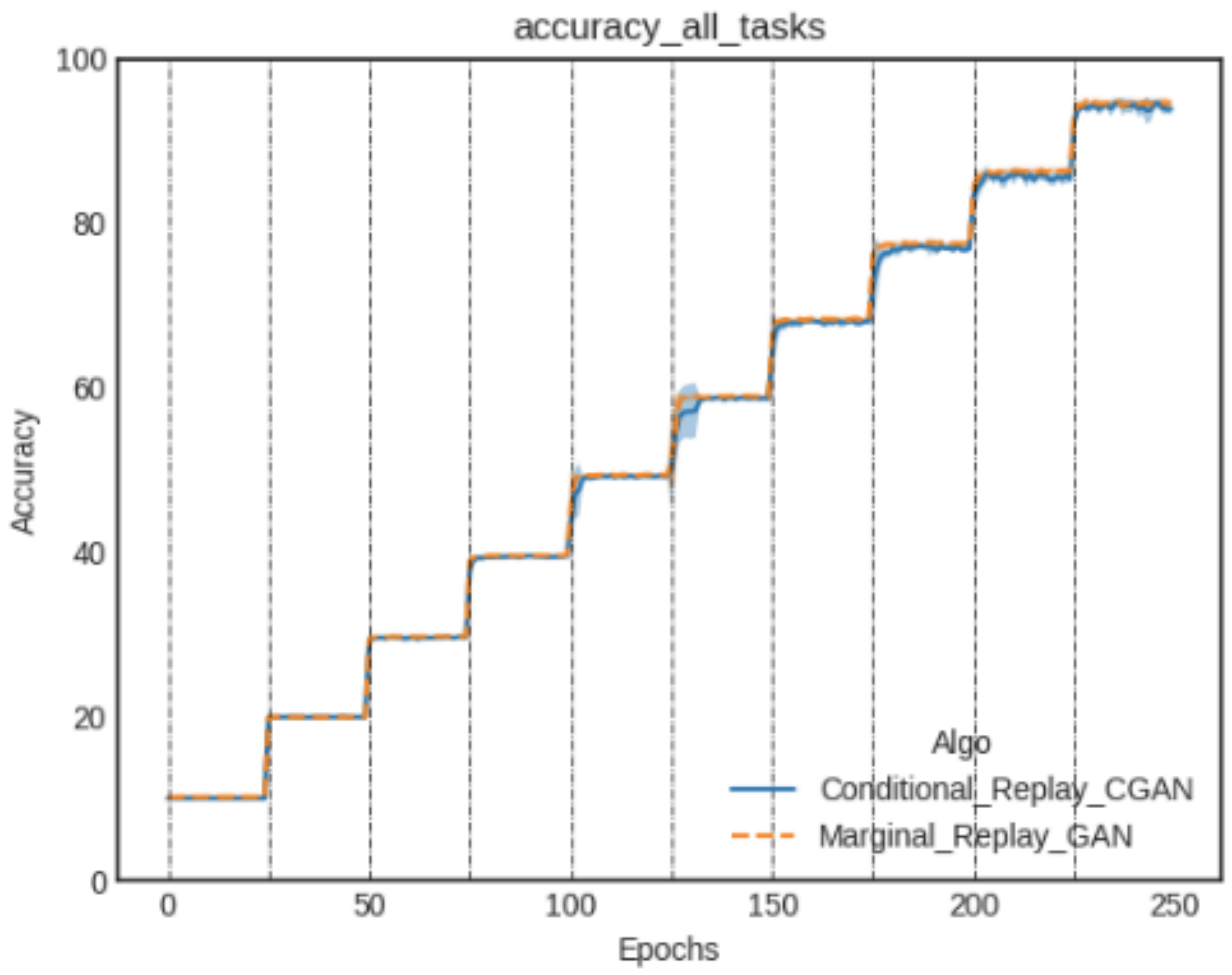
(a) Unbalanced MNIST Disjoint



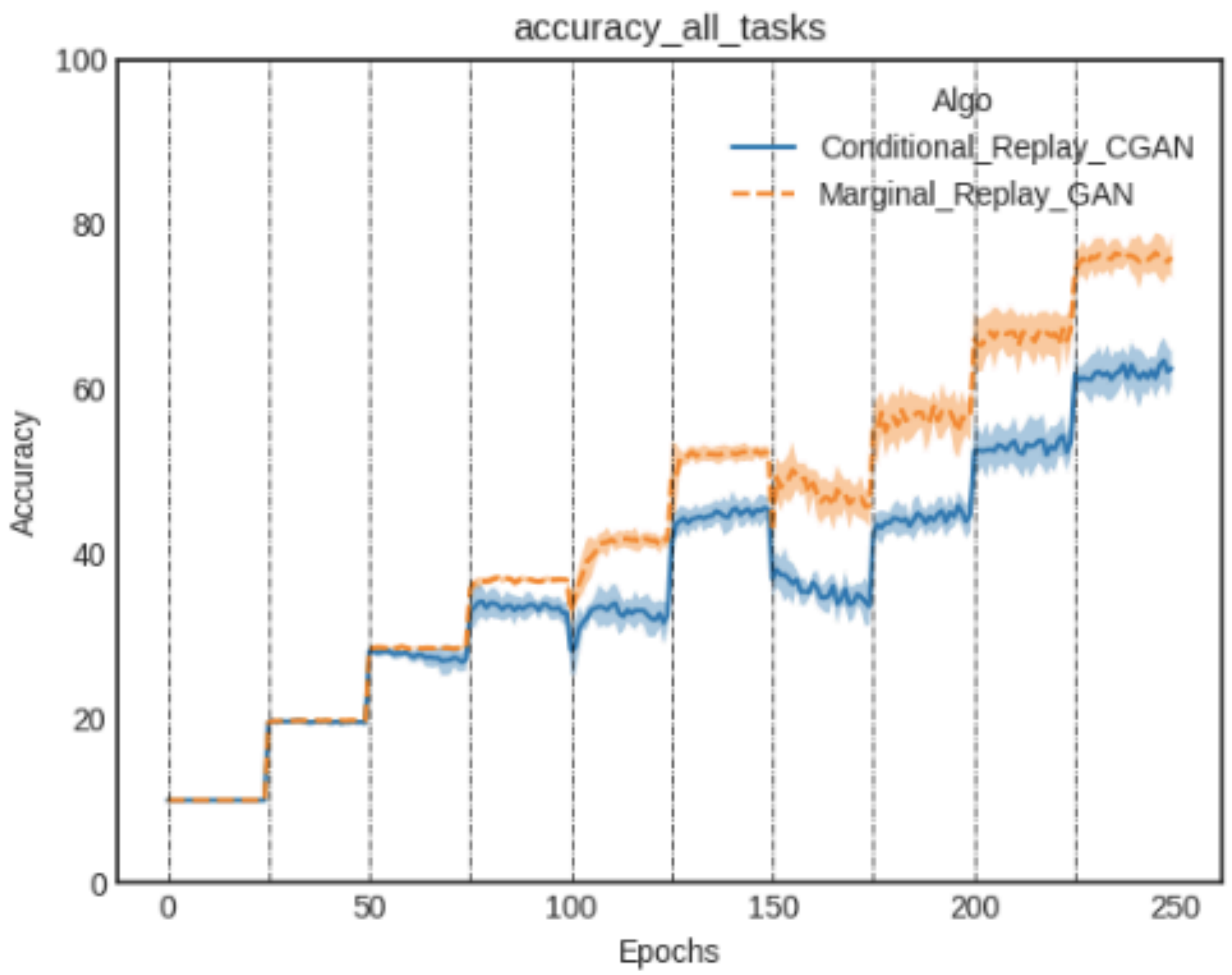
(b) Unbalanced Fashion Disjoint

Observations

- Without memory constraint, marginal replay performs better than conditional replay



(c) Balanced MNIST Disjoint



(d) Balanced Fashion Disjoint

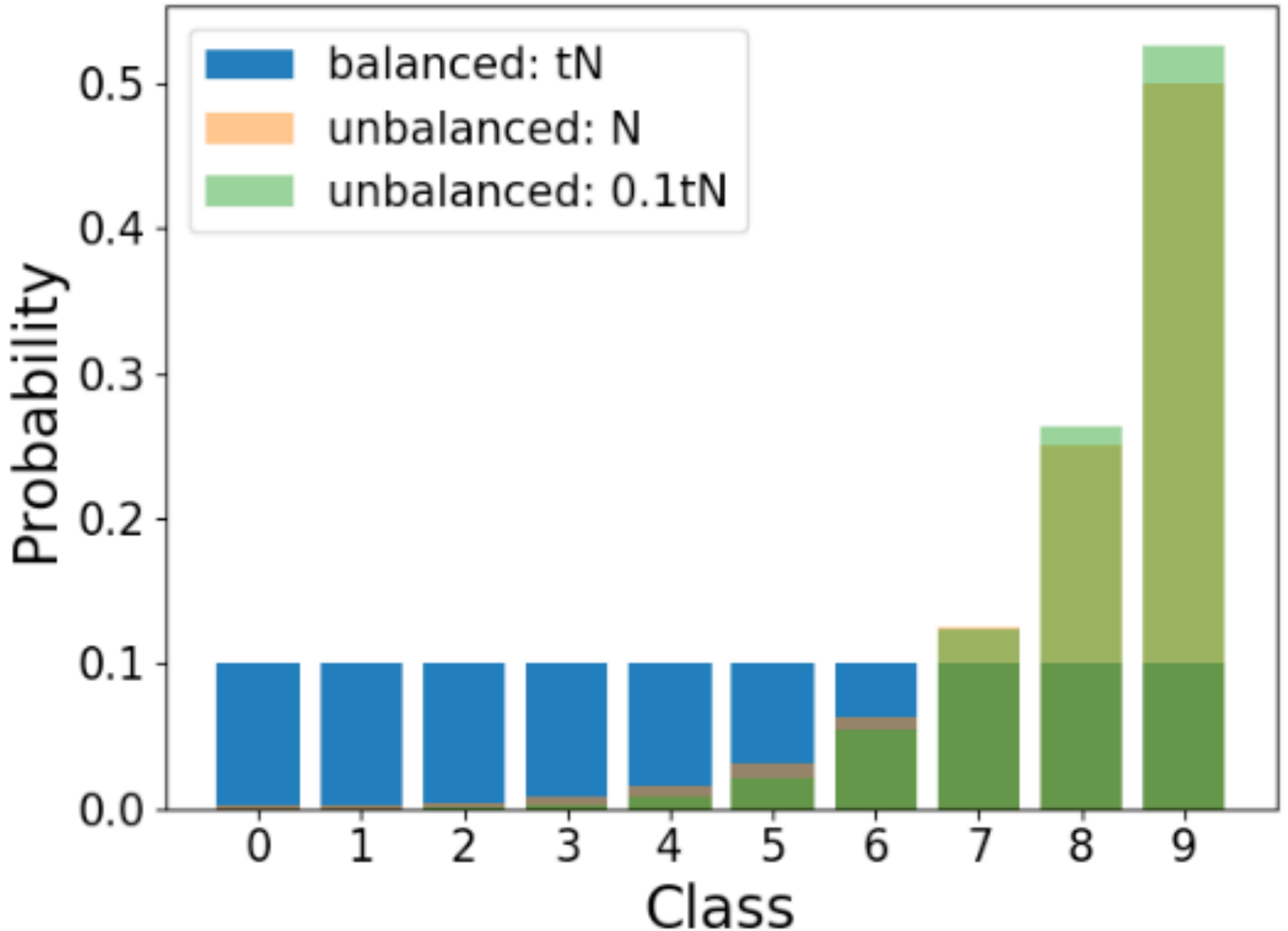
Contributions & Takeaways

- Introduces the use of conditional generators in CLT
- Generative replay outperforms regularization methods
- Disjoint CLTs is still challenging
  - No between-class discrimination from training set
- Conditional replay is more efficient

Still some concerns…

Discussions

- For marginal replay, how to resolve unbalanced class distribution without generating a lot of samples?
  - Claim: tendency to reproduce the distribution it sees at training
  - (Implicit) assumption: each sample is weighted equally



Discussions

$$\mathcal{L}_t = \sum_{x \in \mathcal{D}_{1:t-1}^{replay}} \mathcal{L}_{gen}(x) + \sum_{x \in \mathcal{D}_t} \mathcal{L}_{gen}(x)$$

$$|\mathcal{D}_{1:t-1}^{replay}| = (t-1)|\mathcal{D}_t|$$

(before: with equal weights)

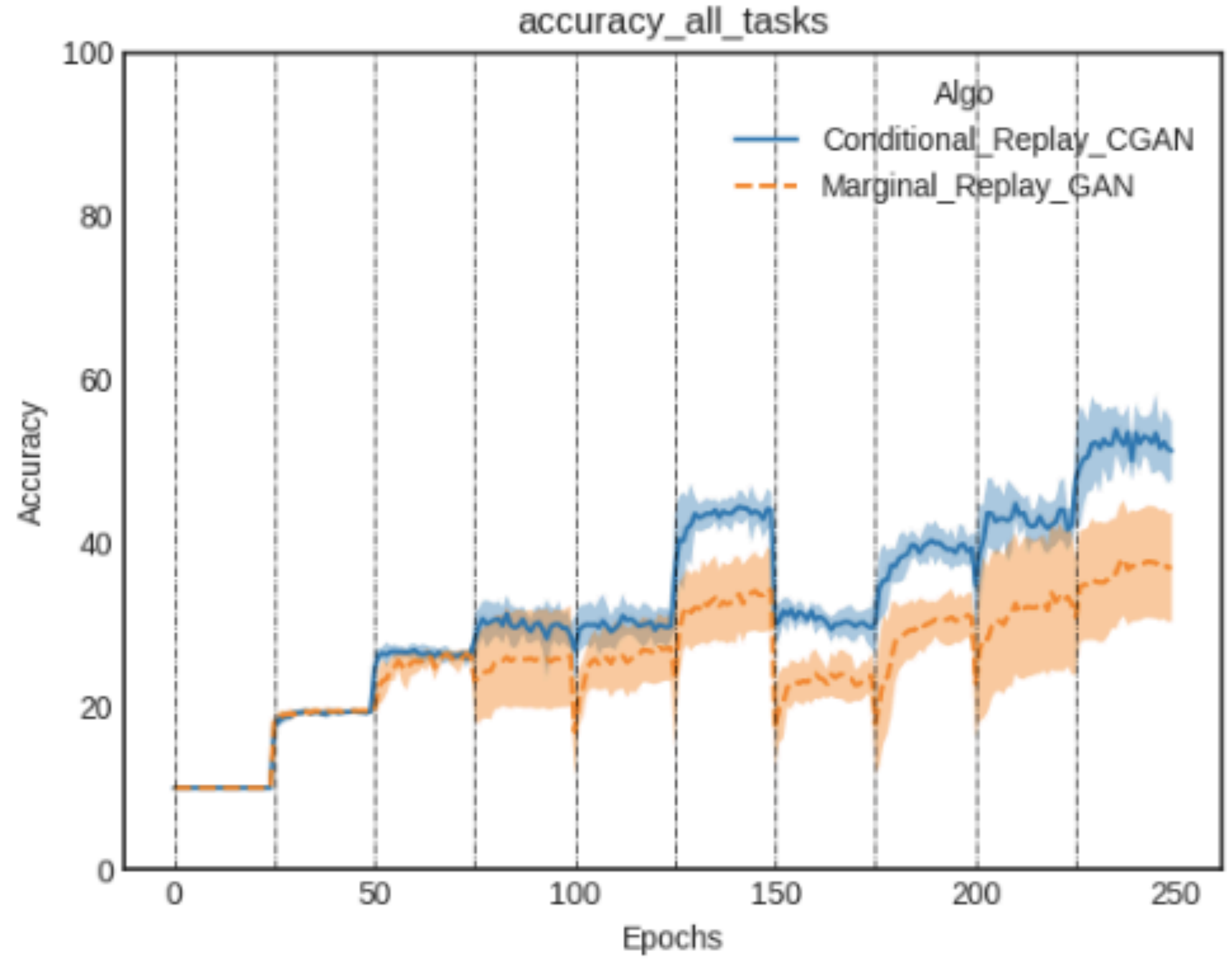
$$\mathcal{L}_t = (t-1) \sum_{x \in \mathcal{D}_{1:t-1}^{replay}} \mathcal{L}_{gen}(x) + \sum_{x \in \mathcal{D}_t} \mathcal{L}_{gen}(x)$$

$$|\mathcal{D}_{1:t-1}^{replay}| = |\mathcal{D}_t|$$

(proposed: with adjusted weights)

Discussions

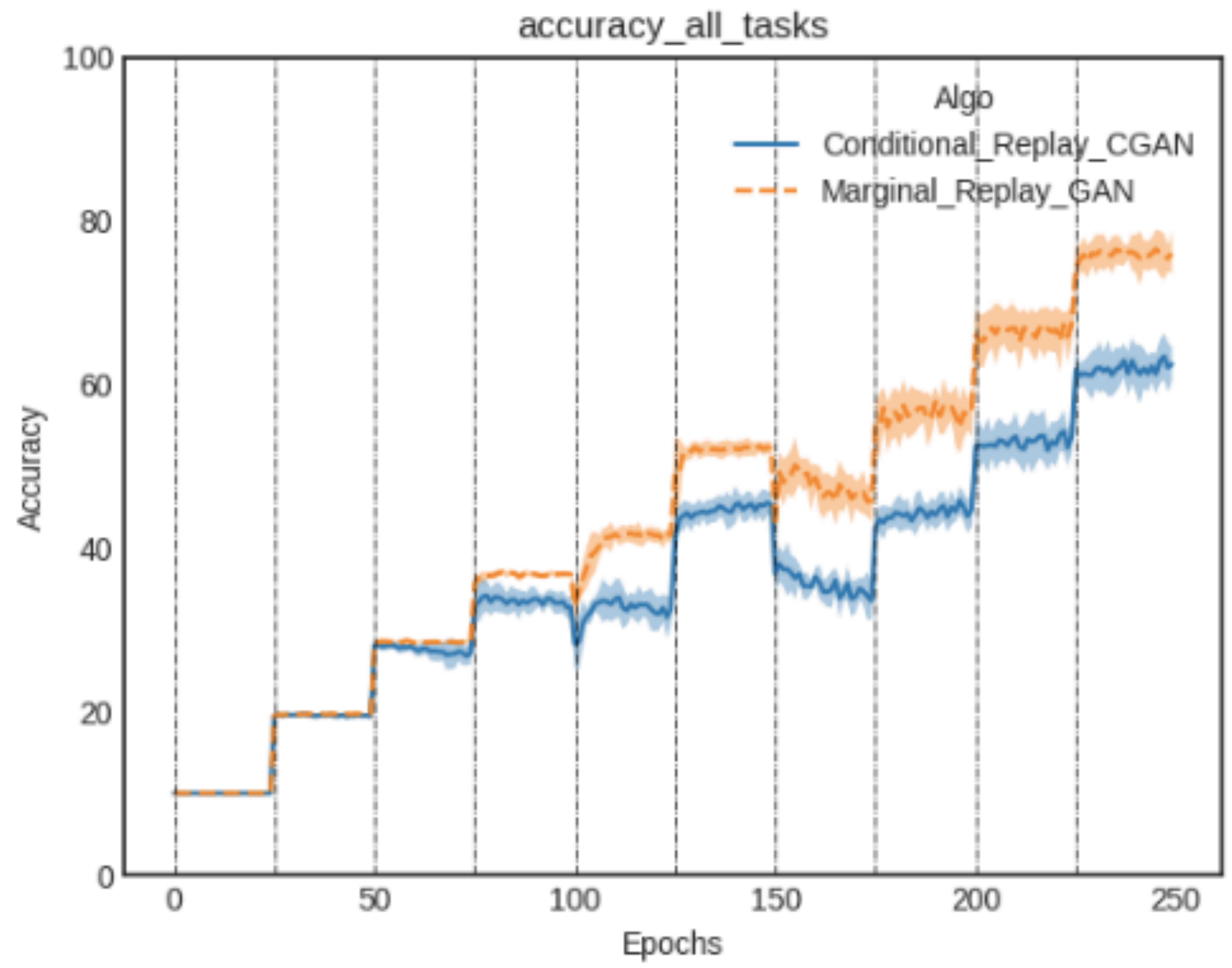
- Low test accuracy for conditional replay with memory constraint
  - Memory constraint -> unbalanced class distribution
  - Impact on the training of classifier?
  - Other metrics such as F1 for each class?
  - Again, weight adjustment?



(b) Unbalanced Fashion Disjoint

Discussions

- Poor accuracy for conditional replay without memory constraint
  - No issue of unbalanced class distribution
  - Conditional generator may produce things not as desired
  - Bring back the classifier?



(d) Balanced Fashion Disjoint

(Improved?) Conditional Replay

- Algorithms

train C_1, G_1 from $\mathcal{D}_1 = (\mathcal{X}_1, \mathcal{Y}_1)$

for t = 2…T

generate $\mathcal{X}^{rep}_{1:t-1}$ from G_{t-1} conditioned on $\mathcal{Y}^{cond}_{1:t-1}$

generate $\mathcal{Y}^{rep}_{1:t-1}$ from C_{t-1}

keep the samples for which $\mathcal{Y}^{rep}_{1:t-1}$ and $\mathcal{Y}^{cond}_{1:t-1}$ agree

train C_t, G_t from $\mathcal{D}_t \cup \mathcal{D}^{rep}_{1:t-1}$

store C_t, G_t and discard C_{t-1}, G_{t-1}

Discussions

- Other continual learning strategies?
  - Rehearsal: select a subset of data as buffer
  - How does generative replay compare with rehearsal methods?
    - More memory required to store data buffer than a generator
    - Will generative replay achieve better performance?

Further Questions

Appendix

Elastic Weight Constraint (EWC)

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2$$

- $\mathcal{L}_B(\theta)$ is the objective of the current task, B
- $\theta^*_{A,i}$ is the weights optimized for previous task A
- F (Fisher information matrix):

$$F = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log p(x_i \mid \theta) \, \nabla_\theta \log p(x_i \mid \theta)^\top$$

GAN vs WGAN

- GAN has a more stable performance than WGAN
- Objectives for GAN:

$$\min_G \max_D \mathbb{E}_x[\log(D(x)] + \mathbb{E}_z[\log(1 - D(G(z)))]$$

- Objectives for WGAN:

$$\min_G \max_{\|D\|_{\text{Lip}\leq 1}} \mathbb{E}_x[D(x)] - \mathbb{E}_z[D(G(z))]$$

  - 1-Lipschitz approximated with gradient penalty
  - $\lambda(\|\nabla_{\hat{x}} D(\hat{x})\|_2^2 - 1)$ with $\hat{x} = tx + (1-t)G(z), 0 \leq t \leq 1$

VAE vs CVAE

- VAE: encoder $q(z \mid x)$ and decoder $p(x \mid z)$

$$\log p(x) \geq \mathbb{E}_{z|x}[p(x \mid z)] - D_{KL}(q(z \mid x) \,||\, p(z))$$

- CVAE: encoder $q(z \mid x, c)$ and decoder $p(x \mid z, c)$

$$\log p(x \mid c) \geq \mathbb{E}_{z|x,c}[p(x \mid z, c)] - D_{KL}(q(z \mid x, c) \,||\, p(z \mid c))$$