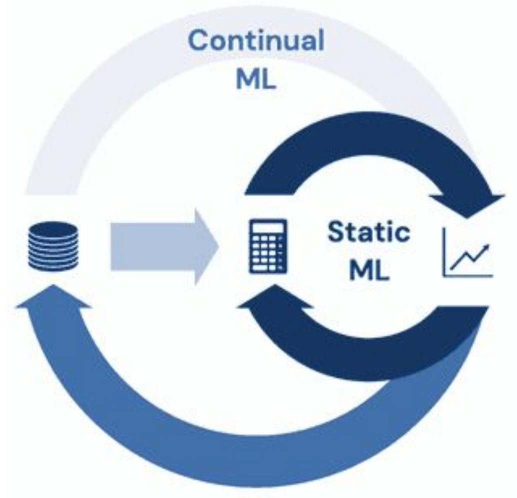
Continual Learning-Based MIMO Channel Estimation: A Benchmarking Study

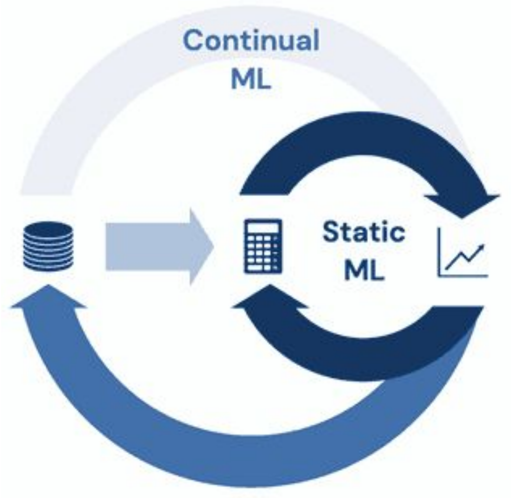
Mohamed Akrout, Amal Feriani,
Faouzi Bellili, Amine Mezghani & Ekram Hossain

June 1st 2023, Rome, Italy

Outline

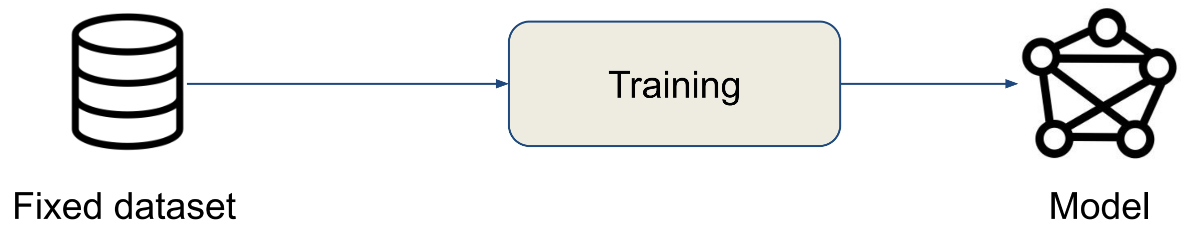


Outline



Motivation behind Continual Learning

- Traditional machine learning: Isolated single-task learning



- different tasks learned separately and independently
    - no knowledge accumulation or sharing
    - model fixed after deployment

- We, humans, never learn in isolation. We learn continually and accumulate knowledge over time.

- Challenge: **catastrophic forgetting**. When learning a new task, the DNN changes its parameters, which may cause deterioration in performance of previous tasks.

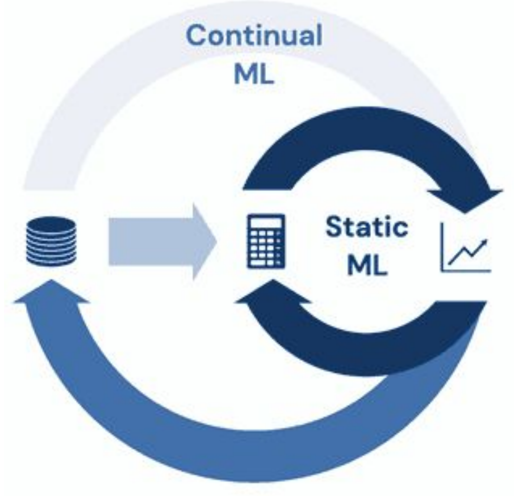
Motivation behind Continual Learning

- Need for a balance between adapting to new data and retaining knowledge acquired from old data:
    - high plasticity leads to catastrophic forgetting
    - low plasticity leads to limited adaptation

- Adaptation (across tasks) is measured using:
    - *Forward* transfer: learning from old tasks helps future task learning
    - *Backward* transfer: learning from future tasks helps improve previous task learning.

- Classes of CL in this paper:
    - **task-incremental learning (Task-IL)**: the identity of the task to be always provided which makes task-specific training possible.
    - **domain-incremental learning (Domain-IL)**: the task identity is not known at inference time.

Outline



Continual learning methods

Three CL techniques

- **Regularization methods**:
  - regularization term in the loss function
  - no buffer memory is used during training
- **Rehearsal methods**:
  - use memory buffers to store samples from previous tasks
  - revisit stored samples during learning (as input to the DNN)
  - constrain the loss function to prevent task interference

$\Bigg\}$ The focus of this paper

- **Architecture-based methods**:
  - append new model parameters to each task
  - each task uses different model parameters while freezing or masking out the parameters of previous tasks

k

Synaptic Intelligence (SI)

- **Type:** regularization method

- **Key idea:** Prevent important parameters for previous tasks to change drastically when learning a new one. If a parameter is important for the task 1, then it will try to stay as close as possible to its value when learning a new task 2

  ⇒ the parameter will converge to a reference value that achieves good performance on all tasks and not on tasks individually

- **Details**:
  - The importance of a parameter k (i.e., weight and bias) is determined by two quantities:
    - a measure proportional to its contribution to the task objective minimization over the entire training trajectory ω_k
    - the distance traveled in the parameter space $\Delta_k^\mu = \theta_k^\mu - \theta_k^{\mu-1}$
  - New surrogate loss for the task μ

$$\tilde{L}_\mu = L_\mu + c\sum_k \Omega_k^\mu \left(\tilde{\theta}_k - \theta_k\right)^2$$

the parameters at the end of the previous task

Per-parameter regularization strength

$$\Omega_k^\mu = \sum_{\nu<\mu} \frac{\omega_k^\nu}{(\Delta_k^\nu)^2 + \xi}$$

damping parameter to avoid division by 0

k

Gradient Episodic Memory (GEM)

- **Type:** replay-based, constraint optimization
- **Key idea:** constrain the new task updates to not interfere with previous tasks using samples stored in episodic memory
- **Details**:
  - The losses on previous tasks estimated using the samples in the memory are treated as inequality constraints when learning a new task

$$\text{minimize}_\theta \quad \ell(f_\theta, \mathcal{D}_t) \quad \text{s.t.} \quad \ell(f_\theta, \mathcal{M}_k) \leq \ell(f_\theta^{t-1}, \mathcal{M}_k) \qquad \forall k < t$$

training dataset for the current task t **NN trained till task t-1** **episodic memory**

→ The inequality constraint avoids loss increase for previous tasks while allowing its decrease (better backward transfer)

  - Projects the proposed gradient direction for the current task **g** when the angle between the **g** and the gradient vectors of previous tasks is greater than 90 by solving the following QP:

$$\text{minimize}_{\tilde{g}} \quad \frac{1}{2}||g - \tilde{g}||_2^2 \quad \text{s.t.} \quad \langle \tilde{g}, g_k \rangle \geq 0 \qquad \forall k < t$$

proposed gradient direction for the task t **projected gradient direction**

Averaged-GEM (A-GEM, A-GEM-R)

- **Type:** replay-based, constraint optimization
- **Key idea:** improves on GEM by averaging over the episodic memory
- **Details of A-GEM**:
    - add one constraint on the **average** loss over the previous tasks instead of each previous task individually (i.e., 1 constraint instead of t-1 constraints in GEM)

$$\text{minimize}_\theta \quad \ell(f_\theta, \mathcal{D}_t) \quad \text{s.t.} \quad \ell(f_\theta, \mathcal{M}) \leq \ell(f_\theta^{t-1}, \mathcal{M}) \qquad \text{where } \mathcal{M} = \cup_{k<t} \mathcal{M}_k$$

training dataset for the current task t **NN trained till task t-1** **episodic memory**

- If the proposed gradient for the current task g violates the constraint, it is projected as follows:

$$\tilde{g} = g - \frac{g^\top g_{ref}}{g_{ref}^\top g_{ref}} g_{ref}$$

where g_{ref} is the gradient computed using a batch randomly sampled from the memory $\mathcal{M}$

k

Averaged-GEM (A-GEM, A-GEM-R)

- **Type:** replay-based, constraint optimization
- **Key idea:** improves on A-GEM by using **reservoir sampling technique and eliminating task boundaries**
- **Details of A-GEM-R**:
  - A-GEM-R is A-GEM with a **reservoir buffer**
  - A-GEM relies on task boundaries to store the samples in the memory buffer
  - A-GEM-R extends A-GEM to have a memory buffer from which it obtain batch samples without **task boundaries**

Experience Replay (ER)

- **Type:** replay-based
- **Key idea:** Retrain the model on samples from the current tasks and a subset of samples from previous tasks
- **Cons**:
    - Prone to overfitting to the subset of stored examples

Dark Experience Replay (DER, DER++)

- **Type:** replay-based
- **Key idea:** use distillation and regularization to minimize the difference between the current and previous model outputs (i.e., logits)
- **Details**:
  - Rely on dark knowledge [1] for distilling past experiences, sampled over the entire training trajectory
  - Store the model logits (pre-softmax) instead of labels
  - Reservoir sampling is used to populate the buffer
  - The loss for the current task

DER

$$\mathcal{L}_{t_c} + \alpha\, \mathbb{E}_{(x,z)\sim\mathcal{M}}\left[\, \|z - h_\theta(x)\|_2^2 \,\right]$$

Memory **Previous network logits for x** **current network logits for x (before softmax)** **current NN output (after softmax)**

DER++

$$\mathcal{L}_{t_c} + \alpha\, \mathbb{E}_{(x',y',z')\sim\mathcal{M}}\left[\, \|z' - h_\theta(x')\|_2^2 \,\right] + \underbrace{\beta\, \mathbb{E}_{(x'',y'',z'')\sim\mathcal{M}}\left[\ell(y'', f_\theta(x''))\right]}_{\text{additional penalty term using the ground truth}}$$

[1] - Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Dark Knowledge. https://www.ttic.edu/dl/dark14.pdf

Function Distance Regularization (FDR)

- **Type:** replay-based
- **Key idea:** Align previous and current outputs in function space using an L2-regularization loss
- **Details**:
  - Saves network outputs at task boundaries

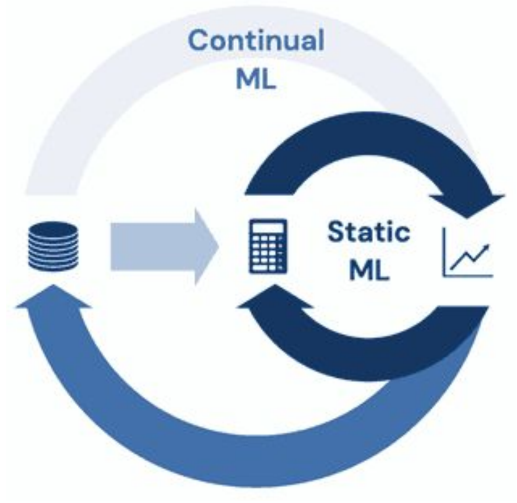
$$L(\theta) = L_B(\theta) + \frac{\lambda}{2}\|f_{\theta_A} - f_{\theta_B}\|$$

function after training the task A (i.e, NN)

current function (i.e, NN)

- **Cons**:
  - Stores data at task boundaries

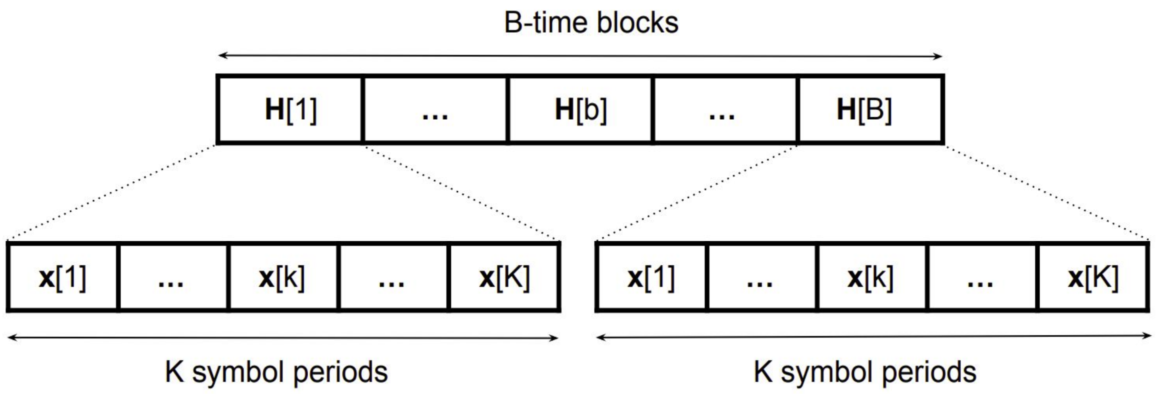
Outline



Continual learning for channel estimation

Channel model

- Gauss-Markov Rayleigh fading MIMO channel between N transmit antennas and M receive antennas.
- The transmission for B time blocks each of which is used to transmit K symbols.
- The block-fading channel is assumed to remain constant over the entire b-th time block, while the total duration of KB symbol periods is used for channel estimation within the overestimated channel coherence time Tc.



Continual learning for channel estimation

University of Manitoba

Channel model

- For every time-block b, the received signal is given by:

$$\boldsymbol{y}_b = \boldsymbol{H}_b \boldsymbol{x}_b + \boldsymbol{n}_b, \qquad b = 1, \ldots, B$$

with

$\boldsymbol{x}_b \in \mathbb{C}^N$ is the transmitted symbol vector

$\boldsymbol{n}_b \in \mathbb{C}^M$ is the additive complete white Gaussian noise, i.e., $\boldsymbol{n}_b \sim \mathcal{CN}(\mathbf{0}_M, \sigma^2_{\boldsymbol{n}_b} \mathbf{I}_M)$

- At a given SNR ρ, the noise variance can be controlled as

$$\sigma^2_{\boldsymbol{n}_b} = \mathbb{E}\left[\|\boldsymbol{H}_b \boldsymbol{x}_b\|_2^2 \right] 10^{-\rho/10}$$

Continual learning for channel estimation

Channel model

- The time-varying Rayleigh fading channel evolves from one block to another according to the Gauss-Markov model:

$$\boldsymbol{H}_b = \sqrt{\alpha}\,\boldsymbol{H}_{b-1} + \sqrt{1-\alpha}\,\boldsymbol{W}_b \qquad \text{for} \quad b = 2, \ldots, B$$

with

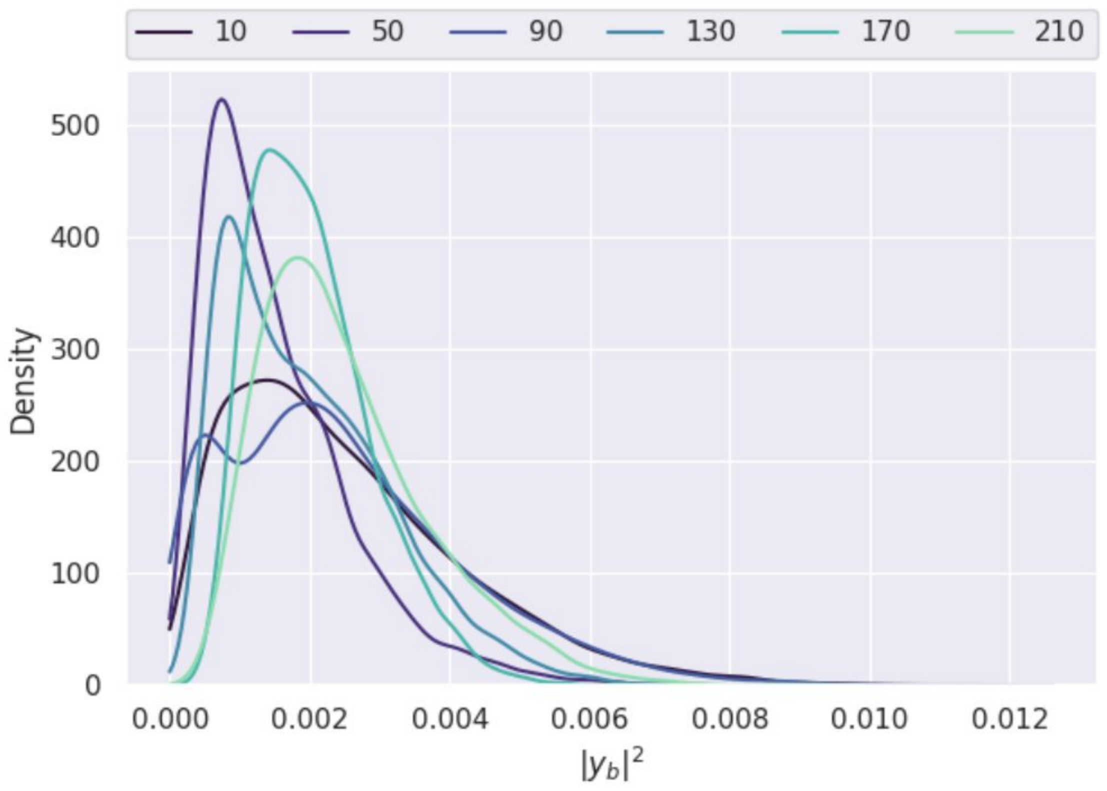
$\alpha \in [0, 1[$ is the channel memory factor.

- The channel can be written in a explicit non-recursive form as:

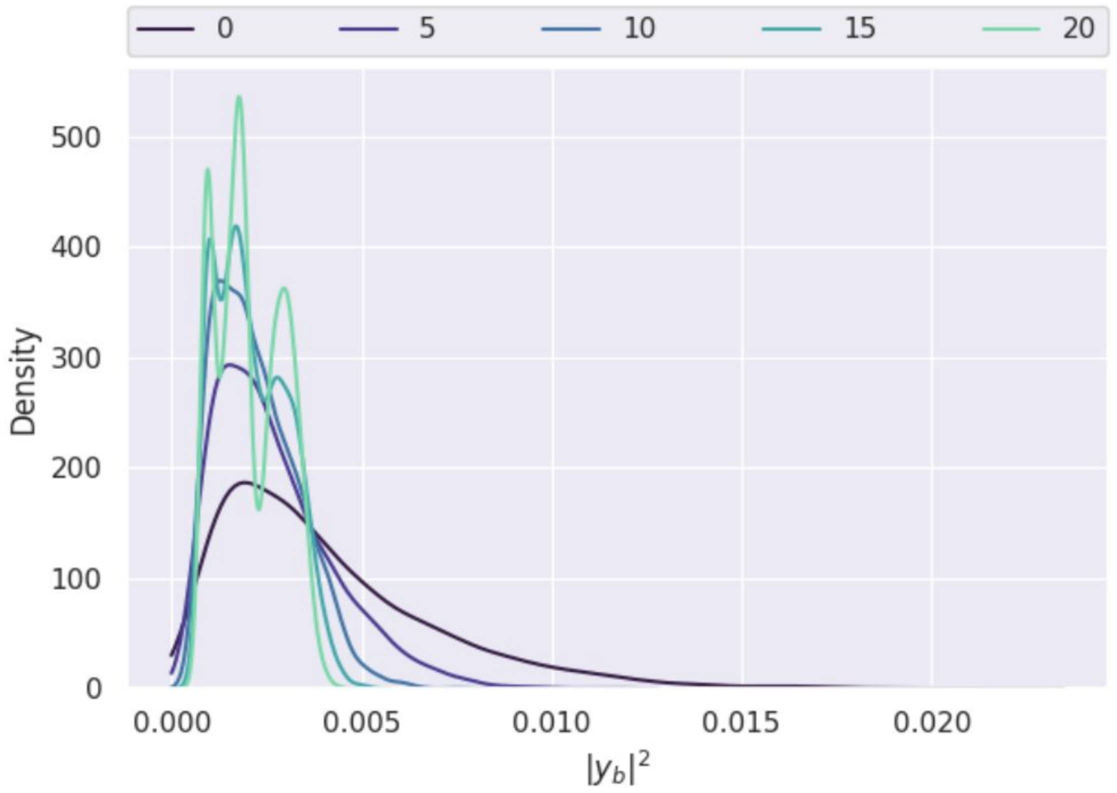
$$\boldsymbol{H}_b = \underbrace{\sqrt{\alpha}^{\,b-1}\,\boldsymbol{H}_1}_{\text{initial value}} + \underbrace{\sqrt{1-\alpha}\sum_{p=2}^{b}\sqrt{\alpha}^{\,b-p}\,\boldsymbol{W}_p}_{\text{exponentially decaying perturbation}}$$

Continual learning for channel estimation

Distribution shift in the power of the received signal



varying the coherence time Tc
at SNR ρ = 10 dB



varying the SNR level ρ
at Tc = 20 symbol periods

Continual learning for channel estimation

Tasks protocols

- We consider two domain-IL tasks:
  - $\mathcal{T}_{\text{SNR}} \triangleq \bigcup_i \{\mathcal{T}_{\text{SNR}=i[\text{dB}]}\}$: channel samples for each SNR level
  - $\mathcal{T}_{T_c} \triangleq \bigcup_j \{\mathcal{T}_{T_c=j}\}$: channel samples for each coherence time Tc
- Note that we consider these tasks as domain-IL since an implicit task-dependent transformation will be applied on the inputs of the channel estimation model (i.e., received signal). However, it is important to highlight that this setting is slightly different from the common assumption in the CL literature where domain-IL tasks are defined using direct transformation on the inputs (e.g., rotation, permutation)

Continual learning for channel estimation

Tasks ordering

- We examine how changing the task order affects the channel estimation.

- Two different task orderings of the sets $\mathcal{T}_{\text{SNR}}$ and $\mathcal{T}_{T_c}$:

    - **Curriculum order (easy to hard)**: from easy to hard to imitates the learning order in human curricula, i.e., from high to low SNR.
    - **Random order**: the tasks are randomly presented to the DNN.

- Task ordering hypothesis: curriculum order leads to a more accurate DNN as compared to the random order of tasks.

Continual learning for channel estimation

Evaluation metrics

- The DNN performance is evaluated **on all previously learned tasks** after it has finished learning a task t_i
- After learning all the T tasks, we obtain the MSE matrix $E \in \mathbb{R}^{T \times T}$ where the element $e_{i,j}$ which represents the MSE of the task t_j after learning the task t_i
- $e_{T,i}$ stands for the error on the task t_i after learning all the T tasks
- Let $\bar{b}$ denote the error vector of a randomly-initialized model

Continual learning for channel estimation

Evaluation metrics

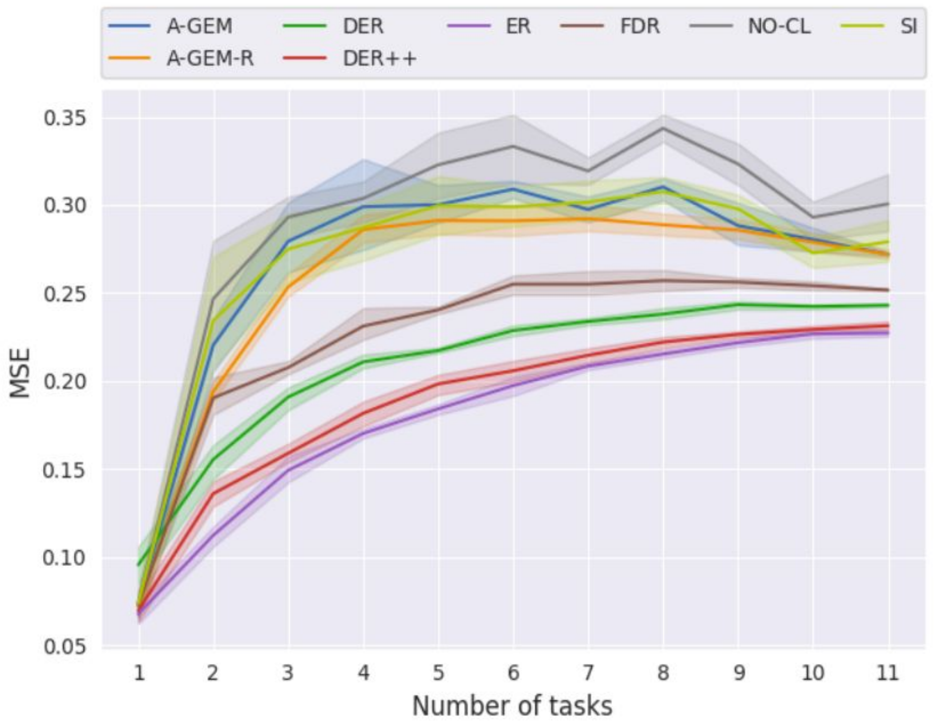
- We assess the ability to transfer knowledge across tasks by measuring:
  - **Average MSE** $(\texttt{AVG\_MSE}) = \frac{1}{T}\sum_{i=1}^{T} e_{T,i},$
  - **Forward transfer** $(\texttt{FWT}) = \frac{1}{T-1}\sum_{i=2}^{T}(\bar{b}_i - e_{i-1,i})$
  - **Backward transfer** $(\texttt{BWT}) = \frac{1}{T-1}\sum_{i=1}^{T-1}(e_{T,i} - e_{i,i})$

$e_{i,j}$	the MSE of the task t_j after learning the task t_i
$e_{T,i}$	the error on the task t_i after learning all the T tasks
$\bar{b}$	the error vector of a randomly-initialized model

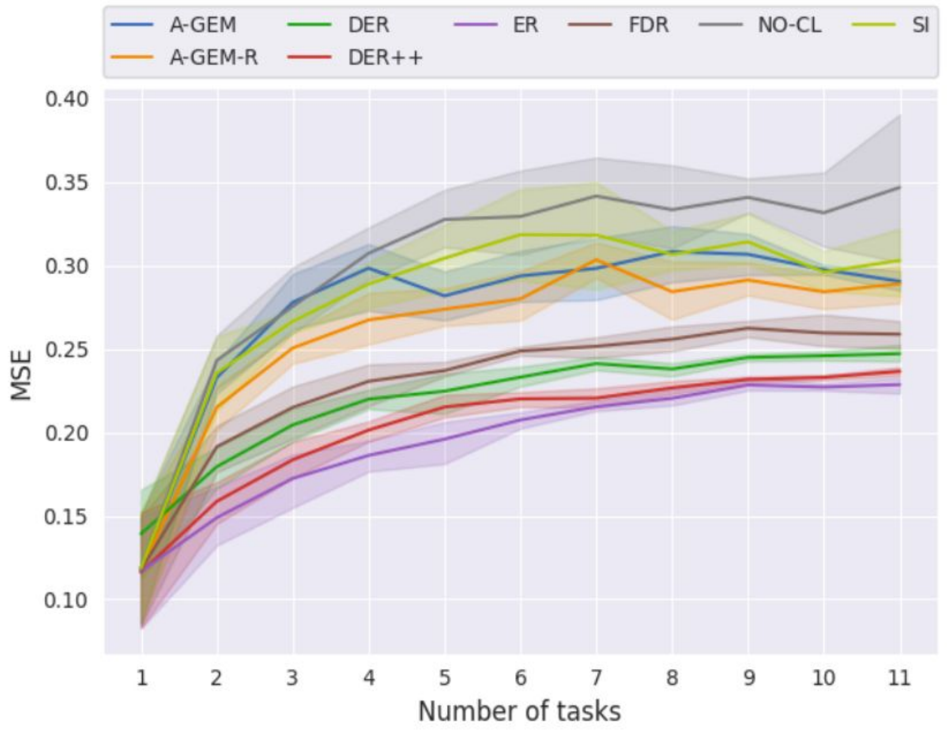
Continual learning for channel estimation

Estimation Accuracy

- The MSE for each one of each CL method when the SNR is varied



$\mathcal{T}_{\text{SNR}}$, curriculum ordering

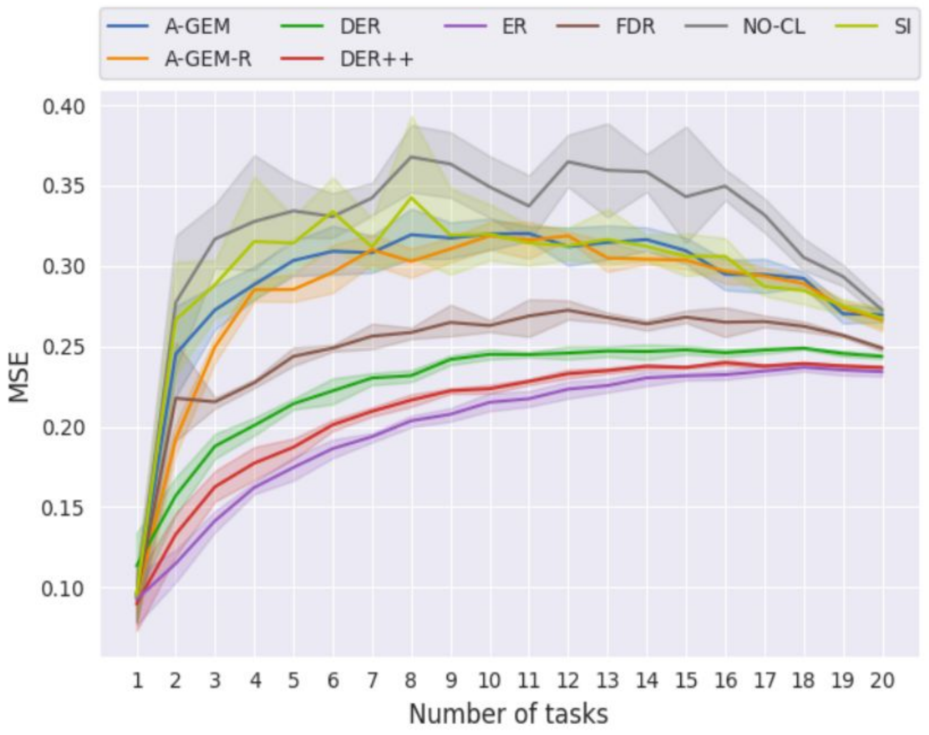


$\mathcal{T}_{\text{SNR}}$, random ordering

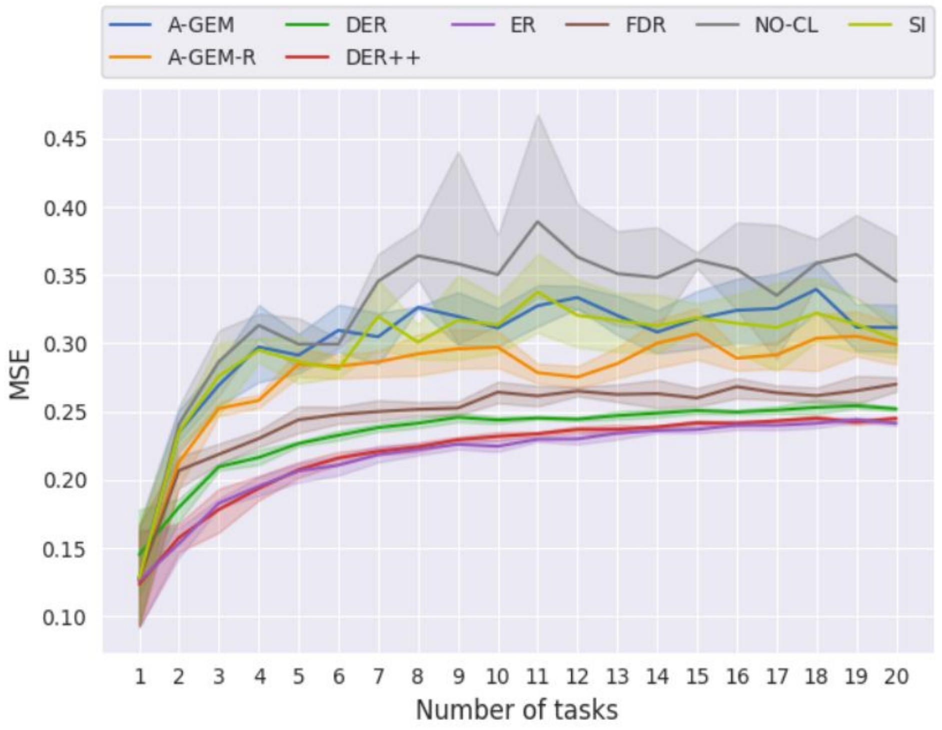
Continual learning for channel estimation

Estimation Accuracy

- The MSE for each one of each CL method when the coherence time is varied



$\mathcal{T}_{T_c}$, curriculum ordering

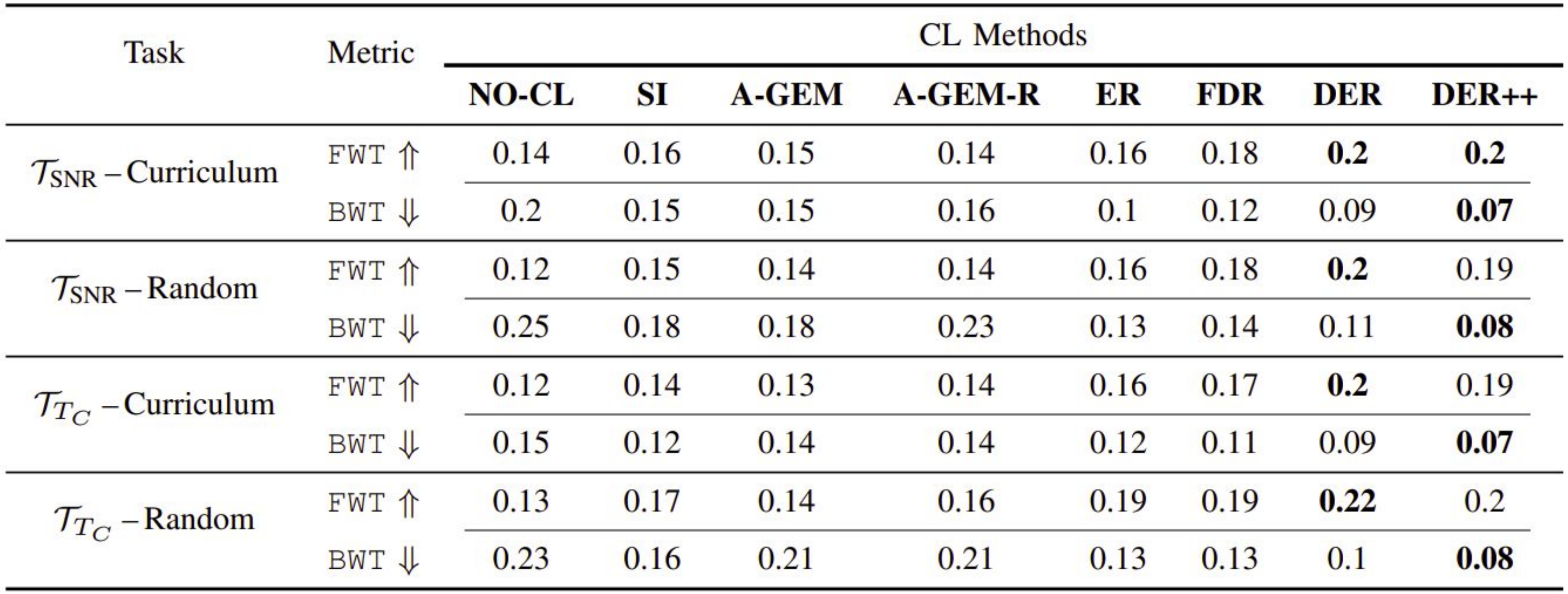


$\mathcal{T}_{T_c}$, random ordering

Continual learning for channel estimation

Task transferability

- Forward and backward transfers for each one of each CL method



Task	Metric	CL Methods							
		NO-CL	**SI**	**A-GEM**	**A-GEM-R**	**ER**	**FDR**	**DER**	**DER++**
$\mathcal{T}_{\text{SNR}}$ – Curriculum	FWT $\Uparrow$	0.14	0.16	0.15	0.14	0.16	0.18	**0.2**	**0.2**
	BWT $\Downarrow$	0.2	0.15	0.15	0.16	0.1	0.12	0.09	**0.07**
$\mathcal{T}_{\text{SNR}}$ – Random	FWT $\Uparrow$	0.12	0.15	0.14	0.14	0.16	0.18	**0.2**	0.19
	BWT $\Downarrow$	0.25	0.18	0.18	0.23	0.13	0.14	0.11	**0.08**
$\mathcal{T}_{T_C}$ – Curriculum	FWT $\Uparrow$	0.12	0.14	0.13	0.14	0.16	0.17	**0.2**	0.19
	BWT $\Downarrow$	0.15	0.12	0.14	0.14	0.12	0.11	0.09	**0.07**
$\mathcal{T}_{T_C}$ – Random	FWT $\Uparrow$	0.13	0.17	0.14	0.16	0.19	0.19	**0.22**	0.2
	BWT $\Downarrow$	0.23	0.16	0.21	0.21	0.13	0.13	0.1	**0.08**

Conclusion

- Distribution shifts is observed in the received power as a function of the SNR level and the coherence time.

- We benchmarked the state-of-the-art continual learning methods to account for the distribution shift in the Gauss-Markov channel estimation problem.

- We examined the learning of these two tasks in both curriculum and random orders. We confirmed the task ordering hypothesis, i.e., curriculum order is better than the random one.

- Simulation results suggest that CL methods with with experience replay outperform the other CL approaches.

Thank you for your attention!

University of Manitoba

IEEE ICC®