Unlearn to Learn Better: Continual Unlearning via Capacity Recycling

ECML PKDD 2026 Research Track

Pengxiang Wang[1] Shaobo Zhang[1] Hongbo Bo[2] Kedian Mu[1]

[1] Peking University, School of Mathematical Sciences, Beijing, China

[2] University of Bristol, Bristol, UK

Continual Learning

Continual learning trains a model on a sequence of tasks as new data and objectives arrive over time.

The model must learn each new task without using the full training data from previous tasks.

Catastrophic forgetting: learning a new task overwrites parameters that encode knowledge of previous tasks.

Continual learning algorithms seek a trade-off to get higher performance averaged on all tasks:

- **Stability:** preserve knowledge for previous tasks
- **Plasticity:** reserve representational space for new tasks

Machine Unlearning

Machine unlearning, under the area of trustworthy and responsible AI:

- Removes influence of specified subset of training data from trained neural networks ("the right to be forgotten")
- Such that the resulting model behaves as if it had never been trained on those data

Applications: data privacy, security, data quality, fairness, etc.

Naive retraining from scratch without the unlearning data is:

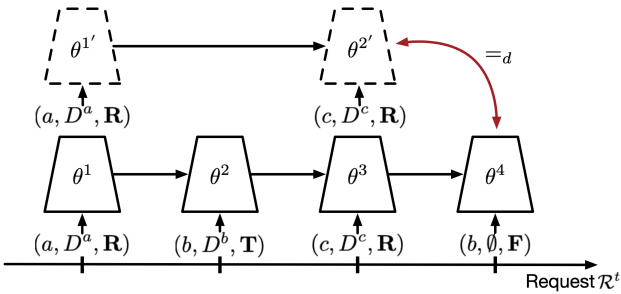
1. Computationally too expensive
2. The original training data might not be available or accessible

Unlearning is difficult.

Continual Unlearning

Continual unlearning: Task-level unlearning from continual learning model

(An underexplored field combines continual learning + machine unlearning)



Objectives:

1. **Unlearning**: Minimal difference from retraining without unlearning tasks
2. **Learning**: Maximal performance on remaining tasks

Uncontrolled task removal causes **catastrophic unlearning**: damage to knowledge that should remain.

Problem Definition

We learn a sequence of tasks $T_1, T_2, ...$, where each task T_t has a distinct dataset D_t.

At the end of task T_t, an unlearning request may specify tasks $u(t) \subseteq \{T_1, \dots, T_t\}$.

$$U(t) = \bigcup_{i=1}^{t} u(i), \qquad R(t) = \{T_1, \dots, T_t\} \setminus U(t)$$

The resulting model should:

- Remove the knowledge of all accumulated unlearned tasks $U(t)$
- Match the retrained reference on the remaining tasks $R(t)$
- Maintain strong performance over the remaining tasks $R(t)$

In practice, each task may have an **unlearnable age**, so only tasks in the permitted set $P(t)$ require preparation for future unlearning.

Existing Approaches

Continual learning

- Gained significant attention in deep learning research since around 2017
- A wide range of algorithms in several categories: replay-, regularization-, architecture-based, etc.

Machine unlearning

- Has recently emerged as an important research area in trustworthy AI since 2020, e.g., SISA, AmnesiacML, etc.
- Has led to research on continual unlearning

Continual unlearning

- A new and underexplored research area
- Related paradigms: LSF, CLPU, etc.

Amnesiac Continual Learning

We propose **Amnesiac Continual Learning (AmnesiacCL)**, a general continual unlearning framework

- Can be integrated into any continual learning approach for unlearning
- Core mechanism: update deletion (simple but effective)

Continual Learning

Store task-wise update record

$$\theta_{l,ij}^{(t)} = \theta_{l,ij}^{(0)} + \sum_{\tau=1}^{t} \Delta\theta_{l,ij}^{(\tau)}$$

Continual Unlearning

Update deletion

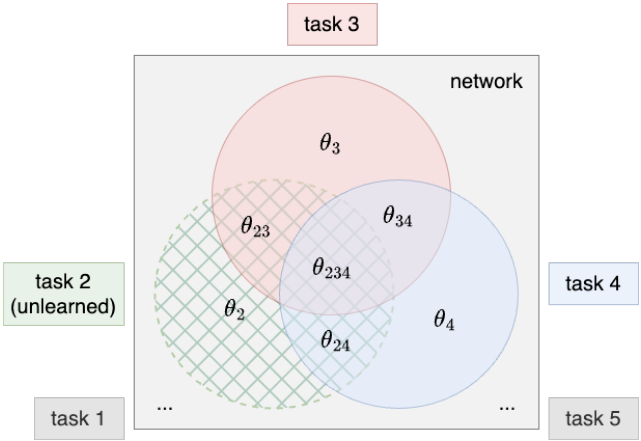
$$\theta_{l,ij}^{(t-u(t))} = \theta_{l,ij}^{(t)} - \sum_{\tau \in u(t)} \Delta\theta_{l,ij}^{(\tau)}$$

Amnesiac Continual Learning

Reversing updates by substraction does not always perfectly invert the training process, and can lead to catastrophic unlearning.

This is due to the stochastic and incremental nature of neural network training.

Motivation for AmnesiacHAT



Architecture-based approaches are naturally compatible for continual unlearning and make it easier:

- Allocate task-specific subnetworks
- Less overlapping means less task interference
- Removing a task explicitly releases its allocated subspace
- Only overlapping parameters are affected by unlearning

AdaHAT $\Rightarrow$ **AmnesiacHAT**

AmnesiacHAT Algorithm

Continual Learning

- **AdaHAT** (Adaptive Hard Attention to the Task) architecture and training
- **DER++** (Dark Experience Replay++) replay regularization

Continual Unlearning

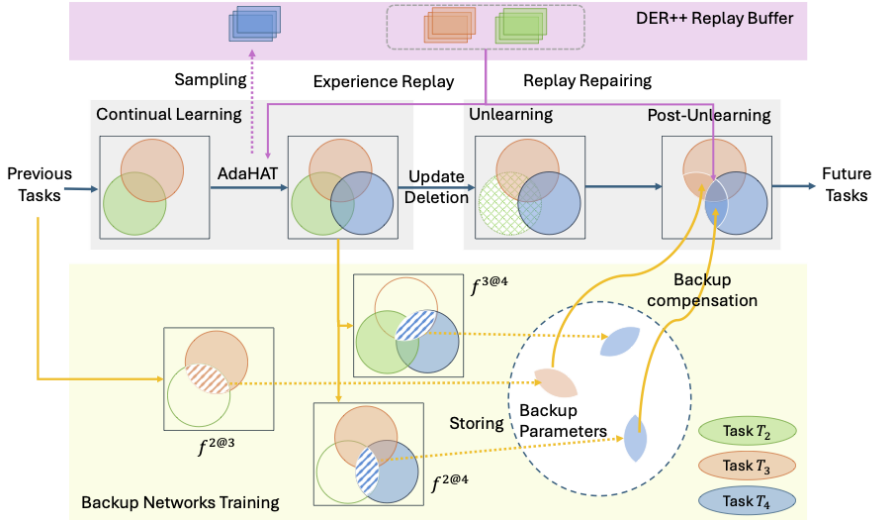
- **AmnesiacCL**: task-wise parameter update storage and deletion

Post-Unlearning (to further mitigate catastrophic unlearning)

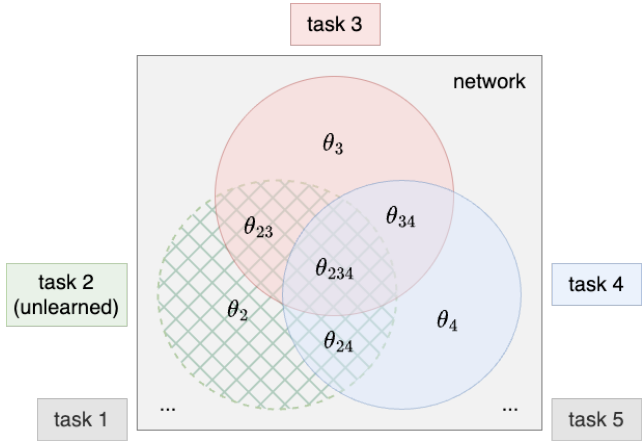
For only overlapping parameters:

- **Backup compensation**: immediate parameter replacement from backup networks
- **Replay repairing**: small-scale repair training to further restore performance

AmnesiacHAT Algorithm

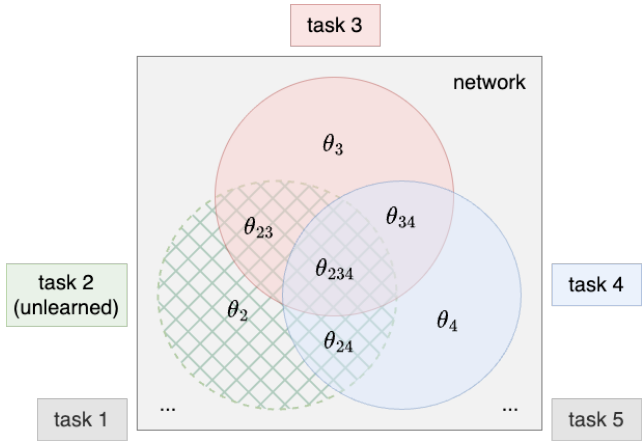


Unlearning Algorithm: Update Deletion



To unlearn task 2, delete $\Delta\theta^{(2)}$

Post-Unlearning: Backup Compensation



Backup when training task 3, 4: $f^{2@3}, f^{2@4}$

Compensation after unlearning task 2: $\theta_{23}, \theta_{24}, \theta_{234}$

Post-Unlearning: Backup Compensation

Continual Learning

When learning task T_t, train parallel **backup networks** $f^{p@t}$ for all permitted unlearnable tasks $T_p \in P(t)$, assuming:

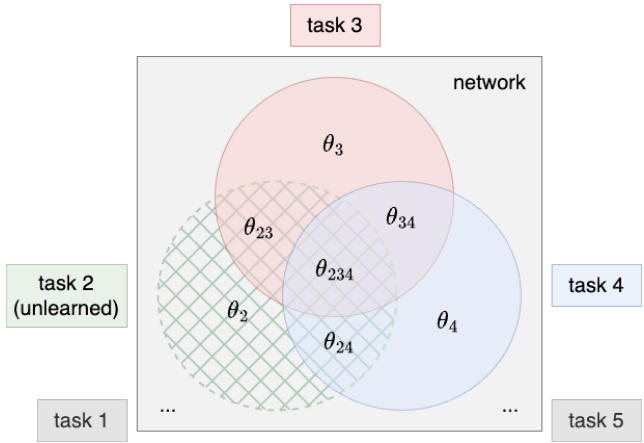
1. Non-overlapping parameters shared with the main network
2. Overlapping parameters freely updated from the state without T_p

$$\mathbf{h}_l^{p@t} = f_l^{p@t}(\mathbf{h}_{l-1}^{p@t})\mathbf{m}_l^t + f_l^t(\mathbf{h}_{l-1}^{p@t})(1 - \mathbf{m}_l^t)$$

Post-Unlearning

After unlearning T_p, replace parameters overlapping with all affected tasks T_t from backup networks after update deletion.

Post-Unlearning: Replay Repairing



Repair after unlearning task 2: $\theta_{23}, \theta_{24}, \theta_{234}$

Using replay data from task 3, 4

Post-Unlearning: Replay Repairing

A DER++ **replay memory** buffer is maintained during training to store a small amount of previous data, benefiting both continual learning and unlearning.

Continual Learning

Replay regularization

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + R_{\text{DER++}}(f(\mathbf{x}^{\text{replay}}), \mathbf{z}^{\text{replay}})$$

$$R_{\text{DER++}} = \lambda_{ce}\mathcal{L}_{cls} + \lambda_{dist} \left\| f(\mathbf{x}) - \mathbf{z} \right\|_2^2$$

Label replay and distillation from previous data

Post-Unlearning

Replay repairing

$$\mathcal{L}_{\text{repair}} = R_{\text{DER++}}(f(\mathbf{x}^{\text{replay}}), \mathbf{z}^{\text{replay}})$$

Repair for s steps, where replay data are from affected tasks by unlearning only

Discussion on Cost

AmnesiacHAT introduces mechanisms that increase computational and memory costs; however, none of them approaches the cost of full task-level computation.

- Updates are effectively restricted to subnetwork parameters; updates for update deletion can be stored sparsely

$$\Delta_s \theta^{T_p} = \Delta \theta^{T_p} \cdot m_\theta^{T_p}, T_p \in P(t)$$

- We choose the underlying architecture-based framework AdaHAT, which reduces task interference and thus lowers the cost of the two post-unlearning processes. Only overlapping parameters are affected by unlearning and require backup and repair.
- Only need to prepare for permitted unlearnable tasks $P(t)$. Once a task exceeds its unlearnable age, its updates, backups, and replay data can be released.

Evaluation Metrics

Continual unlearning is a new research area. We propose three primary metrics:

Average Distribution Distance (ADD): measures unlearning effectiveness by representation distance from the retrained reference model on unlearned tasks

$$\mathrm{ADD}_N = \frac{1}{|U(N)|} \sum_{T_\tau \in U(N)} \frac{1}{|D^\tau|} \sum_{(\mathbf{x},y)\in D^\tau} d\big(f(\mathbf{x}), f^{\text{retrain}}(\mathbf{x})\big)$$

Unlearning Accuracy Loss (UAL): measures catastrophic unlearning by accuracy loss on remaining tasks relative to retrained reference model

$$\mathrm{UAL}_N = \frac{1}{|R(N)|} \sum_{T_r \in R(N)} (a^{\text{retrain}}_{T_r,N} - a_{T_r,N})$$

Average Accuracy on Remaining Tasks (AAR): measures continual learning performance on remaining tasks after unlearning

$$\mathrm{AAR}_N = \frac{1}{|R(N)|} \sum_{T_r \in R(N)} a_{T_r,N}$$

Main Results

Dataset	Approach	ADD (↓)	UAL (↓)	AAR (↑)
Permuted MNIST, 5 tasks, TIL	Finetuning	54.71 ± 3.44	14.91 ± 7.70	71.67 ± 7.24
	LwF	50.06 ± 2.51	27.94 ± 6.26	59.96 ± 5.14
	EWC	56.77 ± 5.25	20.87 ± 6.00	65.19 ± 7.13
	DER	61.62 ± 5.62	28.94 ± 2.68	67.14 ± 2.46
	DER++	62.28 ± 3.14	18.67 ± 4.28	77.65 ± 3.93
	AmnesiacHAT	**41.30 ± 12.74**	**5.46 ± 0.50**	**89.54 ± 0.66**
	CLPU-DER++	59.70 ± 2.52	2.19 ± 0.31	94.13 ± 0.32
	Independent	16.88 ± 1.45	0.05 ± 0.20	96.73 ± 0.13
Permuted MNIST, 5 tasks, DIL	Finetuning	43.19 ± 2.22	15.08 ± 4.85	77.63 ± 3.83
	LwF	45.39 ± 3.15	43.53 ± 10.49	47.35 ± 10.46
	EWC	**42.12 ± 2.26**	17.10 ± 4.88	75.62 ± 3.73
	DER	58.42 ± 1.88	18.50 ± 3.36	77.57 ± 3.06
	DER++	54.22 ± 2.21	11.97 ± 2.29	84.30 ± 2.09
	AmnesiacHAT	45.87 ± 7.35	**5.32 ± 0.70**	**89.79 ± 0.68**
	CLPU-DER++	57.51 ± 2.06	1.83 ± 0.41	94.44 ± 0.28

Main Results

Rotated MNIST, 5 tasks, TIL	Finetuning	28.43 ± 1.35	36.48 ± 4.05	46.83 ± 2.53
	LwF	30.49 ± 3.79	29.51 ± 5.62	63.97 ± 4.83
	EWC	28.89 ± 3.82	34.37 ± 5.80	49.07 ± 5.54
	DER	35.23 ± 2.79	44.53 ± 5.29	51.42 ± 4.79
	DER++	34.00 ± 2.53	39.52 ± 4.00	56.39 ± 3.69
	AmnesiacHAT	$\mathbf{24.98 \pm 2.50}$	$\mathbf{6.90 \pm 0.77}$	$\mathbf{88.05 \pm 0.69}$
	CLPU-DER++	60.04 ± 2.17	-0.81 ± 0.30	96.72 ± 0.07
	Independent	13.19 ± 1.34	0.08 ± 0.22	96.59 ± 0.16
Rotated MNIST, 5 tasks, DIL	Finetuning	26.79 ± 2.38	31.94 ± 6.42	51.29 ± 5.71
	LwF	48.93 ± 5.16	34.84 ± 7.02	50.67 ± 6.36
	EWC	26.94 ± 3.50	32.64 ± 6.39	50.64 ± 5.88
	DER	40.84 ± 0.71	40.03 ± 3.57	54.52 ± 2.90
	DER++	38.50 ± 1.23	37.95 ± 3.58	56.58 ± 2.96
	AmnesiacHAT	$\mathbf{24.06 \pm 3.33}$	$\mathbf{5.90 \pm 1.42}$	$\mathbf{89.10 \pm 1.29}$
	CLPU-DER++	61.67 ± 2.41	-2.15 ± 0.52	96.68 ± 0.16

Main Results

Dataset	Approach	ADD (↓)	UAL (↓)	AAR (↑)
Split CIFAR-10, 5 tasks, TIL	Finetuning	89.83 ± 5.41	17.46 ± 5.15	55.52 ± 4.49
	LwF	89.46 ± 3.90	26.14 ± 6.65	60.76 ± 6.52
	EWC	93.84 ± 3.58	16.41 ± 4.07	54.00 ± 5.14
	DER	95.02 ± 1.63	37.53 ± 2.63	51.48 ± 2.29
	DER++	92.52 ± 2.56	38.57 ± 3.71	50.74 ± 3.57
	AmnesiacHAT	$\mathbf{86.80 \pm 9.42}$	$\mathbf{7.62 \pm 5.74}$	$\mathbf{76.16 \pm 1.54}$
	CLPU-DER++	91.61 ± 0.98	-3.21 ± 1.12	92.22 ± 0.22
	Independent	48.09 ± 6.90	0.19 ± 1.30	92.09 ± 0.51
Combined-5, 5 tasks, TIL	Finetuning	51.82 ± 3.87	45.96 ± 2.26	15.57 ± 2.66
	LwF	45.56 ± 6.76	61.96 ± 3.78	15.30 ± 2.15
	EWC	49.12 ± 3.71	47.52 ± 2.95	13.47 ± 3.71
	DER	46.90 ± 2.14	68.03 ± 3.00	16.93 ± 3.03
	DER++	44.81 ± 3.48	69.18 ± 3.26	15.59 ± 3.13
	AmnesiacHAT	$\mathbf{38.62 \pm 3.65}$	$\mathbf{17.93 \pm 2.20}$	$\mathbf{65.36 \pm 2.32}$
	CLPU-DER++	23.13 ± 2.28	9.46 ± 4.51	70.04 ± 4.66
	Independent	15.80 ± 0.53	0.03 ± 0.26	86.75 ± 0.24
Permuted MNIST, 20 tasks, TIL	Finetuning	67.15 ± 0.97	$\mathbf{9.97 \pm 3.12}$	56.73 ± 2.50
	LwF	$\mathbf{51.34 \pm 4.47}$	16.39 ± 4.84	49.43 ± 1.66
	EWC	64.43 ± 1.66	17.22 ± 4.85	54.26 ± 3.30
	DER	64.23 ± 1.86	27.14 ± 1.54	64.75 ± 1.97
	DER++	64.40 ± 2.32	13.67 ± 1.68	78.25 ± 2.08
	AmnesiacHAT	62.24 ± 4.12	12.61 ± 0.75	$\mathbf{80.54 \pm 0.80}$
	CLPU-DER++	72.73 ± 3.05	28.07 ± 1.67	63.85 ± 1.28
	Independent	6.85 ± 0.60	0.01 ± 0.06	96.68 ± 0.06

Main Results

Settings:

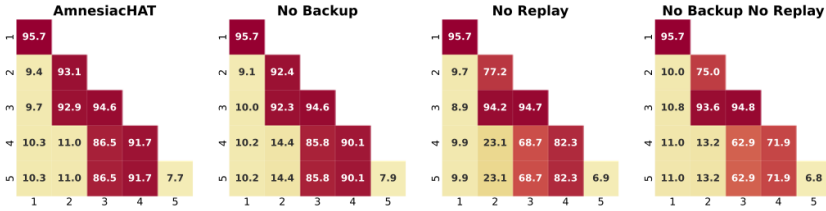
- Datasets: Permuted MNIST, Rotated MNIST, Split CIFAR-10, Combined-5
- Scenarios: Task-Incremental Learning (TIL), Domain-Incremental Learning (DIL)
- Task sequences:
  - 5 tasks (unlearn T_1 after T_2, T_2 after T_4, and T_5 after T_5)
  - 20 tasks (Permuted MNIST only)

Results:

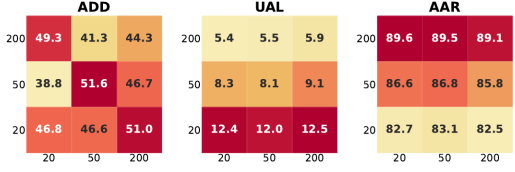
- AmnesiacCL-based approaches show strong ADD
- AmnesiacHAT shows lowest UAL and highest AAR
- AmnesiacHAT approaches upper-bound but costly baselines (Independent, CLPU-DER++)

Conclusions: unlearning in AmnesiacCL is effective; AmnesiacHAT further mitigates catastrophic unlearning and maintains strong continual learning performance.

Ablation and Hyperparameter Study



Backup compensation and replay repairing are both effective in mitigating catastrophic unlearning



Increasing replay buffer size M better mitigates catastrophic unlearning

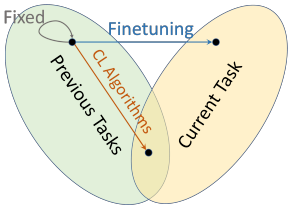
Unlearn to Learn Better

Our work is not only a unlearning solution for trustworthy AI requirement

But also a new perspective on benefiting continual learning from unlearning

Deliberate forgetting can make continual learning better.

Stability-Plasticity Trade-Off in Continual Learning



Continual learning algorithms seek a trade-off to get higher performance averaged on all tasks:

- **Stability:** preserve knowledge for previous tasks
- **Plasticity:** reserve representational space for new tasks

For replay-, regularization-based approaches:

- Neural network is maximally plastic by nature
- Inject stability in their forgetting prevention mechanisms
- But generally lean towards plasticity
- **Catastrophic forgetting:** performance degradation of previous tasks

For architecture-based approaches:

- Distinctly different strategies: allocate task-specific subnetworks
- Overemphasize stability
- **Network capacity problem:** capacity is quickly exhausted as task masks accumulate, performance degradation of future tasks

Unlearning Benefit: Capacity Recycling

Update deletion generally benefits continual learning by recycling network capacity in a passive way:

Most CL approaches lack stability

↓

Later tasks overwrite earlier tasks

↓

Deleting later updates shortens the overwrite tail

↓

Restore stability for earlier tasks

Architecture-based approaches lack plasticity

↓

Task masks occupy a fixed parameter space

↓

Deleting tasks releases their allocated subspaces

↓

Restore plasticity for learning future tasks

AmnesiacCL compensates for whichever side of the stability-plasticity trade-off is lacking through capacity recycling, thus making continual learning better

Metrics for Capacity Recycling Benefits

- Metrics for stability: Backward Transfer (BWT)
- Metrics for plasticity: Forward Transfer (FWT)

The no-unlearning reference is the continual learning model trained on all tasks without unlearning.

$$\mathrm{SG} = \mathrm{BWT}_{R(N)} - \mathrm{BWT}_{R(N)}^{\mathrm{nounlearn}}$$

$$\mathrm{PG} = \mathrm{FWT}_{R(N)} - \mathrm{FWT}_{R(N)}^{\mathrm{nounlearn}}$$

Positive **Stability Gain (SG)** or **Plasticity Gain (PG)** shows which missing side is restored by unlearning.

Results on Capacity Recycling Benefits

Approach	$\text{BWT}^{\text{nounlearn}}_{R(N)}$	$\text{FWT}^{\text{nounlearn}}_{R(N)}$	SG	PG
Finetuning	-48.86 ± 2.81	-0.13 ± 0.08	$\mathbf{15.22 \pm 2.02}$	-6.17 ± 0.77
LwF	-45.18 ± 1.50	0.20 ± 0.10	$\mathbf{7.56 \pm 1.55}$	-9.83 ± 1.47
EWC	-46.50 ± 2.57	-0.08 ± 0.14	$\mathbf{10.02 \pm 4.54}$	-5.86 ± 1.48
DER	-29.53 ± 1.76	0.02 ± 0.10	$\mathbf{3.78 \pm 2.96}$	-6.19 ± 0.82
DER++	-31.71 ± 2.45	-0.00 ± 0.04	$\mathbf{18.26 \pm 2.32}$	-4.98 ± 1.78
AmnesiacHAT (original)	-16.28 ± 1.26	-31.85 ± 1.20	-1.31 ± 2.35	$\mathbf{11.13 \pm 3.20}$
AmnesiacHAT	-5.47 ± 0.70	-1.64 ± 0.09	-12.11 ± 1.94	-2.57 ± 0.64
CLPU-DER++	-31.71 ± 2.45	-0.00 ± 0.04	-1.37 ± 3.06	0.26 ± 0.09
Independent	0.00 ± 0.00	0.09 ± 0.10	0.00 ± 0.00	-0.09 ± 0.10

Conclusion

We studied continual unlearning as the joint problem of sequential learning and selective task removal.

We propose **AmnesiacCL**:

- A general continual unlearning framework that can be integrated into any continual learning approach to enable unlearning
- A passive mechanism that recycles network capacity, which balances stability-plasticity trade-off and makes continual learning better

We develop **AmnesiacHAT** based on AmnesiacCL:

- Makes unlearning easier and less catastrophic by reducing task interference
- Mitigates catastrophic unlearning through two post-unlearning processes

Insights: Unlearning is not only a trustworthy AI requirement; it can also passively make continual learning better.

Thank You

Thank you for your attention!

Please feel free to ask any questions or reach out to us at:

wangpengxiang@stu.pku.edu.cn

Our work is implemented as a Python package for continual learning and unlearning research: pengxiang-wang.com/projects/continual-learning-arena

(feel free to give it a star! :D)



北京大学数学科学学院
School of Mathematical Sciences,Peking University

University of
BRISTOL