Fine-Grained Neuron Importance Guided Architecture-Based Continual Learning

Pengxiang Wang[1] Rongyu Zhu[1] Junjie Liu[1] Kedian Mu[1]
[1]School of Mathematical Sciences, Peking University, Beijing, China
ECML PKDD 2026 Research Track
wangpengxiang@stu.pku.edu.cn

Continual Learning and the Stability–Plasticity Trade-Off

Continual learning (CL) is a machine learning paradigm that learns sequential tasks and adapts to new information over time. One of its major problems is **catastrophic forgetting**: drastic performance degradation on previous tasks after learning new tasks.

In Task-Incremental Learning (TIL), a model is trained on a sequence of tasks $t = 1, \ldots, T$, where each task is associated with a distinct data distribution $D^t = \{x^t, y^t\}$. The model has no access to the data of previous tasks and should perform well on all learned tasks.

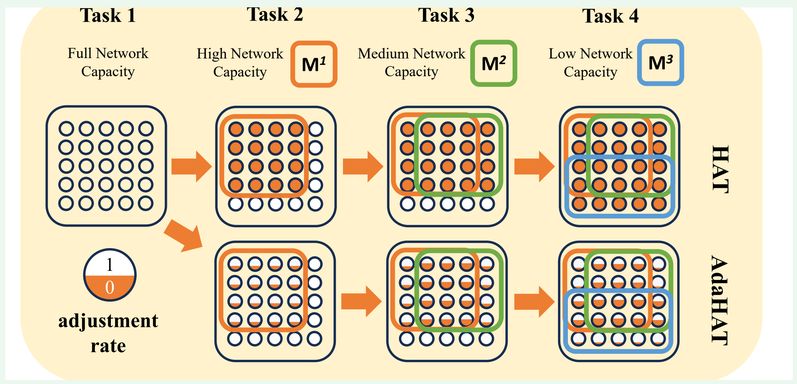
Continual learning is a trade-off between:

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

Continual learning algorithms must trade them off to get higher performance averaged on all tasks.

Architecture-Based Continual Learning: HAT and AdaHAT

Architecture-based approaches assign each task a network subspace. Typical strategies include parameter allocation (PackNet, WSN), neuron allocation (HAT, NISPA), modular networks (Progressive Networks, PathNet), and model decomposition (APD, PGMA).



- **HAT hard-clips gradients** through binary masks. Active parameters become strictly frozen, leading to insufficient network capacity, reduced learning plasticity, and drastic performance degradation on new tasks.
- **AdaHAT soft-clips gradients** through task information. It allows small updates to parameters allocated to previous tasks, adaptively reserves capacity for future tasks, and rebalances the stability–plasticity trade-off.

Fine-Grained Task Information and Contributions

Fine-grained task information plays a crucial role among other continual learning approaches, meaningfully contributing to guiding the training of new tasks. EWC uses parameter-wise Fisher information; MAS and OGD leverage parameter gradients; TAG uses gradient correlation; Continual Backpropagation uses neuron-wise contribution utility.

Architecture-based approaches rarely exploit fine-grained information. AdaHAT uses task information to guide training, but it is less fine-grained and lacks granularity: parameter importance is an integer value from 0 to $t-1$, while network sparsity is a single scalar value for the whole network.

Our contributions:

- We develop **FG-AdaHAT**, a general gradient adjustment framework to which any neuron-wise importance measure and its scheduler can be applied.
- We propose novel architecture-based approaches, each guided by a representative type of neuron importance.
- Our experiments show that incorporating fine-grained information benefits continual learning from the architecture-based perspective.

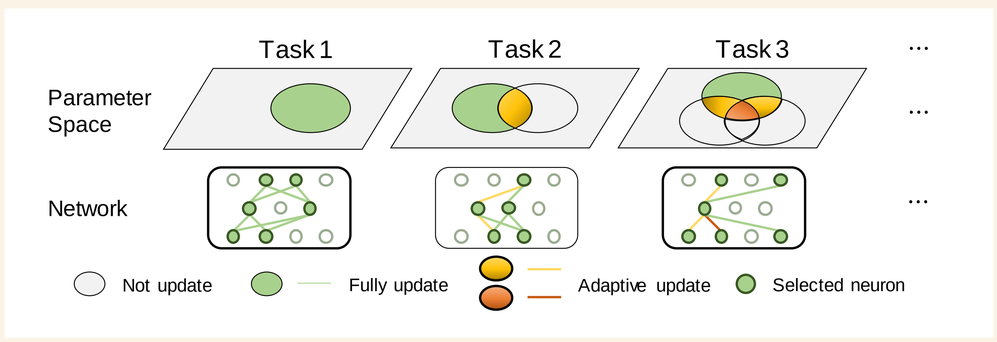
FG-AdaHAT: Fine-Grained Adaptive Gradient Adjustment

The room for gradient adjustment is limited: even small but improperly managed updates can lead to catastrophic forgetting. The adjustment rate must therefore be refined and carefully designed.

$$a_{l,ij}^{\text{FG-AdaHAT}} = \alpha \left[c^t \cdot \mathrm{Agg}\left(I_{l,i}^{<t}, I_{l-1,j}^{<t}\right) + \alpha \right]^{-1}.$$

- $I_{l,i}^{<t}, I_{l-1,j}^{<t} \geq 0$: neuron importance scores for previous tasks;
- c^t: a task-specific factor modulating the influence of importance scores;
- $\alpha > 0$: the overall intensity of gradient adjustment;
- $\mathrm{Agg}(\cdot)$ aggregates two neuron scores into the importance of their connecting parameter (e.g., min, max, mean).

The more important the parameter is to previous tasks, the less it should be updated. FG-AdaHAT is a general framework to which any importance measure can be applied.



1. Not selected by the current task: the original gradient is blocked; the parameter is not updated.
2. Selected by the current task but not by previous tasks: importance is zero or nearly zero; $a_{l,ij} = 1$ and the parameter is free to update.
3. Selected by both current and previous tasks: importance is positive; $0 < a_{l,ij} < 1$ allows controlled updates (typically between 10^{-7} and 10^{-5}).

Task-Specific Importance Scheduling

The **importance scheduler** scales parameter importance to further modulate the gradient adjustment rate:

$$c^t = (t + b_L)\left[R\left(\mathsf{M}^t, \mathsf{M}^{<t}\right) + b_R\right], \qquad b_L, b_R > 0.$$

- $(t + b_L)$ increases importance as more tasks arrive, making adjustment rates steadily smaller;
- $R(\mathsf{M}^t, \mathsf{M}^{<t})$ is the HAT sparsity regularization term, closely related to current network capacity usage;
- b_L and b_R ensure that adjustment rates start from small values.

When network capacity is sufficient, there is less need to update parameters allocated to previous tasks. This helps approximate HAT in early tasks.

Computing Fine-Grained Neuron Importance

Sample-wise local importance → task-wise global importance → importance to previous tasks

$$I_{l,i}^{<t} = \sum_{\tau<t} I_{l,i}^{\tau}, \qquad I_{l,i}^{\tau} = \frac{1}{|D^\tau|} \sum_{(\mathbf{x},y)\in D^\tau} I_{l,i}^{\tau}(\mathbf{x}, y).$$

The local importance measures must satisfy:

1. values lie in $[0, 1]$;
2. unselected neurons have zero importance; selected neurons have strictly positive importance.

Preprocessing uses layer-wise min–max scaling, followed by base offset and mask filtering:

$$I_l^\tau(\mathbf{x}, y) \leftarrow \frac{I_l^\tau(\mathbf{x}, y) - \min_i I_{l,i}^\tau(\mathbf{x}, y)}{\max_i I_{l,i}^\tau(\mathbf{x}, y) - \min_i I_{l,i}^\tau(\mathbf{x}, y)},$$

$$I_{l,i}^\tau(\mathbf{x}, y) \leftarrow m_{l,i}^\tau\left(I_{l,i}^\tau(\mathbf{x}, y) + b_I\right), \qquad b_I > 0.$$

Representative Fine-Grained Neuron Importance Measures

Neuron importance in FG-AdaHAT is constructed from diverse information sources, which is finer-grained than parameter importance and network sparsity in AdaHAT.

Training information: Neuron activations, model weights, parameter gradients, empirical Fisher information, etc.

Measures:

- Input Gradients (IG): $I_{l,i}^\tau(\mathbf{x}, y) = \sum_j |g'_{l,ij}|$
- Output Gradients (OG): $I_{l,j}^\tau(\mathbf{x}, y) = \sum_i |g'_{l+1,ij}|$
- Contribution Utility (CU): $I_{l,j}^\tau(\mathbf{x}, y) = |h_{l,j}(\mathbf{x})| \sum_i |w_{l+1,ij}|$
- Input Contribution Utility (ICU): $I_{l,i}^\tau(\mathbf{x}, y) = |h_{l,i}(\mathbf{x})| \sum_j |w_{l,ij}|$
- Activation Fisher Information (AFI): $I_{l,i}^\tau(\mathbf{x}, y) = |h_{l,i}(\mathbf{x})| \sum_j |g'_{l,ij}|^2$

Layer attribution methods from XAI:

- Feature Ablation (FA): perturbation-based attribution, measuring the change in output when a feature is removed;
- Internal Influence (II): gradient-based attribution, measuring a feature's influence through its gradient;
- Gradient SHAP (GS): gradient-based attribution using Shapley values;
- DeepLIFT (DL): backpropagation-based attribution.

Experimental Setup and Evaluation Metrics

Datasets

- Permuted MNIST
- Split CIFAR-100
- Combined-20
- 20 tasks

Metrics for performance

- Average Accuracy (AA) over all tasks
- Forgetting Rate (FR)

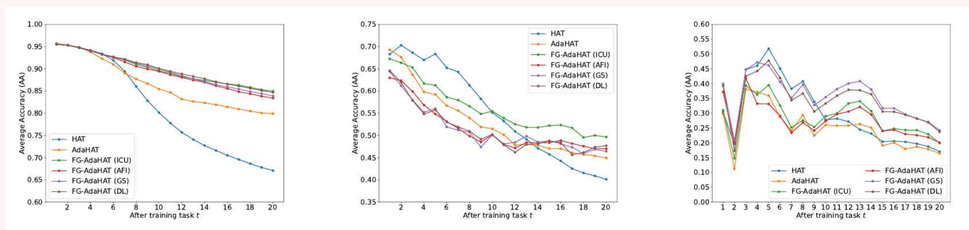
Metrics for stability–plasticity trade-off

- Backward Transfer (BWT)
- Forward Transfer (FWT)

Main Results

Approach	Permuted MNIST AA (↑)	Permuted MNIST FR (↑)	Split CIFAR-100 AA (↑)	Split CIFAR-100 FR (↑)	Combined-20 AA (↑)	Combined-20 FR (↑)
Finetuning	32.06 ± 0.96	−73.19 ± 1.11	34.19 ± 1.84	−72.47 ± 3.34	10.88 ± 0.44	−74.45 ± 1.72
LwF	32.95 ± 1.35	−72.12 ± 1.55	31.78 ± 2.36	−77.20 ± 5.34	10.92 ± 0.91	−74.36 ± 1.04
HAT	67.10 ± 0.56	−30.68 ± 0.65	40.12 ± 2.17	−59.12 ± 4.01	17.05 ± 2.57	−74.63 ± 4.16
AdaHAT	79.91 ± 1.47	−12.43 ± 6.46	44.93 ± 1.29	−50.28 ± 2.51	16.46 ± 2.49	−68.31 ± 4.74
FG-AdaHAT (IG)	84.05 ± 1.71	−10.12 ± 1.86	45.41 ± 3.30	−49.36 ± 5.87	**24.67 ± 4.15**	−56.65 ± 4.74
FG-AdaHAT (OG)	**85.05 ± 0.71**	**−8.92 ± 0.82**	45.54 ± 2.19	−49.55 ± 3.69	22.05 ± 2.77	−59.11 ± 4.17
FG-AdaHAT (CU)	84.11 ± 0.91	−10.05 ± 1.05	45.71 ± 3.47	−49.12 ± 6.42	22.26 ± 1.69	−59.98 ± 3.76
FG-AdaHAT (ICU)	84.97 ± 0.63	−9.01 ± 0.72	**49.66 ± 1.88**	**−40.93 ± 3.75**	19.95 ± 3.93	−62.37 ± 4.80
FG-AdaHAT (AFI)	83.43 ± 1.28	−10.88 ± 1.47	47.00 ± 1.79	−46.66 ± 3.31	20.09 ± 5.16	−61.29 ± 5.90
FG-AdaHAT (FA)	84.86 ± 0.89	−9.14 ± 1.03	—	—	—	—
FG-AdaHAT (II)	83.66 ± 0.74	−10.60 ± 0.85	45.70 ± 1.66	−48.85 ± 2.85	22.32 ± 1.01	−59.10 ± 4.24
FG-AdaHAT (GS)	83.91 ± 0.70	−10.30 ± 0.81	46.43 ± 3.83	−47.40 ± 7.32	24.25 ± 6.52	**−56.05 ± 9.23**
FG-AdaHAT (DL)	84.70 ± 0.57	−9.34 ± 0.66	47.70 ± 4.64	−45.02 ± 7.92	23.71 ± 4.49	−57.79 ± 5.34

Performance Evolution



(a) Permuted MNIST (b) Split CIFAR-100 (c) Combined-20

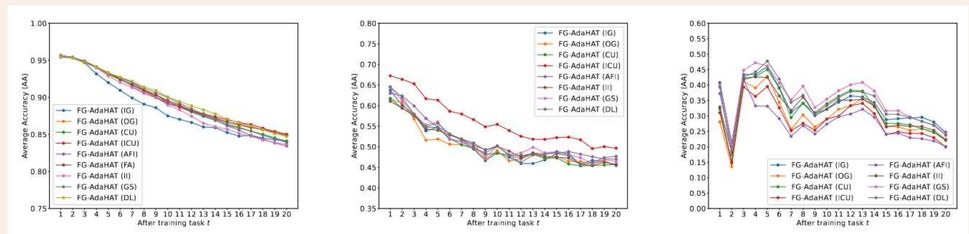
Results:

- FG-AdaHAT using 9 fine-grained importance measures all outperforms AdaHAT and other baselines
- More balanced stability–plasticity trade-off
- Outperforms AdaHAT in the early tasks

Conclusions:

- Fine-grained neuron importance guidance helps more effectively balance the stability–plasticity trade-off via more accurate gradient adjustment, thereby improving overall performance within the HAT architecture
- FG-AdaHAT's importance scheduler helps to further approximate HAT in early tasks

Comparison of Fine-Grained Importance Measures

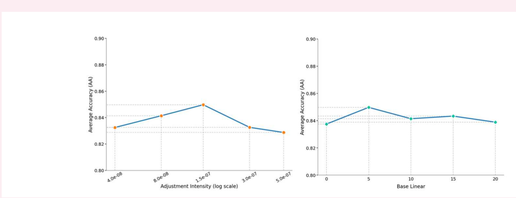


(a) Permuted MNIST (b) Split CIFAR-100 (c) Combined-20

Results: Performance evolution is largely similar and consistent across different fine-grained neuron importance measures, with only minor differences.
Conclusions: FG-AdaHAT framework is consistently effective and shows wide applicability.

Ablation and Hyperparameter Study

Approach	Components: linear t	Components: base b_L	Components: fine-grained importance	AA ↑
FG-AdaHAT	✓	✓	✓	**84.97**
Case 1	×	✓	✓	84.73
Case 2	✓	×	✓	83.75
AdaHAT	×	×	×	79.91



Conclusion

We proposed **FG-AdaHAT**, a general adaptive gradient adjustment framework for architecture-based continual learning.

- It leverages fine-grained task information as neuron-wise importance
- An importance scheduler guides gradient adjustment more accurately and effectively
- 9 representative measures are constructed from diverse information sources, all shown effective
- All measures outperform AdaHAT and other existing approaches on long task sequences by better balancing stability and plasticity

Limitation: based on heuristics and lack a theoretical foundation

Future work:

- Apply fine-grained measures to more architecture-based approaches
- Explore the theoretical foundations behind, e.g., information theory

Package Information

Our work is implemented as a Python package for continual learning research:
pengxiang-wang.com/projects/continual-learning-arena
(feel free to give it a star! :D)