Fine-Grained Neuron Importance Guided Architecture-Based Continual Learning

ECML PKDD 2026 Research Track

Pengxiang Wang[1] Rongyu Zhu[1] Junjie Liu[1] Kedian Mu[1]

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

Continual Learning and Catastrophic Forgetting

Continual Learning

- A machine learning paradigm
- Learn sequential tasks and adapt to new information over time
- One of the key features of human intelligence

Catastrophic Forgetting

- Drastic performance degradation on previous tasks after learning new tasks
- A major problem for continual learning algorithm to address

Problem Definition

Continual Learning (CL): learning a sequence of tasks $t = 1, \cdots, N$ in order, with datasets $D^t = \{x^t, y^t\}$

Task-Incremental Learning (TIL): continual learning scenario, aim to train a model f that performs well on all learned tasks

$$\max_f \sum_{t=1}^{N} \text{metric}(f(x^t), y^t), \{x^t, y^t\} \in D^t$$

Key assumptions when training and testing task t:

- No access to the entire data from previous tasks $1, \cdots, t-1$
- Testing on all seen tasks $1, \cdots, t$
- For TIL testing, task ID t of each test sample is known by the model

Existing Approaches for TIL

Replay-based Approaches

- Prevent forgetting by storing parts of the data from previous tasks
- Replay algorithms use them to consolidate previous knowledge
- E.g. GEM, DGR, DER, ...

Regularization-based Approaches

- Introduce regularization terms constructed using information about previous tasks to the loss function when training new tasks
- E.g. LwF, EWC, SI, IMM, VCL, ...

Architecture-based Approaches

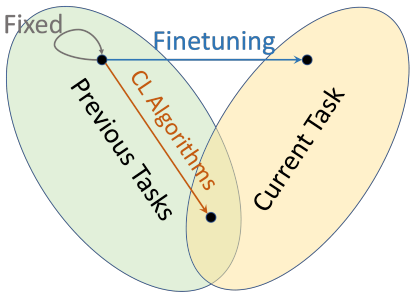
- Allocate network parameters in different parts of the network to different tasks
- Focus on reducing representational overlap in the network
- Keep the parameters learned in previous tasks from being significantly changed
- E.g. Progressive Networks, PathNet, PackNet, Piggyback, HAT, CPG, SupSup, ...

Stability-Plasticity Trade-Off

Continual learning is a trade-off between stability and plasticity.

- **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.



For replay-, regularization-based approaches:

- Inject stability in their forgetting prevention mechanisms
- But generally still lean towards plasticity

For architecture-based approaches:

- Distinctly different strategies that overemphasize stability
- Shifting the trade-off towards stability instead

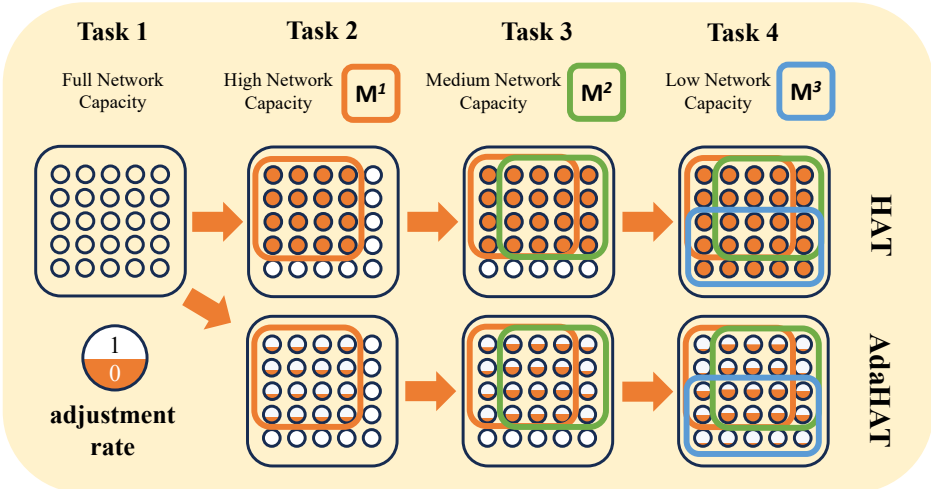
Architecture-Based Continual Learning

Architecture-based approaches assign each task an network subspace.

- Paramater allocation, e.g. PackNet, WSN
- Neuron allocation, e.g. HAT, NISPA
- Modular network, e.g. Progressive Networks, PathNet
- Model decomposition, e.g. APD, PGMA

Gradient adjustment approaches allow small updates to parameters allocated to previous tasks through adjusting gradients, while other approaches freeze them, e.g. AdaHAT

HAT and AdaHAT



HAT and AdaHAT

HAT **hard-clips gradients** through binary masks:

$$g'_{l,ij} = a^{\text{HAT}}_{l,ij} \cdot g_{l,ij},\ a^{\text{HAT}}_{l,ij} \in \{0, 1\}$$

AdaHAT **soft-clips gradients** through task information:

$$g'_{l,ij} = a^{\text{AdaHAT}}_{l,ij} \cdot g_{l,ij},\ a^{\text{AdaHAT}}_{l,ij} \in [0, 1]$$

As more tasks arrive...

Active parameters become strictly frozen

↓

Insufficient network capacity

↓

Reduced learning plasticity, inbalanced stability-plasticity trade-off

↓

Drastic performance degradation on new tasks

Allow small updates to parameters allocated to previous tasks

↓

Adaptively reserve capacity for future tasks

↓

Rebalanced stability-plasticity trade-off

↓

Improved performance after network capacity runs out

Fine-Grained Task Information and Applications

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 in regularization
- MAS and OGD leverage parameter gradients in importance scores and directions
- TAG uses gradient correlation to adapt learning rates
- Continual Backpropagation uses neuron-wise contribution utility to restore plasticity

Architecture-based approaches rarely exploit fine-grained information. AdaHAT is one of which uses task information to guide training, but it is less fine-grained and lack granularity:

- Parameter Importance is an integer value from 0 to $t-1$
- Network Sparsity is a single scalar value for the whole network

AdaHAT $\Rightarrow$ **FG-AdaHAT (Fine-Grained AdaHAT)**

FG-AdaHAT: HAT-Based Adaptive Gradient Adjustment Framework

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.

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

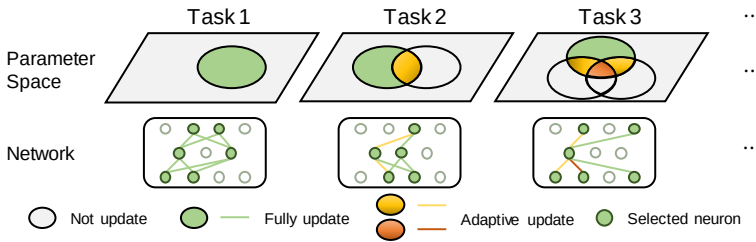
- $I_{l,i}^{<t}, I_{l-1,j}^{<t} \geq 0$: neuron importance scores for previous tasks

FG-AdaHAT is a general framework to which any importance measure can be applied

- c^t: a task-specific factor modulating the influence of importance scores
- $\alpha > 0$: the overall intensity of gradient adjustment
- $\text{Agg}(\cdot)$: aggregates two neuron scores into the importance of their connecting parameter, e.g., min, max, mean, etc

More important the parameter to previous tasks, less it should be updated

FG-AdaHAT: Situational Gradient Adjustment



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})

FG-AdaHAT: Task-Specific Importance Scheduling

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 $\rightarrow$ Task-wise global importance

$\rightarrow$ 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:

1. Layer-wise min-max scaling: $I_l^{\tau}(\mathbf{x}, y) \leftarrow \frac{I_l^{\tau}(\mathbf{x},y) - \min_i I_{l,i}^{\tau}(\mathbf{x},y)}{\max_i I_{l,i}^{\tau}(\mathbf{x},y) - \min_i I_{l,i}^{\tau}(\mathbf{x},y)}$
2. Base offset and mask filtering: $I_{l,i}^{\tau}(\mathbf{x}, y) \leftarrow m_{l,i}^{\tau} \left(I_{l,i}^{\tau}(\mathbf{x}, y) + b_I \right), b_I > 0$

Representative Fine-Grained Neuron Importance Measures

Neuron Importance used in FG-AdaHAT is constructed from various representative information sources, which is finer-grained than Parameter Importance and Network Sparsity in AdaHAT.

Training information

- Neuron activations, model weights, and parameter gradients
- Empirical Fisher information

Measures:

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

Representative Fine-Grained Neuron Importance Measures

Neuron Importance used in FG-AdaHAT are constructed from various representative information sources, which is finer-grained than Parameter Importance and Network Sparsity in AdaHAT.

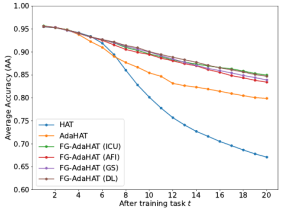
Layer attribution methods from Explainable AI (XAI) provides more fine-grained neuron importance measures from an alternative perspective:

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

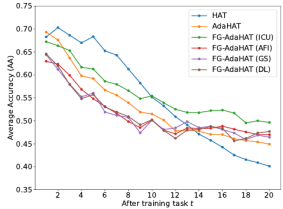
Main Results

Approach	Permuted MNIST		Split CIFAR-100		Combined-20	
	AA (↑)	FR (↑)	AA (↑)	FR (↑)	AA (↑)	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

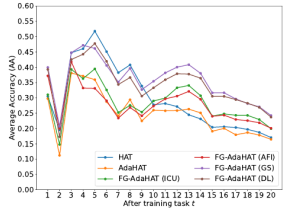
Main Results



(a) Permuted MNIST



(b) Split CIFAR-100



(c) Combined-20

Main Results

Approach	Permuted MNIST		Split CIFAR-100		Combined-20	
	BWT (↑)	FWT (↑)	BWT (↑)	FWT (↑)	BWT (↑)	FWT (↑)
Finetuning	−68.52 ± 0.96	0.42 ± 0.02	−51.17 ± 1.87	10.70 ± 0.40	−64.88 ± 0.57	9.87 ± 0.22
LwF	−67.42 ± 1.34	0.25 ± 0.05	−51.78 ± 3.40	8.88 ± 1.06	−64.75 ± 0.81	9.70 ± 0.21
HAT	−0.00 ± 0.00	−31.12 ± 0.57	−21.40 ± 2.40	−12.43 ± 0.72	−19.18 ± 3.98	−27.31 ± 4.40
AdaHAT	−12.46 ± 1.42	−5.19 ± 0.09	−27.23 ± 1.06	−1.60 ± 0.97	−40.91 ± 2.29	−6.16 ± 1.29
FG-AdaHAT (IG)	−7.73 ± 1.67	−5.55 ± 0.17	−19.90 ± 3.13	−8.13 ± 0.82	−33.90 ± 2.45	−5.11 ± 2.62
FG-AdaHAT (OG)	−5.33 ± 0.70	−6.90 ± 0.19	−19.66 ± 2.08	−8.12 ± 1.01	−34.55 ± 2.13	−6.56 ± 1.80
FG-AdaHAT (CU)	−6.09 ± 0.90	−7.14 ± 0.11	−22.61 ± 2.62	−5.00 ± 0.77	−35.31 ± 1.57	−5.79 ± 0.71
FG-AdaHAT (ICU)	−5.57 ± 0.67	−6.75 ± 0.11	−23.60 ± 1.70	−0.41 ± 0.69	−38.10 ± 2.72	−5.37 ± 1.55
FG-AdaHAT (AFI)	−7.32 ± 1.24	−6.62 ± 0.14	−21.92 ± 2.23	−4.40 ± 1.18	−37.06 ± 2.86	−6.58 ± 2.71
FG-AdaHAT (FA)	−6.71 ± 0.87	−5.73 ± 0.09	—	—	—	—
FG-AdaHAT (II)	−6.02 ± 0.70	−7.69 ± 0.12	−20.57 ± 1.54	−7.01 ± 1.58	−34.48 ± 4.14	−6.59 ± 3.23
FG-AdaHAT (GS)	−4.75 ± 0.69	−8.68 ± 0.25	−20.92 ± 2.61	−6.06 ± 1.83	−34.58 ± 4.19	−4.83 ± 1.80
FG-AdaHAT (DL)	−3.89 ± 0.57	−8.70 ± 0.25	−20.03 ± 3.92	−5.63 ± 1.11	−34.65 ± 2.26	−5.29 ± 1.70

Main Results

Settings:

- 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)

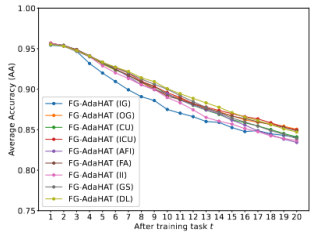
Results:

- 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

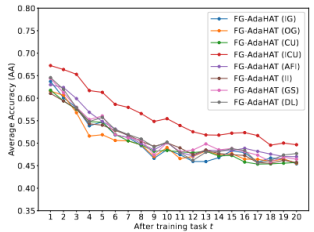
Conclusions:

- 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

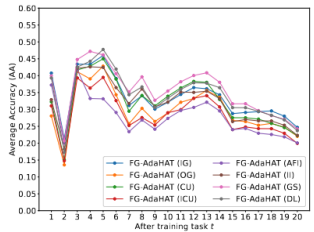
Comparison of Fine-Grained Importance Measures



(a) Permuted MNIST



(b) Split CIFAR-100



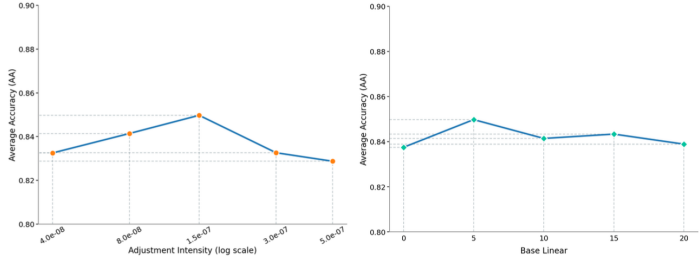
(c) Combined-20

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			AA (↑)
	linear t	base linear b_L	fine-grained importance	
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

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 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