Enhancing Efficiency and Performance in Deepfake Audio Detection through Neuron-level Dropin & Neuroplasticity Mechanisms

1st Yupei Li*
Department of Computing and Chair of Health Informatics
Imperial College London and Technical University of Munich
London and Munich, United Kingdom and Germany
yl7622@ic.ac.uk

2nd Shuaijie Shao*
Department of Computer Science
University College London
London, United Kingdom
shuaijie.shao.25@ucl.ac.uk

3rd Manuel Milling
Chair of Health Informatics
Technical University of Munich
Munich, Germany
manuel.milling@tum.de

4th Björn Schuller
Chair of Health Informatics and Department of Computing
Technical University of Munich and Imperial College London
Munich and London, Germany and United Kingdom
schuller@tum.de

* Yupei Li and Shuaijie Shao contributed equally to this work.

***Abstract*—Current audio deepfake detection has achieved remarkable performance using diverse deep learning architectures such as ResNet, and has seen further improvements with the introduction of large models (LMs) like Wav2Vec. The success of large language models (LLMs) further demonstrates the benefits of scaling model parameters, but also highlights one bottleneck where performance gains are constrained by parameter counts. Simply stacking additional layers, as done in current LLMs, is computationally expensive and requires full retraining. Furthermore, existing low-rank adaptation methods are primarily applied to attention-based architectures, which limits their scope. Inspired by the neuronal plasticity observed in mammalian brains, we propose novel algorithms, dropin and further plasticity, that dynamically adjust the number of neurons in certain layers to flexibly modulate model parameters. We evaluate these algorithms on multiple architectures, including ResNet, Gated Recurrent Neural Networks, and Wav2Vec. Experimental results using the widely recognised ASVSpoof2019 LA, PA, and FakeorReal dataset demonstrate consistent improvements in computational efficiency with the dropin approach and a maximum of around 39% and 66% relative reduction in Equal Error Rate with the dropin and plasticity approach among these dataset, respectively. The code and supplementary material are available at Github link.**

***Index Terms*—Neuroplasticity, Audio Deepfake Detection, Wav2Vec, Efficiency, Finetuning**

I. Introduction

Large models (LMs) have recently surpassed the capability bottlenecks of traditional deep learning models across a wide range of tasks [1], including audio-related applications such as deepfake audio detection. For instance, audio LLMs, such as QwenAudio [2], have demonstrated superior performance compared to conventional small-scale deep learning models. This performance improvement aligns with established scaling laws indicating increasing the number of model parameters generally enhances representational capacity and downstream task performance [3] given sufficient training data and computation.

Additionally, we argue that large-scale models are beneficial for capturing the complexity of audio deepfake detection. Although this task is framed as a binary classification problem, the implicit discrepancies between spoofed and bona fide audio require the network to learn subtle and high-dimensional acoustic representations [4]. Existing approaches employ diverse architectural paradigms for feature extraction: convolutional neural network (CNN)-based models such as ResNet [5] process audio data as Mel-spectrograms by treating them as image-like features; recurrent neural network (RNN)-based architectures, including the Light Convolutional Gated Recurrent Neural Network (LC-GRNN), analyse spectrograms as temporally structured sequential data [6]; and attention-based pretrained models, notably Wav2Vec 2.0 [7], directly operate on raw audio waveforms to capture fine-grained temporal dependencies. Additional models have been proposed for this task, including ASSIST [8] and RawNet2 [9], etc. However, the approaches aforementioned have not fully resolved the challenges of audio deepfake detection, as evidenced by their performance in the recent ASVspoof 2019 challenge [10]. Nevertheless, it is a consistent trend that models with more parameters tend to achieve superior performance.

These models commonly rely on fixed architectures, often incorporating fine-tuning model parameters pretrained on a different task. However, inspired by the mechanisms of the mammalian brain, it may be argued that modern models should aim for dynamic neuron-level adjustment called (neuro-)plasticity, enabling them to remain memory-efficient while unlocking additional capacity for target tasks. Li et al. [11]

reviewed the principles of plasticity in biology and deep learning methods that adopt similar dropin (adding more neurons) and plasticity concepts, yet without specific implementation details and any empirical tests. This idea has also been explored in continual learning, however, focus only on layer-level adjustments (i.e., adding or removing entire layers), such as Exhubert [12]. In addition, many loss designs have been proposed to mitigate catastrophic forgetting [13], but these methods merely adjust weights that should be updated, without altering the number of parameters.

Previous algorithms have only sparsely explored pioneering 'drop-in–like' strategies, such as determining when and where to add neurons [14]. These methods identify triggers based on gradient information [15], exposure to unseen experiences [16], or model performance [17]. However, they do not explicitly draw inspiration from the mammalian brain to formulate dropin as a general technique; instead, their motivation is largely analytical, deriving from mathematical formulations aimed at increasing model capacity. Moreover, although various dropout and structural pruning strategies have been developed, they have not been integrated with dropin mechanisms to mirror the concepts of neuroplasticity.

Therefore, we propose a more fine-grained algorithm that incorporates a clearly defined dropin strategy alongside a novel, neuron-level plasticity mechanism. We evaluate this approach on multiple deepfake audio detection model clusters using the ASVspoof2019 LA, PA [10], and FakeorReal (FoR) [18] dataset. Experimental results show that the Equal Error Rate (EER) is considerably reduced and training time is shortened, all while maintaining limited memory usage. This leads to our **contributions**: To the best of our knowledge, this is the first neuron-level clearly defined dropin and novel plasticity algorithm that have been verified by experiments. Additionally, it enables state-of-the-art (SOTA) performance on ASVspoof2019 LA while preserving memory efficiency and reducing training time. Moreover, it is applicable to a variety of models.

II. Related Work

Previous work has explored brain-inspired methods from both biological and deep learning perspectives, providing evidence that such designs can improve both performance and computational efficiency [19], [20]. Representative examples include spiking neural networks with brain-like signalling mechanisms [21], Hebbian synaptic learning for representation learning [22], [23], and hippocampus-inspired neuroplasticity algorithms [24]. While these approaches demonstrate the potential of brain-inspired designs to enhance neural network performance, they are generally not scalable to a wide range of conventional neural network architectures.

Plasticity-related approaches, such as structural pruning and progressive networks, have been shown in prior work to be effective, although they are not always explicitly motivated by biological principles. Progressive networks [25] demonstrate that dynamically growing a network can enhance its learning capacity, while other studies increase model capacity by stacking multiple layers, for example, in ExHuBERT [12] and NN-stacking [26]. Additionally, adaptive pruning methods based on feature information entropy have been proposed to reduce training costs while maintaining performance [27]. Moreover, synaptic plasticity–based regularisation techniques have been introduced to reduce the complexity of certain network architectures [28]. However, these methods do not model a comprehensive process of neurogenesis and plasticity.

Even works that more closely mimic neuroplasticity, such as determining neuron addition and pruning based on certain triggers, have been proposed. Related methods trigger insertion based on gradient information [29], exposure to novel experiences [30], or other heuristics. However, these approaches are not grounded in a unified biological model of neurogenesis or neuroplasticity, relying instead on task- or heuristic-specific criteria.

Staying on the theory side cannot by itself prove algorithm effectiveness. Audio deepfake detection is an application that requires substantial feature understanding [31], [32]. While some small models have been shown to be effective [33], and LLMs are also being explored for this task [34], practical deployment often raises additional challenges. In particular, selecting an appropriate network size is crucial: a model that is too small may underfit and fail to capture decisive features [35], while a model that is too large may be inefficient and not generalise [36]. Dynamically adjusting networks with a neuroplasticity structure provides a flexible mechanism to maintain performance under computational constraints and evolving data distributions.

III. Methodology: Neuron-level dropin & Plasticity

Our method addresses the parameter bottleneck by selectively adding neurons and parameters to alleviate capacity limitations. Specifically, we propose dropin neuron addition on certain layers as a preliminary step to validate the effectiveness of neuron-level expansion. To avoid uncontrolled model growth while leveraging the benefits of dropin, we further introduce plasticity, which first adds neurons and subsequently prunes them to maintain a compact model size.

A. Dropin algorithm

To more closely emulate the neurogenesis observed in the mammalian brain, rather than extending entire layers, we introduce additional neurons selectively into specific layers. We add neurons to randomly selected layers and freeze the remainder of the network, training only the newly introduced neurons to examine their effects and behavior. This design choice is made as the complexity of determining where and when neurons are added in the mammalian brain is not easy to replicate in common state-of-the-art artificial neural networks. In addition to a a general experimental validation of the mechanism, we evaluate the effects of adding neurons at different layers from a statistical perspective. This approach operates at a more granular level and offers greater flexibility for the learning

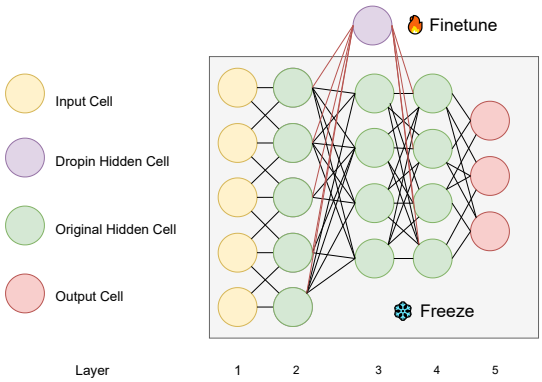

Fig. 1. High-level dropin process: During the dropin phase, we load the original pretrained model weights and keep them frozen. Additional neurons are then introduced into randomly selected layers, and only the connection weights associated with these newly added neurons are trained.

process, as individual neurons can be modulated directly by the algorithm. The overall framework is illustrated in Fig. 1.

The above figure provides a simplified representation of a general neural network, in which each cell may represent any type of neuron and the connections correspond to the parameters of the model. As such, the proposed algorithm is not restricted to a specific architecture and can be applied to various network designs, including convolutional, encoder-based, and recurrent layers. More detailed illustrations of the implementation of our dropin mechanism for different architecture types are presented in Fig. 2.

The dropin weights are concatenated as block matrices to facilitate matrix multiplication, enabling architectural adaptation at the granularity of individual neurons. Specifically, the channel dimension of certain layers is expanded, allowing additional neurons to be incorporated into the convolutional kernels for CNN. Likewise, the hidden dimensions of the update and reset gate weights in the GRU are enlarged, introducing more neurons within these gating mechanisms. Furthermore, in terms of attention encoder, the weight matrices used to generate the query, key, and value representations in attention mechanisms are expanded, permitting the inclusion of additional neurons. Importantly, the core computational mechanisms remain unchanged, for example, the convolution operations themselves are preserved, while the network capacity is increased through the addition of neurons. This demonstrates that the dropin approach is broadly applicable to a wide range of neural network architectures. For simplicity, we present a high-level version to illustrate the plasticity algorithm.

Additionally, this algorithm leverages pretrained models, which are now predominant in fine-tuning applications. In contrast to previous literature, which typically fine-tunes the entire network, our approach updates only the dropin neurons. This strategy considerably reduces training time while enhancing the model's capability through a lightweight parameter increase, rather than relying on extensive stacking of layers. Related work in continual learning has explored similar ideas,

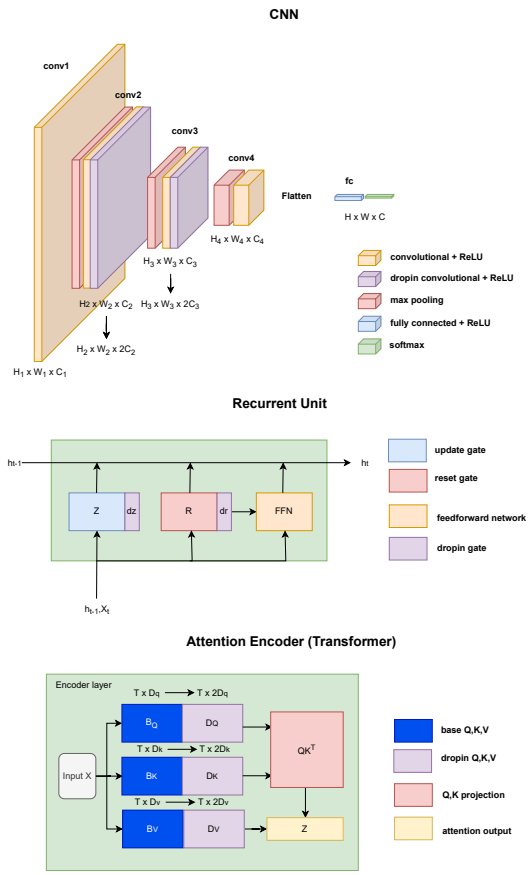

Fig. 2. Convolutional layers, recurrent units, and attention encoders dropin process. These represent the specific dropin techniques adapted to different model architectures. From a mathematical perspective, this involves increasing the kernel size for CNNs, expanding the gate weight dimensions for GRUs, and enlarging the query, key, and value weight dimensions for attention mechanisms.

such as Low-Rank Adaptation (LoRA) [37], which introduces additional low-rank matrices to acquire new information and update existing parameters. However, LoRA does not modify the network architecture to the same extent as our proposed drop-in mechanism. Instead, it introduces additional adapters that remain largely decoupled from the original architecture, making it less aligned with our biologically inspired motivation based on mammalian brain function. Additionally, LoRA is mainly suited for attention-based models, as it relies on updating a small number of parameters relative to the original weights. Therefore, its efficiency drops when many small layers lead to a large overall parameter count. In contrast, our method applies to any architecture by directly generating new neurons, independent of component parameterisation.

B. Plasticity algorithm

Following the dropin process, the memory usage of the model increases. In contrast, the mammalian brain exhibits a natural mechanism of eliminating redundant or unused

neurons to optimise efficiency. Building on the neuron-level granularity of the dropin mechanism, we take a further step by proposing a complementary algorithm, termed plasticity. The proposed pipeline is illustrated in Fig. 3. This approach aims to push beyond the model's capability limits on the target task. Accordingly, we design the entire training pipeline with an emphasis on maintaining the overall parameter count rather than optimising for time efficiency. The process begins with standard training of the model. **Unlike the controlled dropin experiments described above, where existing neurons are frozen to isolate the effects of newly added ones, in this stage we allow all parameters to remain trainable.** Our goal here is to investigate a complete biologically inspired setting from the neurogenesis to the neuroapotosis stage, in which network-wide interactions, analogous to biochemical processes in the brain, are permitted to emerge prior to the selective removal or deactivation of neurons.

The algorithm is divided into three stages. First, the model is trained for a number of epochs until its performance reaches a plateau. In the second stage, new neurons are introduced and the model continues training without reinitialising or freezing any previously learned parameters, such that all existing weights are updated jointly. This continuous training process enables the expanded architecture to adapt to the task and to reorganise its internal representations. As in the dropin procedure, the new neurons are added to randomly selected layers; for simplicity, the number of added neurons matches that of the original architecture. After this stage, the added neurons are pruned, and the full model undergoes a final phase of continued training, again with parameter initialisation from the previous training step. The algorithm is summarised in Algorithm 1.

This not only restores the original parameter size but also preserves the performance improvements gained during the intermediate expansion phase. We note that a promising direction for future investigation, once the mechanisms by which the brain prunes neurons are better understood, is to develop principled criteria for determining which neurons should be removed. At the current stage, however, we prune only the newly added neurons in order to control the number of trainable parameters and keep the overall model size constant. We contend that the plasticity algorithm could endow the model with a dynamic memory space, enabling it to retain valuable knowledge across all three stages while discarding irrelevant information, which forms the core intuition driving the model's breakthrough in performance. Each stage has been trained for equal epoches for a fair comparison.

Algorithm 1 Neuroplasticity Training

Require: Network $\mathcal{N}(\theta)$ with layers $\{L_i\}_{i=1}^{K}$, dataset $\mathcal{D}$
Ensure: Final parameters θ^*_{pruned}

1: Initial parameters:

$$\theta_{\text{init}}, \quad \theta_{\text{init}} \sim \mathcal{N}(0, \sigma^2)$$

2: Initial training:

$$\theta^*_{\text{init}} = \arg\min_{\theta_{\text{init}}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}\big(f(x;\theta_{\text{init}}), y\big)\right]$$

3: Sample layer indices:

$$\mathcal{I} \sim \text{Uniform}(\{1, \ldots, K\})$$

4: **for** each $i \in \mathcal{I}$ **do**

5: Neuron expansion in layer L^i with weights θ^i_{init}:

$$\theta^i_{\text{dropin}} \leftarrow \theta^i_{\text{init}} \cup \theta^i_{\Delta}, \quad \theta^i_{\Delta} \sim \mathcal{N}(0, \sigma^2), |\theta^i_{\Delta}| = |\theta^i_{\text{init}}|$$

6: **end for**

7: Continued training with dropin neurons:

$$\theta^*_{\text{dropin}} = \arg\min_{\theta_{\text{dropin}}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}\big(f(x;\theta_{\text{dropin}}), y\big)\right]$$

8: Neuron pruning removes all added neurons:

$$\theta_{\text{pruned}} \leftarrow \mathcal{P}\big(\theta^*_{\text{dropin}}\big), \quad |\theta_{\text{pruned}}| = |\theta_{\text{init}}|$$

9: Final continued training:

$$\theta^*_{\text{pruned}} = \arg\min_{\theta_{\text{pruned}}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}\big(f(x;\theta_{\text{pruned}}), y\big)\right]$$

10: **return** θ^*_{pruned}

IV. Experiments

We evaluate our method on the ASVspoof 2019 Logical Access (LA), physical access (PA) subset, and FoR, two widely used benchmarks for audio deepfake detection. Experiments are conducted on three representative models with distinct input modalities: ResNet18 (CNN, Mel spectrograms), GRNN (RNN, linear cepstral coefficients), and Wav2Vec 2.0 (attention-based, raw waveforms). This setup enables assessment of both architectural and feature diversity to verify the scalability of our pipeline. We fix the random seed to 42 for all random number generators (Python, NumPy, PyTorch) to ensure reproducibility. All experiments are run on an NVIDIA A6000 GPU, with other hyperparameters aligned to those in the original model papers [5]–[7]. We report the best-performing model for each setting. The maximum number of training epochs is determined through preliminary trials to ensure that each experimental configuration can converge within the allotted epochs. Model selection is performed based on validation set performance, and the selected model is subsequently evaluated on the test set. Wav2Vec 2.0 uses the pretrained model from Hugging Face[1], while ResNet18 and GRNN are trained from scratch for all experimental settings including baseline. To examine parameter scaling, we also train a scratch Wav2Vec variant with a reduced 256-dimensional hidden size. Full hyperparameter details are provided in Github link [2].

We apply the dropin and plasticity procedures once for each base model, introducing the same number of additional neurons as the original size of that layer. We compare the

[1]https://huggingface.co/facebook/wav2vec2-base
[2]https://github.com/ShawnNotFound/Dropin

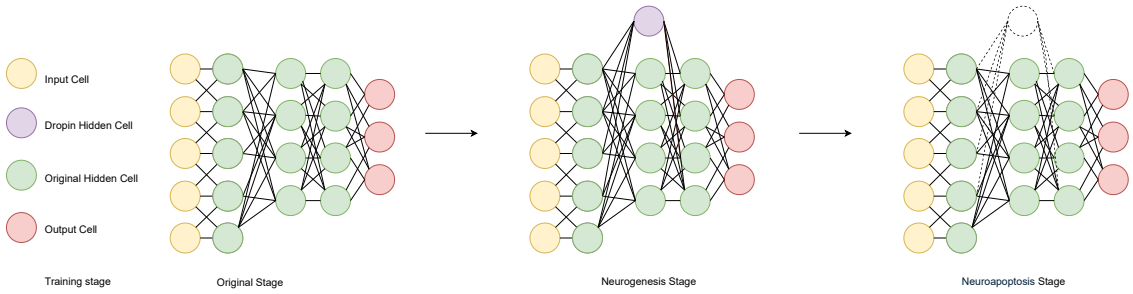

Fig. 3. Plasticity process: The pipeline is divided into three stages. The first stage is identical to conventional training, where the model is optimised using the standard objective function. In the second stage, new neurons are dropped in, a process analogous to neurogenesis. The whole model is then retrained. After the newly introduced information has been assimilated and distributed across existing neurons. The third stage emulates neuroapoptosis by pruning the added neurons, followed by a final retraining.

performance of five configurations: (i) the original baseline model, (ii) the dropin model without parameter freezing (representing an enlarged model capacity), (iii) LoRA-based fine-tuning (where applicable), (iv) our dropin method, and (v) our plasticity method.

V. Results

In the following, we discuss the results of all models on the three datasets presented in Table I.

Accuracy improvement on plasticity: The proposed dropin and plasticity strategies consistently outperform the baseline across multiple architectures and datasets in terms of Test EER. In particular, plasticity achieves the lowest EER in most settings, often by a large margin. For example, on the ASV LA dataset with Wav2Vec 2.0, plasticity reduces the EER from 2.45% (baseline) to 0.04%, substantially outperforming both dropin unfrozen (0.44%) and dropin frozen (1.64%). Similar improvements can be observed across ResNet and GRNN models on ASV LA and PA, demonstrating that the proposed mechanisms provide consistent accuracy gains over the original architectures.

Efficiency improvement on dropin: In terms of training efficiency, the dropin frozen strategy consistently achieves the lowest backward time among all compared methods. By freezing the original network and training only the newly introduced neurons, dropin frozen considerably reduces computational overhead while maintaining competitive detection performance. For instance, on ASV LA with ResNet, the backward time is reduced from 5.46,ms (baseline) to 3.49,ms, and on Wav2Vec 2.0, it is reduced from 0.24,ms to 0.18,ms. This efficiency advantage is also observed across PA and FoR datasets, indicating that dropin frozen is particularly suitable for scenarios where training speed is a primary concern.

Plasticity achieves a superior trade-off between parameter size and accuracy: When both accuracy and model size are considered jointly, plasticity emerges as the most favorable strategy. While dropin unfrozen occasionally achieves slightly lower EER, this improvement is primarily attributable to its substantially larger number of trainable parameters. In contrast, plasticity maintains the original model size and achieves comparable—or in many cases superior—performance. For example, on ASV PA with Wav2Vec 2.0, plasticity achieves an EER of 4.34%, compared to 7.54% for LoRA and 8.18% for dropin unfrozen, without increasing the number of trainable parameters. These results suggest that plasticity provides a more balanced trade-off between accuracy and computational cost for model size, making it particularly attractive for practical deployment.

Plasticity benefits more from larger models: We observe that plasticity yields more pronounced performance gains in larger-capacity models compared to smaller ones. In particular, when comparing Wav2Vec 2.0 with ResNet, plasticity leads to substantially larger reductions in Test EER for Wav2Vec 2.0 across all evaluated datasets. For example, on ASV LA, plasticity reduces the EER of Wav2Vec 2.0 from 2.45% to 0.04%, whereas the improvement on ResNet is more moderate, decreasing from 14.69% to 9.70%. This trend is consistently observed on ASV PA and FoR, suggesting that larger models with higher representational capacity provide a more favorable substrate for plasticity-driven adaptation. These results indicate that plasticity is particularly effective when applied to high-capacity architectures, where additional neurons and dynamic adaptation can be more fully exploited.

Comparative drawbacks of LoRA: Finally, LoRA consistently underperforms compared to both dropin-based strategies and plasticity across all evaluated models and datasets. Although LoRA introduces only a small number of trainable parameters, its detection performance remains close to or worse than the baseline in most cases. For instance, on ASV LA with Wav2Vec 2.0, LoRA achieves an EER of 2.08%, which is notably higher than plasticity (0.04%) and dropin unfrozen (0.44%). Similar trends are observed on PA and FoR datasets, indicating that parameter-efficient fine-tuning alone is insufficient for capturing the complex spoofing patterns addressed by our proposed methods but more mammal brain mimicing is needed.

We have also conducted **ablation studies** on effects of which layer to perform these mechanism and analyse explainability why the results performing better for our proposed method. Using the Wav2vec 2.0 small model with the dropin strategy as representatives, the results are presented in Fig. 5.

Dataset	Model	Training Strategy	Test EER (%)↓	Backward Time per step (ms)↓	Parameters (M)↓	Trainable Params (M)↓
ASV LA	ResNet	baseline	14.69	5.46	**11.17**	11.17
		dropin unfrozen	**7.60**	6.72	14.28	14.28
		dropin frozen (ours)	11.98	**3.49**	14.28	**1.47**
		plasticity (ours)	9.70	/	**11.17**	/
	GRNN	baseline	19.84	343.35	**0.06**	0.06
		dropin unfrozen	**16.64**	477.79	0.10	0.10
		dropin frozen (ours)	18.48	**240.45**	0.10	**0.03**
		plasticity (ours)	17.82	/	**0.06**	/
	Wav2Vec 2.0	baseline	2.45	0.24	**95.57**	95.57
		dropin unfrozen	0.44	0.25	102.83	102.83
		LoRA	2.08	/	95.57	**0.29**
		dropin frozen (ours)	1.64	**0.18**	102.83	8.47
		plasticity (ours)	**0.04**	/	**95.57**	/
	Wav2Vec 2.0 (Small)	baseline	11.06	1.15	**11.25**	11.25
		dropin unfrozen	**8.44**	1.21	14.21	14.21
		LoRA	11.10	/	11.25	**0.09**
		dropin frozen (ours)	12.20	**0.74**	14.21	3.02
		plasticity (ours)	9.51	/	**11.25**	/
ASV PA	ResNet	baseline	14.08	5.82	**11.17**	11.17
		dropin unfrozen	9.14	6.84	14.28	14.28
		dropin frozen (ours)	**6.11**	**3.79**	14.28	**1.47**
		plasticity (ours)	9.70	/	**11.17**	/
	GRNN	baseline	19.13	355.88	**0.06**	0.06
		dropin unfrozen	18.68	494.30	0.10	0.10
		dropin frozen (ours)	**17.50**	**250.41**	0.10	**0.03**
		plasticity (ours)	18.19	/	**0.06**	/
	Wav2Vec 2.0	baseline	7.89	94.23	**95.57**	95.57
		dropin unfrozen	8.18	86.64	102.83	102.83
		LoRA	7.54	/	95.57	**0.29**
		dropin frozen (ours)	7.87	**29.09**	102.83	8.47
		plasticity (ours)	**4.34**	/	**95.57**	/
	Wav2Vec 2.0 (Small)	baseline	11.98	21.94	**11.25**	11.25
		dropin unfrozen	14.72	25.25	14.21	14.21
		LoRA	16.52	/	11.25	**0.09**
		dropin frozen (ours)	11.97	**10.27**	14.21	3.02
		plasticity (ours)	**10.75**	/	**11.25**	/
FoR	ResNet	baseline	18.96	6.20	**11.17**	11.17
		dropin unfrozen	**12.72**	7.15	14.28	14.28
		dropin frozen (ours)	13.70	**3.97**	14.28	**1.47**
		plasticity (ours)	14.55	/	**11.17**	/
	GRNN	baseline	17.65	368.81	**0.06**	0.06
		dropin unfrozen	16.27	506.93	0.10	0.10
		dropin frozen (ours)	19.14	**255.87**	0.10	**0.03**
		plasticity (ours)	**14.05**	/	**0.06**	/
	Wav2Vec 2.0	baseline	21.81	193.55	**95.57**	95.57
		dropin unfrozen	21.13	202.77	102.83	102.83
		LoRA	16.67	/	95.57	**0.29**
		dropin frozen (ours)	24.30	**142.69**	102.83	8.47
		plasticity (ours)	**12.94**	/	**95.57**	/
	Wav2Vec 2.0 (Small)	baseline	33.27	252.30	**11.25**	11.25
		dropin unfrozen	20.02	364.57	14.21	14.21
		LoRA	30.53	/	11.25	**0.09**
		dropin frozen (ours)	19.82	**209.59**	14.21	3.02
		plasticity (ours)	**12.60**	/	**11.25**	/

TABLE I

COMPARISON OF DIFFERENT TRAINING STRATEGIES ACROSS ASV LA, PA, AND FOR DATASETS. DROPIN UNFROZEN DENOTES AN ENLARGED NETWORK TRAINED CONVENTIONALLY, WHILE BASELINE REFERS TO THE ORIGINAL MODEL. FOR PLASTICITY, BACKWARD TIME AND TRAINABLE PARAMETERS ARE NOT REPORTED AS THEY VARY ACROSS STAGES. THE SYMBOL "/" INDICATES UNAVAILABLE OR NON-APPLICABLE MEASUREMENTS.

It can be observed that dropping in neurons at different layers has a relatively small impact on overall performance. This suggests that neural networks are able to automatically leverage the added parameters to improve learning, regardless of the specific layer in which they are introduced. Importantly, this consistency indicates that our observed improvements are not due to randomness but represent a systematic effect. While some of the previous literature implies that selecting single layers based solely on mathematical criteria, such as gradients would be beneficial, we would argue that the selection process should rather be informed by processes in the mammalian brain, which are still to be understood well enough.

Lastly, we applied Grad-CAM [38] to each experimental setting to examine whether attention patterns change during training. As a case study, we selected the ResNet model on a single example from the ASVspoof 2019 LA dataset. The resulting attention visualisations are presented in Fig. 4.

It can be observed that in the baseline model, Grad-CAM activations are particularly prominent in the shallow layers (L1–L2), where focus is broadly distributed across edge regions. Although deeper layers show slightly more focused activations, the highlighted areas remain coarse. This suggests that

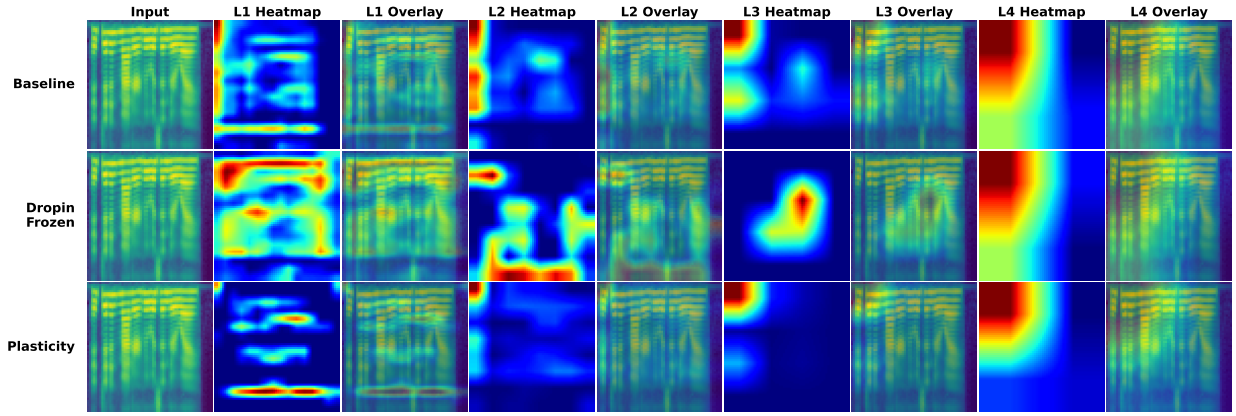

Fig. 4. GradCAM results on Resnet for one case from ASVSpoof 2019 LA.

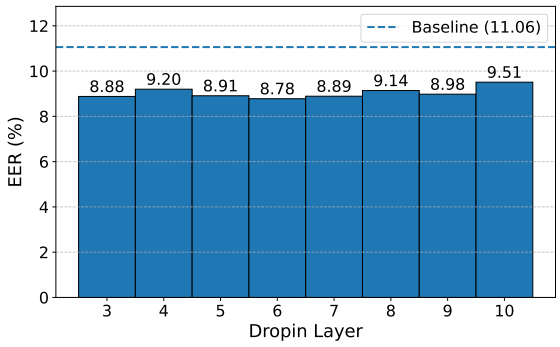

Fig. 5. EER results for different dropin layers. "3" represents that we dropped in new neurons on the 3rd encoding layer in Wav2vec 2.0 small on ASVSpoof2019 LA.

the baseline relies on redundant or spurious features, resulting in limited feature selectivity and suboptimal hierarchical representations. However, under the dropin configuration, GradCAM maps become noticeably more compact from intermediate layers onwards. Compared to the baseline, the focus is more concentrated on many feature maps especially L1, which provides a potential explanation for the observed performance improvements. For the plasticity configuration, focus is even more selective across layers. Early layers exhibit relatively low activations, indicating enhanced feature selection capability, while deeper layers show highly focused and semantically aligned responses. By keeping parameters size the same, the model is able to adaptively reorganize feature representations, suppress non-informative responses, and achieve improved cross-layer consistency. Notably, the last layer's attention of the three remains relatively similar, suggesting that there is still room for refinement in future work to further enhance focus and performance.

VI. Conclusion

Our work investigated the use of the newly introduced dropin and plasticity algorithms, which can be applied to various model architectures. We applied it in the field of audio deepfake detection to enhance both performance and efficiency in terms of computation and parameter usage. The model achieved SOTA performance on the ASVSpoof2019 LA subset. In scenarios with strict constraints on model size such as deployment on mobile devices or applications demanding high efficiency, including real-time update and detection tasks, our method presents a more practical and effective alternative. In future work, we aim to explore a biologically inspired plasticity algorithm that more closely mimics the behaviour of the mammalian brain. Specifically, this would involve structural pruning to remove neurons that are functionally redundant, a process that requires further rigorous definition.

References

[1] I. R. McKenzie, A. Lyzhov, M. Pieler, A. Parrish, A. Mueller, A. Prabhu, E. McLean, A. Kirtland, A. Ross, A. Liu, A. Gritsevskiy, D. Wurgaft, D. Kauffman, G. Recchia, J. Liu, J. Cavanagh, M. Weiss, S. Huang, T. F. Droid, T. Tseng, T. Korbak, X. Shen, Y. Zhang, Z. Zhou, N. Kim, S. R. Bowman, and E. Perez, "Inverse scaling: When bigger isn't better," *Transactions on Machine Learning Research*, 2023.

[2] Y. Chu, J. Xu, X. Zhou, Q. Yang, S. Zhang, Z. Yan, C. Zhou, and J. Zhou, "Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models," *arXiv preprint arXiv:2311.07919*, 2023.

[3] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[4] Z. Almutairi and H. Elgibreen, "A review of modern audio deepfake detection methods: challenges and future directions," *Algorithms*, vol. 15, no. 5, p. 155, 2022.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[6] A. Gomez-Alanis, A. M. Peinado, J. A. Gonzalez, and A. M. Gomez, "A light convolutional gru-rnn deep feature extractor for asv spoofing detection," in *Proc. Interspeech*, vol. 2019, 2019, pp. 1068–1072.

[7] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[8] J.-w. Jung, H.-S. Heo, H. Tak, H.-j. Shim, J. S. Chung, B.-J. Lee, H.-J. Yu, and N. Evans, "Aasist: Audio anti-spoofing using integrated spectro-temporal graph attention networks," in *ICASSP 2022-2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2022, pp. 6367–6371.

[9] H. Tak, J. Patino, M. Todisco, A. Nautsch, N. Evans, and A. Larcher, "End-to-end anti-spoofing with rawnet2," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6369–6373.

[10] X. Wang, J. Yamagishi, M. Todisco, H. Delgado, A. Nautsch, N. Evans, M. Sahidullah, V. Vestman, T. Kinnunen, K. A. Lee *et al.*, "Asvspoof 2019: A large-scale public database of synthesized, converted and replayed speech," *Computer Speech & Language*, vol. 64, p. 101114, 2020.

[11] Y. Li, M. Milling, and B. W. Schuller, "Neuroplasticity in artificial intelligence – an overview and inspirations on drop in & out learning," 2025.

[12] S. Amiriparian, F. Packań, M. Gerczuk, and B. W. Schuller, "Exhubert: Enhancing hubert through block extension and fine-tuning on 37 emotion datasets," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*. Kos Island, Greece: International Speech Communication Association, 2024, pp. 2635–2639.

[13] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton, "Loss of plasticity in deep continual learning," *Nature*, vol. 632, no. 8026, pp. 768–774, 2024.

[14] K. Maile, E. Rachelson, H. Luga, and D. G. Wilson, "When, where, and how to add new neurons to anns," in *Proceedings of the First International Conference on Automated Machine Learning*, ser. Proceedings of Machine Learning Research, I. Guyon, M. Lindauer, M. van der Schaar, F. Hutter, and R. Garnett, Eds., vol. 188. PMLR, 25–27 Jul 2022, pp. 18/1–12.

[15] U. Evci, B. van Merriënboer, T. Unterthiner, M. Vladymyrov, and F. Pedregosa, "Gradmax: Growing neural networks using gradient information," in *International Conference on Learning Representations (ICLR)*, 2022.

[16] P. Eriksson and L. Westlund Gotby, "Dynamic network architectures for deep q-learning: Modelling neurogenesis in artificial intelligence," 2019.

[17] A. Sowrirajan, P. Srinivasan, S. A. Srinivasan, and S. P. Devi, "Enhancing neurofuzzy plasticity: A fusion of lstm and artificial neurogenesis," in *2024 International Conference on Emerging Techniques in Computational Intelligence (ICETCI)*. IEEE, 2024, pp. 150–154.

[18] R. Reimao and V. Tzerpos, "For: A dataset for synthetic speech detection," in *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*. IEEE, 2019, pp. 1–10.

[19] S. Woźniak, A. Pantazi, T. Bohnstingl, and E. Eleftheriou, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 325–336, 2020.

[20] G. Dellaferrera, S. Wozniak, G. Indiveri, A. Pantazi, and E. Eleftheriou, "Learning in deep neural networks using a biologically inspired optimizer," *arXiv preprint arXiv:2104.11604*, 2021.

[21] S. Schmidgall, R. Ziaei, J. Achterberg, L. Kirsch, S. Hajiseyedrazi, and J. Eshraghian, "Brain-inspired learning in artificial neural networks: a review," *APL Machine Learning*, vol. 2, no. 2, 2024.

[22] N. Ravichandran, A. Lansner, and P. Herman, "Unsupervised representation learning with hebbian synaptic and structural plasticity in brain-like feedforward neural networks," *Neurocomputing*, vol. 626, p. 129440, 2025.

[23] A. Journé, H. G. Rodriguez, Q. Guo, and T. Moraitis, "Hebbian deep learning without feedback," *arXiv preprint arXiv:2209.11883*, 2022.

[24] T. Rudroff, O. Rainio, and R. Klen, "Neuroplasticity meets artificial intelligence: A hippocampus-inspired approach to the stability–plasticity dilemma," *Brain Sciences*, vol. 14, no. 11, p. 1111, 2024.

[25] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[26] V. Coscrato, M. H. de Almeida Inacio, and R. Izbicki, "The nn-stacking: Feature weighted linear stacking through neural networks," *Neurocomputing*, vol. 399, pp. 141–152, 2020.

[27] S. Chen, Z. Liu, and H. You, "Brain-inspired efficient pruning: Exploiting criticality in spiking neural networks," *Concurrency and Computation: Practice and Experience*, vol. 37, no. 27-28, p. e70404, 2025.

[28] Q. Yousef and P. Li, "Synaptic plasticity-based regularizer for artificial neural networks," *Scientific Reports*, vol. 15, no. 1, p. 14330, 2025.

[29] T. Miconi, K. Stanley, and J. Clune, "Differentiable plasticity: training plastic neural networks with backpropagation," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3559–3568.

[30] J. N. Heidenreich, C. Bonatti, and D. Mohr, "Transfer learning of recurrent neural network-based plasticity models," *International Journal for Numerical Methods in Engineering*, vol. 125, no. 1, p. e7357, 2024.

[31] Y. Li, M. Milling, L. Specia, and B. W. Schuller, "From audio deepfake detection to ai-generated music detection–a pathway and overview," *arXiv preprint arXiv:2412.00571*, 2024.

[32] T. Chen, A. Kumar, P. Nagarsheth, G. Sivaraman, and E. Khoury, "Generalization of audio deepfake detection." in *Odyssey*, 2020, pp. 132–137.

[33] L. Pham, P. Lam, T. Nguyen, H. Nguyen, and A. Schindler, "Deepfake audio detection using spectrogram-based feature and ensemble of deep learning models," in *2024 IEEE 5th International Symposium on the Internet of Sounds (IS2)*. IEEE, 2024, pp. 1–5.

[34] Y. Li, L. Wang, Y. Wang, L. Wang, R. Cai, J. Shi, B. W. Schuller, and Z. Wu, "Dfallm: Achieving generalizable multitask deepfake detection by optimizing audio llm components," *arXiv preprint arXiv:2512.08403*, 2025.

[35] K. Shahriar, "Lightweight resolution-aware audio deepfake detection via cross-scale attention and consistency learning," *arXiv preprint arXiv:2601.06560*, 2026.

[36] H. Gu, J. Yi, C. Wang, J. Tao, Z. Lian, J. He, Y. Ren, Y. Chen, and Z. Wen, "Allm4add: Unlocking the capabilities of audio large language models for audio deepfake detection," in *Proceedings of the 33rd ACM International Conference on Multimedia*, 2025, pp. 11 736–11 745.

[37] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models." *ICLR*, 2022.

[38] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.