LOAD BALANCING NEURONS: CONTROLLING FIRING RATES IMPROVES PLASTICITY IN CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural networks in continual learning often lose plasticity: some neurons become inactive, while others fire almost constantly. This limits adaptation to shifting data and wastes capacity. Prior work mitigates this by periodically reinitializing low-utility units, but such resets can destroy previously learned features and do not proactively prevent low utility. We study a simple diagnostic measure: the firing rate of ReLU units, defined as the fraction of positive pre-activations. Low rates identify dead units, while very high rates indicate linearized, always-on units. Based on this view, we introduce a lightweight load-balancing mechanism that adjusts per-neuron thresholds to keep firing rates within a target range. Across continual ImageNet and class-incremental CIFAR-100, improvements in firing-rate distributions help explain differences in plasticity across approaches, including our load-balancing mechanism and well-known techniques, notably L2 regularization and non-affine normalization.

1 INTRODUCTION

Deep learning models have achieved state-of-the-art performance on a broad range of tasks when trained on fixed datasets (He et al., 2016; Liu et al., 2022). However, performance often degrades in continual learning settings, where the data distribution evolves over time (Khetarpal et al., 2022; Lee et al., 2023; Abbas et al., 2023). Simply continuing training on new data often leads to optimization instabilities and catastrophic forgetting, and the mechanisms behind these failures are not yet fully understood (Lyle et al., 2024). A common remedy is to retrain a new model from scratch, which restores accuracy but is often impractical due to computational cost.

Ideally, a continual learning system would strike a balance between *plasticity*, the ability to adapt to new data, and *stability*, the ability to preserve what it has already learned (Carpenter, 1987; Elsayed & Mahmood, 2024). Declining stability is known as *catastrophic forgetting* (McCloskey & Cohen, 1989), while declining plasticity is commonly called *plasticity loss*. Overcoming these challenges would allow models to evolve as data changes over time, mimicking the way humans and animals continuously learn without retraining from scratch.

Contributions:

- We introduce firing-rate analysis as a simple, interpretable diagnostic tool for neuron utilization and plasticity loss in continual learning, offering a unified view across tasks and architectures.
- We analyze and compare the firing-rate distributions of existing methods, including L2 regularization and non-affine normalization, and show how these patterns relate to plasticity loss.
- We propose a load-balancing mechanism that dynamically adjusts neuron activity by shifting activation inputs. It keeps firing rates balanced and outperforms previous baselines.

2 BACKGROUND

Plasticity loss. The phenomenon of plasticity loss has been observed in both supervised learning (Ash & Adams, 2020; Berariu et al., 2021; Dohare et al., 2024) and reinforcement learning (Lyle et al., 2022; Nikishin et al., 2022; Dohare et al., 2024). Several approaches have been proposed to improve plasticity. For instance, injecting noise during training (Ash & Adams, 2020), periodically resetting the last layers (Nikishin et al., 2022), expanding the network during training (Nikishin et al., 2023), using the Concatenated ReLU activation function Abbas et al. (2023), reinitializing low-utility neurons (Sokar et al., 2023; Dohare et al., 2024), perturbing gradients of low-utility neurons Elsayed & Mahmood (2024), and regularizing weights toward their initial values Kumar et al. (2024). Some studies combine multiple techniques. For example, Lee et al. (2023) combines layer resets with Concatenated ReLUs, LayerNorm (Ba et al., 2016), and the sharpness-aware minimization optimizer (Foret et al., 2021).

Many of these interventions restore plasticity by potentially destroying learned information, by resetting neurons or entire layers of the networks, rather than preventing collapse in the first place. In contrast, we ask how to sustain plasticity proactively through minimal architectural changes.

Measuring neuron utility. Many of the above methods rely on a measure of neuron utility. For example, Dohare et al. (2024) define a neuron's *contribution utility* as the magnitude of its activation multiplied by the magnitudes of all outgoing weights, tracked as a moving average during training. Sokar et al. (2023) define the *dormancy score* as the neuron's activation magnitude, normalized by the magnitudes of other neurons in the same layer and averaged over the dataset; a neuron is considered *dormant* if the dormancy score falls below a threshold. Elsayed & Mahmood (2024) define *weight utility* as the change in loss after zeroing a weight, which yields a global ranking over all weights but is more expensive to compute and requires second-order Taylor approximations.

In contrast, we introduce a firing-rate diagnostic that reveals the *distribution* of neuron utilization, across layers and over time, rather than collapsing activity into a single summary statistic.

Continual learning benchmarks. Many benchmarks have been proposed to evaluate continual learning, but there is no single standard, and new variants are often introduced, making comparisons difficult. In this work, we adopt the benchmarking setup of Dohare et al. (2024) to assess our method. We therefore focus on the *continual ImageNet* and *class-incremental CIFAR-100* benchmarks, which they used to evaluate continual backpropagation (CBP).

3 PLASTICITY THROUGH THE LENS OF FIRING RATES

Plasticity loss often reflects neuron-level collapse in ReLU networks, where some neurons stop firing, and others become always active. Plasticity, in turn, depends on whether a neuron spends time on both sides of its threshold. We therefore use the *firing rate*, the fraction of pre-activations for which a neuron is active, as a diagnostic tool of neuron use. Tracking firing-rate distributions reveals training anomalies, and keeping them in a balanced range correlates with improved adaptation under distribution shift as we will demonstrate in the experimental section.

Firing rate. A ReLU implements a two-regime, piecewise-linear map: *off* for $x < 0$ (slope 0) and *on* for $x \geq 0$ (slope 1). We therefore use the firing rate to diagnose whether a neuron remains stuck in one regime or alternates between them over inputs and time. Concretely, consider a pre-activation x passed through a ReLU, producing $y = \max(0, x)$. For a batch of pre-activations $x_1, \ldots, x_n \in \mathbb{R}$ for a particular neuron, define the *instantaneous batch firing rate* as

$$\rho = \frac{1}{n} \sum_{i=1}^{n} [x_i \geq 0], \tag{1}$$

where $[\cdot] \in \{0, 1\}$ denotes the Iverson bracket. Thus, ρ is an empirical estimate of $\Pr(x \geq 0)$ under the current data distribution. Because continual learning is non-stationary, instantaneous batch estimates can be volatile. We therefore track the *exponentially averaged firing rate*, denoted $\bar{\rho}$, to track the longer-term trend:

$$\bar{\rho} \leftarrow \tau \bar{\rho} + (1 - \tau)\rho, \tag{2}$$

with decay $\tau \in [0, 1]$. We initialize $\bar{\rho}$ to 50% and use $\tau = 0.99$ in our experiments. The exponential moving average smooths batch noise while remaining responsive to a changing data distribution.

Interpretation. The firing rate $\bar{\rho}$ is directly tied to the unit's nonlinearity. Near $\bar{\rho} = 0$ the neuron is *dead* or *dormant* and contributes neither activation nor gradient. Near $\bar{\rho} = 1$ it is a linear pass-through with $\frac{\partial y}{\partial x} = 1$. Gradients continue to flow, but its behavior is linear. In both extremes the neuron adds no curvature to the overall network, reducing effective capacity. Intermediate firing rates correspond to neurons that regularly cross the threshold in both directions, providing nonlinearity to shape decision boundaries.

Connection to plasticity loss. Plasticity loss at the neuron level manifests as saturation of $\bar{\rho}$ toward 0 or 1. In either case, the neurons ability to adapt its decision boundary to shifting data diminishes, even if its weights continue to update. This view motivates using firing rates both as a diagnostic for plasticity and as a control signal for the load-balancing mechanism introduced later.

3.1 Optimal firing rate

From a functional standpoint, a ReLU is most useful when it alternates between active and inactive regimes. This occurs when it is *on* about half the time, suggesting an optimal firing rate of 50%.

An information-theoretic view makes this precise: view a neuron's activation state as a binary random variable $S = [x \geq 0]$. Under the pre-activation distribution encountered during continual training, $S \sim \text{Bernoulli}(p)$ with $p = \Pr(x \geq 0)$, which we estimate with the average firing rate $\bar{\rho}$. The Shannon entropy

$$H(p) = -p \log_2 p - (1 - p) \log_2 (1 - p) \tag{3}$$

quantifies how unpredictable the activation state is across the data stream. Higher entropy means the state carries more information. Formally, because S is a deterministic function of the pre-activation x, the mutual information between x and S equals the entropy, i.e., $I(x; S) = H(S)$. Maximizing the mutual information $I(x; S)$ therefore amounts to maximizing $H(S)$, which has a unique maximum at $p = 0.5$. Departures toward 0 or 1 reduce entropy and hence reduce the neuron's effective capacity and plasticity. This is a per-neuron argument; network-level entropy also depends on dependencies among units, as discussed in Appendix B.

While this is a per-neuron, theoretical argument, our experiments will show that targeting a firing rate near 50% often yields the strongest performance.

3.2 Firing-rate dynamics in continual benchmarks

In this subsection, we study firing-rate behavior on the continual ImageNet and class-incremental CIFAR-100 benchmarks. These settings build on widely used datasets, are tailored to CNN and ResNet architectures that provide a stronger test of plasticity than simpler MLP models, and are straightforward to interpret. Together, they offer a useful testbed for examining how neuron firing rates evolve under distribution shift.

Continual ImageNet: firing-rates and accuracy. In the continual ImageNet benchmark (Dohare et al., 2024), the model is trained on a sequence of binary classification tasks. For each task it learns to discriminate two randomly sampled classes from ImageNet (Deng et al., 2009) before moving on to a new pair. Because all tasks have comparable difficulty, accuracy degradation across tasks indicates reduced plasticity (Dohare et al., 2024). Following Dohare et al. (2024), images are downsampled to 32×32 resolution, which preserves ImageNet's class diversity while making long sequential runs computationally feasible for thousands of tasks. This setup is substantially more demanding than MNIST-scale benchmarks. We use a compact CNN with ReLU activations (see Appendix D for training details).

Firing-rate distributions and test accuracies are shown in Figure 1. Accuracy curves report the mean over multiple seeds, with shaded bands denoting ± 1 standard error of the mean. Because tasks are formed by randomly sampled binary class pairs, this benchmark exhibits higher run-to-run variability. The error bands make this variability explicit without altering the overall ranking among

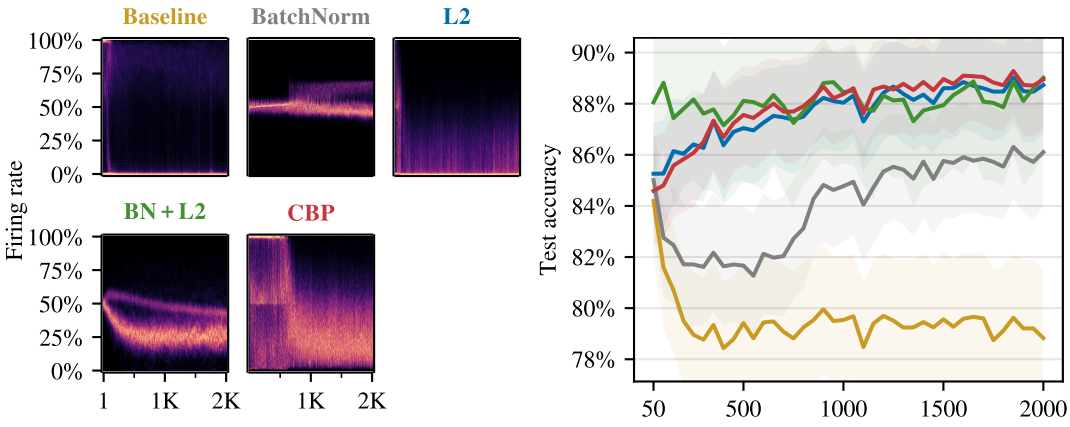

Figure 1: **Continual ImageNet:** Evolution of firing rate distributions (left) and test accuracy with error bands (right), averaged over 10 seeds. Standard training collapses neuron activity, while BatchNorm, L2 regularization, and CBP maintain more balanced firing. The temporal alignment between balanced firing rates and rising accuracy highlights their connection to plasticity.

methods, and results are best interpreted in terms of the trajectory of accuracy curves, which captures plasticity dynamics. Absolute accuracies for all methods and tasks are listed in Appendix G.

Throughout this paper, firing-rate distributions are visualized as two-dimensional heatmaps: the horizontal axis denotes training progress (number of tasks), the vertical axis spans firing rates from 0 to 100%, and each column shows a histogram over all neurons in the network. Brighter regions indicate many neurons with that firing rate, whereas darker regions indicate fewer. These plots reveal how activity patterns evolve over time.

Key findings:

1. **Baseline.** Continual training on changing tasks leads to rapid accuracy deterioration after only a few tasks, consistent with prior findings (Dohare et al., 2024). Firing rates collapse toward near 0% for most neurons, aligning with the drop in accuracy due to limited plasticity.
2. **BatchNorm.** Adding Batch Normalization (Ioffe & Szegedy, 2015) improves accuracy and plasticity over the baseline. The firing rates do not collapse and accuracy eventually improves after $\approx$ 500 tasks. This illustrates that firing rates are more closely tied to plasticity than to accuracy.
3. **L2.** With L2 regularization, test accuracy is almost as good as CBP. The firing rates show a distinctive pattern: many collapse to near 0%, whereas consistently some are in the 1-50% range. Plasticity remains strong in this case, indicating that no single firing-rate distribution universally characterizes plasticity.
4. **BatchNorm + L2.** Combining Batch Normalization and L2 regularization prevents extreme firing rates. Both accuracy and plasticity remain high in this setting.
5. **CBP.** Continual Backpropagation (Dohare et al., 2024) largely prevents neurons from collapsing to 0% firing rates. Early in training, many neurons saturate near 100%, indicating linearized ReLU behavior. After $\approx$ 500 tasks, the distribution re-centers and broadens, producing more balanced activity and coinciding with improved accuracy. While purely correlational, the temporal alignment between more balanced firing-rate distributions and accuracy suggests that maintaining intermediate firing rates supports plasticity.

Class-incremental CIFAR-100: firing-rates and accuracy. In the class-incremental CIFAR-100 benchmark, the model begins by learning to classify a small set of randomly sampled classes fomr CIFAR-100 (Krizhevsky et al., 2009). Additional classes are gradually introduced, and the model must classify all classes encountered so far until the full set of 100 is included. Because the classification problem grows harder as more classes accumulate, performance can decline even without

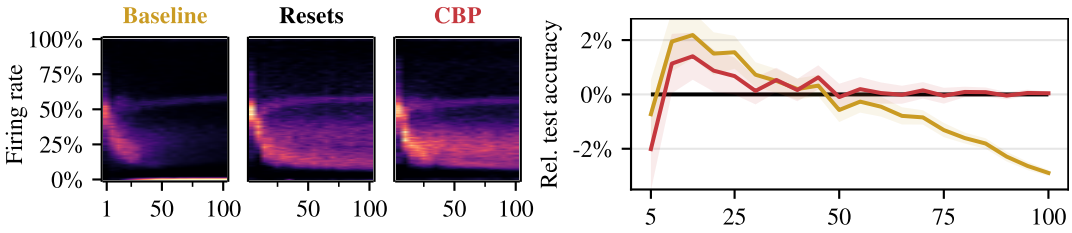

Figure 2: **Class-incremental CIFAR-100:** Firing-rate distributions (left) and test accuracy with error bands (right), averaged over 15 seeds. Baseline training leads to gradual neuron inactivity, while resets and CBP maintain non-collapsed firing patterns. Accuracy closely follows these trends, indicating that balanced firing rates preserve plasticity.

plasticity loss (Dohare et al., 2024). To disentangle this effect, we compare continual training with a model trained from scratch on the same subset. If the continually trained model performs worse than a model trained from scratch, this indicates plasticity loss, whereas superior performance signals knowledge transfer (Dohare et al., 2024). We use a ResNet-18 with ReLU activations (see Appendix D for training details).

Firing-rate distributions and test accuracies are shown in Figure 2. Results are averaged over multiple seeds, with shaded bands denoting ± 1 standard error of the mean.

Key findings:

1. **Baseline.** As in continual ImageNet, neurons progressively fall inactive, though the collapse is less drastic. Over time, many units reach zero firing, limiting the model's capacity to learn new tasks.
2. **Resets.** Resetting the network after each task prevents firing-rate collapse. Neurons consistently adapt to the new classes from their initialization. The distribution shifts as the number of classes grows, reflecting the increasing difficulty of the task.
3. **CBP.** The firing-rate distribution under Continual Backpropagation closely resembles the Resets setup. Test accuracy closely tracks the Resets baseline, and firing rates are concentrated in the 10-30% range with some units reaching around 60%. This pattern indicates that neurons remain active and avoid collapse without full network reinitialization.

Summary of findings. In summary, firing-rate analysis provides a straightforward lens on plasticity in continual learning. Across both benchmarks, conventional training rapidly leads to inactive neurons, while CBP maintains balanced firing-rate distributions that closely resemble those of fully reset networks. This shows that CBP preserves plasticity without reinitialization and validates firing rates as a useful diagnostic tool.

4 Load Balancing: Bounding Per-Neuron Firing Rates

In this section, we investigate whether explicit control of firing rates can achieve performance beyond full network resets between tasks. We introduce a lightweight mechanism, *load balancing*, that shifts each neuron's pre-activation to keep its long-term firing rate within target bounds. Concretely, before a ReLU we replace the pre-activation x by a shifted value

$$\tilde{x} = x + \mathrm{sg}(\beta), \tag{4}$$

where $\beta \in \mathbb{R}$ is a per-neuron offset initialized to zero and $\mathrm{sg}(\cdot)$ denotes a stop-gradient (i.e., β is treated as constant for backpropagation). Rather than updating β by gradient descent, we adjust it after each batch using a simple controller based on the exponentially averaged firing rate $\bar{\rho}$:

$$\beta \leftarrow \begin{cases} \beta + \alpha & \text{if } \bar{\rho} < \rho_{\min} \\ \beta - \alpha & \text{if } \bar{\rho} > \rho_{\max} \\ \beta & \text{otherwise,} \end{cases} \tag{5}$$

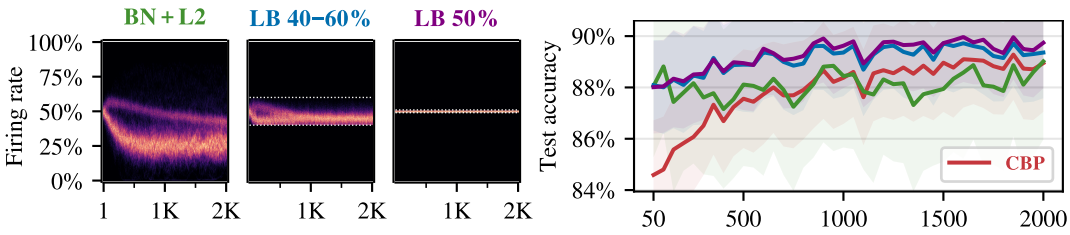

Figure 3: **Continual ImageNet:** Firing-rate distributions (left) and test accuracy (right) under BN + L2 with and without load balancing. Constraining neurons to intermediate regimes (40-60% or 50%) improves accuracy over both the unconstrained baseline and CBP.

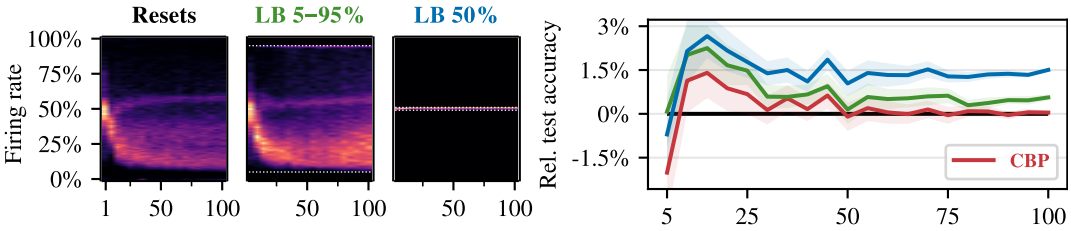

Figure 4: **Class-incremental CIFAR-100:** Firing-rate distributions (left) and test accuracy (right) for different load-balancing ranges. Loose bounds (5-95%) reproduce reset-like behavior with improved accuracy, while tighter control near 50% yields further gains, surpassing both resets and CBP.

where $\alpha > 0$ is a small step size (we use $\alpha = 0.001$) and $[\rho_{\min}, \rho_{\max}] \subset [0, 1]$ specifies the target range. Because the update mechanism uses feedback from measured firing rates, we compute those rates on the shifted pre-activations $\tilde{x}$, which is the signal the ReLU actually receives. During inference, β is fixed. Intuitively, this shifts the effective activation threshold horizontally so that neurons spending too little (or too much) time in the active regime are nudged toward the target range.

Practical considerations. In practice, the shift β can compete with the learned bias of the preceding layer. We mitigate this by placing a BatchNorm immediately before the shift and using L2 regularization. This configuration is already present in the class-incremental CIFAR-100 setup; for continual ImageNet we adopt the BN + L2 configuration. We implement load balancing as a small layer inserted before each activation, which measures firing rates and adapts the shifts. PyTorch code for this layer is provided in Appendix E.

Load balancing in continual ImageNet. Using BN + L2 as the base configuration, we evaluate load balancing with different bounds on the continual ImageNet benchmark, with results shown in Figure 3. Constraining firing rates to an intermediate band (40-60%) yields higher accuracy than both unconstrained BN + L2 and CBP, and fixing all neurons to 50% achieves comparable performance. We omit a broader 5-95% setting because BN + L2 already discourages extreme firing rates near 0% and 100%.

Load balancing in class-incremental CIFAR-100. In the class-incremental CIFAR-100 benchmark, we evaluate two load-balancing configurations and compare their performance relative to the Resets baseline, where the network is reinitialized after each task. The results are presented in Figure 4. Using loose bounds of 5-95% reproduces firing-rate patterns similar to the reset-based baseline. This configuration already achieves higher accuracy than both resets and CBP. Pushing the bounds more aggressively to 50% yields even better performance.

Summary of findings. This section shows that explicitly controlling firing rates with load balancing improves continual-learning performance. Constraining rates to intermediate ranges, e.g., 40-60% or a fixed 50%, yields higher accuracy than both the reset-based baseline and CBP on the two benchmarks. These settings maintain diverse activation patterns and align with the earlier ar-

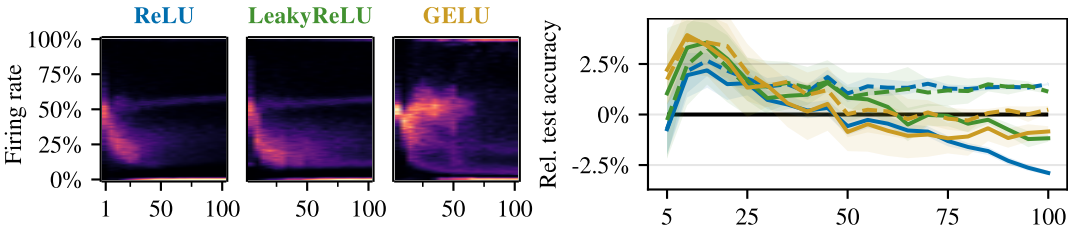

Figure 5: **Class-incremental CIFAR-100:** Firing-rate distributions (left) and test accuracy (right) for ReLU, LeakyReLU, and GELU, all based on the BN + L2 setup. In the right plot the dashed lines show the results for load balancing 50%. Results are averaged over 15 seeds for ReLU, and 4 seeds for LeakyRELU and GELU.

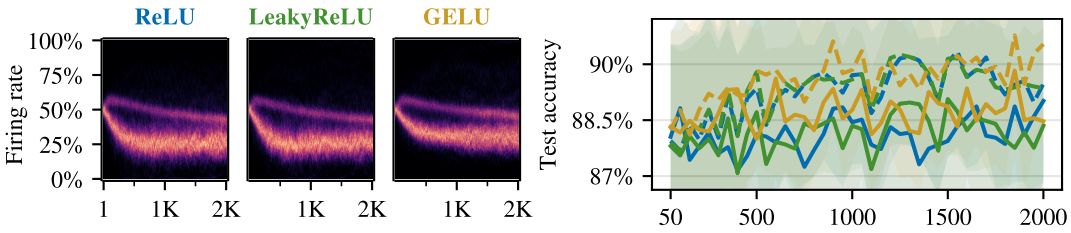

Figure 6: **Continual ImageNet:** Firing-rate distributions (left) and test accuracy (right) for ReLU, LeakyReLU, and GELU, all based on the BN + L2 setup. In the right plot the dashed lines show the results for load balancing at 50%. Results are averaged over 10 seeds for ReLU and LeakyReLU, and 5 seeds for GELU.

gument that a 50% firing rate maximizes activation-state entropy and supports nonlinearity. Taken together, the results highlight balanced neuron utilization as a useful target and indicate that firing-rate control is a practical mechanism for improving plasticity and knowledge transfer.

5 Further Experiments

5.1 Beyond ReLU

The main focus of this paper is on ReLU. Of course, it is interesting to also study the effect of controlling the load for other activation functions. As a preliminary study we applied the load balancing approach of Section 4 to LeakyReLU (Maas et al., 2013) and GELU (Hendrycks & Gimpel, 2016). Our results are shown in Figures 5 and 6. In each of the figures, the left panels show the firing-rate distributions without load balancing. The right panels compare the performance without load balancing (solid lines) and with load balancing at 50% (dashed lines). The colors correspond to the colored text above the left panels.

For the class-incremental CIFAR-100 in Figure 5 we see that the firing rates for all three activation functions show a loss of plasticity. Accordingly, all three profit from load balancing at 50% as can be seen in the right panel. Curiously, the ReLU activation function profits most (see right edge of the accuracy panel): without load balancing it has the worst performance, with 50% load balancing it has the best (together with load balanced LeakyReLU).

For continual ImageNet, we see from the firing rates distribution that loss of plasticity is not a big issue for any of the three activation functions (left panels of Figure 6). As discussed earlier, this is due to the use of BatchNorm and L2 regularization for this setup. It appears that all activation functions profit from load balancing, but the large variances (the background is completely covered greenish), makes the continual ImageNet experiment somewhat inconclusive.

5.2 Reinforcement Learning

We conducted a preliminary reinforcement learning experiment with Deep Q-Networks (DQN) on *DemonAttack*, a setting previously used to demonstrate plasticity loss (Sokar et al., 2023). We compared a LayerNorm baseline to LayerNorm augmented with load balancing targeting a 50% firing rate. The load-balanced variant achieved higher evaluation scores as can be seen in Figure 9.

5.3 Case study: designing networks for balanced firing rates

The preceding sections showed that plasticity correlates with non-collapsed firing-rate distributions, with the best results near 50%. While load balancing enforced this explicitly, we now ask whether architecture alone can induce the same behavior. Guided by firing-rate diagnostics, tracking layer-wise distributions, we identify minimal, effective modifications. The goal is not exhaustive architecture design, but a targeted case study of how the firing-rate lens translates into practical design choices.

Non-affine normalization. A direct way to keep ReLU units near the 50% firing regime is to keep their pre-activations centered at the threshold (zero). Normalization layers compute a standardized signal and then apply a learned affine transform with parameters γ and α:

$$\tilde{x} = \frac{x - \mathbb{E}[x]}{\sqrt{\text{Var}(x) + \epsilon}} \cdot \gamma + \alpha. \tag{6}$$

BatchNorm (Ioffe & Szegedy, 2015) estimates the mean and variance over the batch; LayerNorm (Ba et al., 2016) does so over features. Crucially, the learned shift α can move the normalized signal away from zero, so these layers do not, in general, keep the activation centered at the ReLU threshold.

Therefore, we consider non-affine variants that drop the learned scale γ and shift α. As shown in Figure 10, non-affine BatchNorm + L2 yields firing rates tightly clustered around $\approx 50\%$ and matches the performance of CBP. In contrast, non-affine LayerNorm + L2 produces firing rates broadly distributed across 0-100% and achieves the highest accuracy on this benchmark, surpassing even our load balancing results. This indicates that while centering promotes balanced activity, performance gains can also arise from wider, layer-wise variability in firing rates.

Adding load balancing that targets a strict 50% firing rate on top of non-affine LayerNorm reduces accuracy relative to non-affine LayerNorm alone (see Figure 10). This suggests that enforcing a single target rate across all units is not universally optimal; instead, balanced utilization distributed across the full range of firing rates, varying by layer and unit, appears beneficial.

In Figure 7, non-affine BatchNorm also improves accuracy on class-incremental CIFAR-100 and already exceeds the reset-based baseline and CBP. The corresponding firing-rate distributions are concentrated in an intermediate range rather than collapsing to 0% or saturating near 100%.

Normalized skip connections. Building on the normalization view, Figure 8 plots the firing rates of the first and second ReLU in each ResNet block for class-incremental CIFAR-100. Despite using non-affine BatchNorm, the distribution of the second ReLU is skewed (second panel). The cause is architectural: following the classical post-activation block of He et al. (2016) as used by Dohare et al. (2024) (`Conv-BatchNorm-ReLU-Conv-BatchNorm`, skip connection, second ReLU), the pre-activation to the second ReLU is *not* normalized. In the "Norm. skip connection" variant, we apply a non-affine BatchNorm to the skip path. The right panels of Figure 8 show that this yields a more symmetric distribution around 50%. Figure 7 confirms the same trend network-wide together with a corresponding gain in accuracy. While it is known that the original post-activation block has drawbacks, our results provide a complementary explanation through the lens of firing-rate balance.

Summary of findings. Non-affine normalization layers and normalized skip connections shift firing-rate distributions toward balanced regimes, often near 50%, and consistently improve accuracy in continual benchmarks. While their benefits have been discussed from other perspectives, viewing them through firing-rate dynamics highlights why such modifications improve performance.

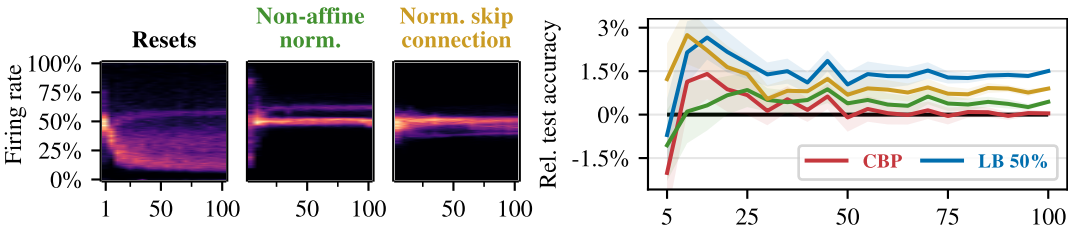

Figure 7: **Class-incremental CIFAR-100:** Firing-rate distributions (left) and test accuracy (right) under non-affine normalization and normalized skip connections. Both modifications steer firing rates toward balanced regimes and improve accuracy beyond resets and CBP.

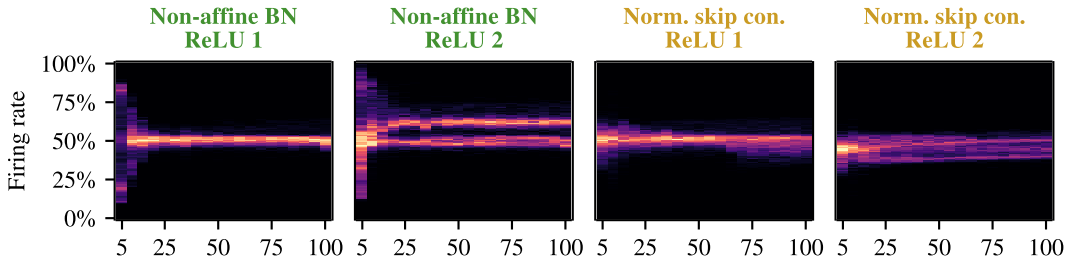

Figure 8: **Class-incremental CIFAR-100:** Firing-rate distributions of the first and second ReLU within all ResNet blocks. **Left:** Non-affine BatchNorm baseline; the second ReLU distribution is skewed. **Right:** Non-affine BatchNorm on the skip path before addition re-centers the second ReLU pre-activation, yielding a more symmetric distribution around 50%.

6 Limitations

Load balancing beyond the ReLU family. For the ReLU activations (and to some degree also for LeakyReLU and GELU, see Section 5.1) the load is easily definable via the sign of the pre-activation. However, for saturating symmetric activations such as tanh or sigmoid it is less clear how the load should be measured. Options include controlling the amount of saturation (large absolute pre-activations) and the degree of linearity (pre-activations around zero). We leave it to future work to link the entropy-based view of the binary abstraction we pursued to these cases.

Beyond normalization layers. Our load-balancing layer is designed to operate in conjunction with normalization layers, which stabilize pre-activations so that firing-rate shifts are meaningful. Methods like CBP are more agnostic to the underlying architecture. Nonetheless, load balancing remains simple to implement, has minimal overhead, and integrates naturally into widely used architectures such as ResNets.

7 Conclusion

This work identifies neuron firing rates as a simple, interpretable signal for diagnosing and improving plasticity in continual learning. We show that continual training often drives rates to extremes, producing dead or effectively linear units. To counter this, we introduce a lightweight load-balancing mechanism that maintains firing rates within a target range and improves performance on two established benchmarks, surpassing reset-based baselines in some cases. We further demonstrate that architectural choices, such as non-affine normalization and normalized skip connections, implicitly encourage similar balance. Taken together, these results position activation statistics as a practical design tool for sustaining plasticity and building more robust continual learners.

USE OF LARGE LANGUAGE MODELS (LLMS)

Large language models (LLMs) were used in the preparation of this paper as a general-purpose writing assistant. They helped with phrasing, improving clarity and flow of the manuscript, and refining figure captions. In addition, LLMs were occasionally consulted for expository assistance on background concepts, such as entropy and mutual information, but all theoretical contributions, research ideas, experimental design, and analysis originated from the authors. No parts of the paper were generated solely by an LLM without subsequent author revision and verification.

REPRODUCIBILITY STATEMENT

Our experiments build on the official implementation of Dohare et al. (2024)[1], using the same hyperparameters as reported in their paper. We make two deviations. First, we add our load-balancing layer before each ReLU activation (implementation details are in Appendix E). Second, when using L2 regularization we apply it to weights only, excluding biases and normalization-layer parameters (see Appendix D for more details). All new hyperparameters are specified in the main text.

REFERENCES

Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C. Machado. Loss of plasticity in continual deep reinforcement learning. In Sarath Chandar, Razvan Pascanu, Hanie Sedghi, and Doina Precup (eds.), *Conference on Lifelong Learning Agents, 22-25 August 2023, McGill University, Montréal, Québec, Canada*, volume 232 of *Proceedings of Machine Learning Research*, pp. 620–636. PMLR, 2023. URL https://proceedings.mlr.press/v232/abbas23a.html.

Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/288cd2567953f06e460a33951f55daaf-Abstract.html.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

Tudor Berariu, Wojciech Czarnecki, Soham De, Jörg Bornschein, Samuel L. Smith, Razvan Pascanu, and Claudia Clopath. A study on the plasticity of neural networks. *CoRR*, abs/2106.00042, 2021. URL https://arxiv.org/abs/2106.00042.

GA Carpenter. Self organization of stable category recognition codes for analog input patterns. *Applied Optics*, 3:4919–4930, 1987.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Shibhansh Dohare, J Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A Rupam Mahmood, and Richard S Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026): 768–774, 2024.

Mohamed Elsayed and A. Rupam Mahmood. Addressing loss of plasticity and catastrophic forgetting in continual learning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=sKPzAXoylB.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

[1] https://github.com/shibhansh/loss-of-plasticity

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75:1401–1476, 2022.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.

Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity in continual learning via regenerative regularization, 2024. URL https://openreview.net/forum?id=lyoOWX0e0O.

Hojoon Lee, Hanseul Cho, Hyunseung Kim, Daehoon Gwak, Joonkee Kim, Jaegul Choo, Se-Young Yun, and Chulhee Yun. PLASTIC: improving input and label plasticity for sample efficient reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/c464fc4516aca4e68f2a14e67c6f0402-Abstract-Conference.html.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.

Clare Lyle, Mark Rowland, and Will Dabney. Understanding and preventing capacity loss in reinforcement learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=ZkC8wKoLbQ7.

Clare Lyle, Zeyu Zheng, Khimya Khetarpal, Hado van Hasselt, Razvan Pascanu, James Martens, and Will Dabney. Disentangling the causes of plasticity loss in neural networks. *arXiv preprint arXiv:2402.18762*, 2024.

Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3. Atlanta, GA, 2013.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8. URL https://www.sciencedirect.com/science/article/pii/S0079742108605368.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron C. Courville. The primacy bias in deep reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16828–16847. PMLR, 2022. URL https://proceedings.mlr.press/v162/nikishin22a.html.

Evgenii Nikishin, Junhyuk Oh, Georg Ostrovski, Clare Lyle, Razvan Pascanu, Will Dabney, and André Barreto. Deep reinforcement learning with plasticity injection. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/75101364dc3aa7772d27528ea504472b-Abstract-Conference.html.

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32145–32168. PMLR, 2023. URL https://proceedings.mlr.press/v202/sokar23a.html.

A Additional figures

This section collects supplementary visualizations referenced in the main text.

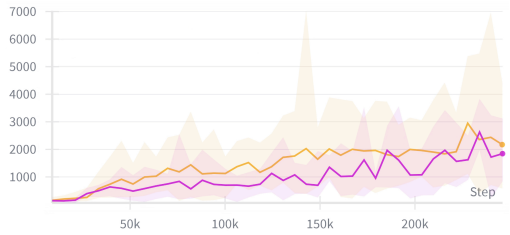

Figure 9: **DQN:** Episode rewards for DQN on DemonAttack with LayerNorm, once with load balancing at 50% (yellow) and once without (pink). Results are averaged over 8 seeds.

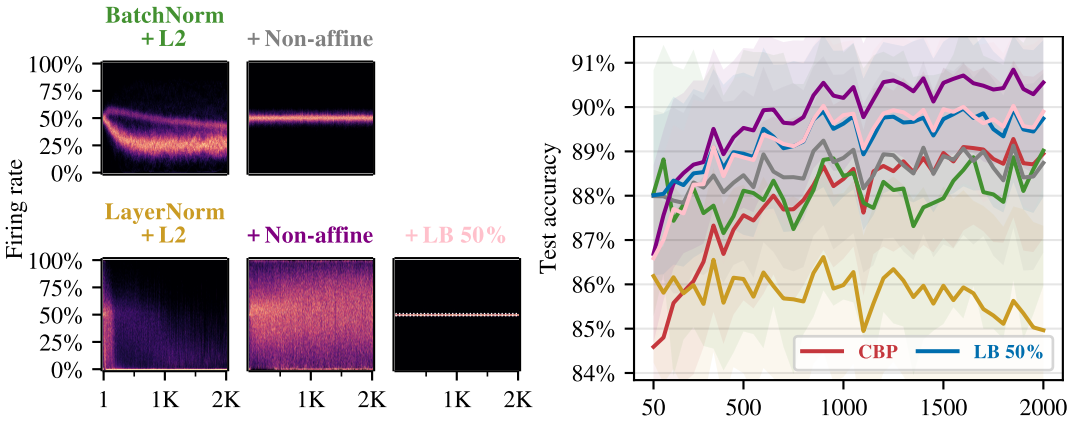

Figure 10: **Continual ImageNet:** Effects of non-affine normalization on firing-rate distributions (left) and test accuracy (right). Non-affine BatchNorm + L2 clusters firing rate near 50%, while non-affine LayerNorm + L2 broadens the distribution across layers. Both configurations improve accuracy, with non-affine LayerNorm achieving the highest performance of all methods. Enforcing 50% via load balancing on top of non-affine LayerNorm tightens rates but lowers accuracy.

B Discussion on firing rate definition

Locality. Our firing-rate perspective is inherently local, whereas network-level capacity depends on dependencies among neurons. Consider the vector of binary activation states $(S_1, \ldots, S_m)$ of all m units of the network. Its joint entropy satisfies the subadditivity property

$$H(S_1, \ldots, S_m) \leq \sum_{i=1}^{m} H(S_i) \tag{7}$$

with equality only if the states are mutually independent. The gap, known as *total correlation*, measures redundancy. Maximizing per-neuron entropy raises the upper bound on joint entropy, but the actual joint entropy depends on correlations between units. To maximize global entropy one would therefore need both high-entropy marginals, to raise *potential* joint entropy, and low dependence across units to tighten the bound. Thus firing-rate entropy should be viewed not as a proof of global optimality, but as a principled local diagnostic.

Magnitude invariance. While informative, the binary-state view ignores activation magnitudes. Even a unit with $\bar{\rho} = 1$ (always active) can still convey information through its activation magnitudes. The Bernoulli abstraction only isolates the thresholding effect. Hence maximizing firing-rate entropy does not guarantee maximal network-level nonlinearity, but it promotes maximal threshold-induced nonlinearity at the unit level.

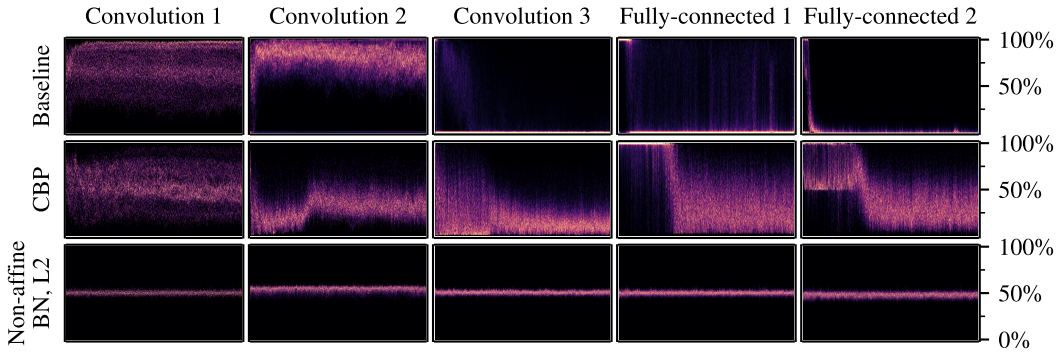

Figure 11: **Continual ImageNet:** Layer-wise firing rate heatmaps over tasks. Rows correspond to methods, columns correspond to successive layers. Early layers stay active while later layers collapse in the baseline; CBP shows heterogeneous behavior; non-affine BN + L2 maintains near-blanced firing rates across layers.

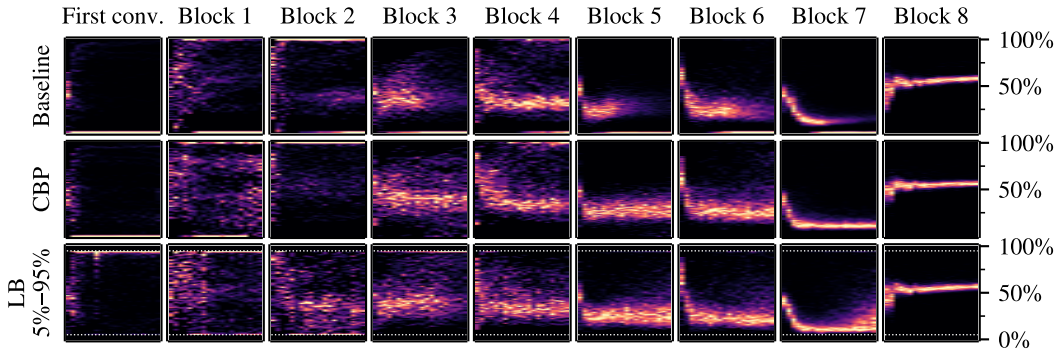

Figure 12: **Class-incremental CIFAR-100:** Layer-wise firing rate heatmaps over tasks. Continual training and CBP exhibit linearization in Block 2.

C Layer-wise firing rates

Thus far we have visualized firing rates aggregated over all neurons. Layer-wise views reveal where collapse or linearization emerges. All heatmaps follow the visualization convention introduced in the main text.

In Figure 11, we show layer-wise firing-rates dynamics for continual ImageNet. In the first row (baseline), the first two convolutional layers sustain high activity while subsequent layers collapse toward 0%. In the second row (CBP), activity varies considerably across layers over time. By contrast, non-affine BatchNorm + L2 (third row) maintains a stable, near-balanced firing rates across layers. Comprehensive results for all approaches are provided in Figure 15.

In Figure 12 (class-incremental CIFAR-100), we observe linearization emerging in the second ResNet block (Block 2) under continual training and CBP. Load balancing with an upper limit of 95% prevents this effect, which may contribute to its higher test accuracy. However, under load balancing, firing rates sometimes accumulate at the 95% limit in early layers, indicating saturation that is also suboptimal. Comprehensive results for all approaches are provided in Figure 16.

D Training details

Continual ImageNet. Each task in the continual ImageNet benchmark is trained for 250 epochs. The model is a compact CNN consisting of three convolutional layers followed by two fully connected layers with ReLU activations. L2 regularization is applied to weights only, excluding biases

and parameters of normalization layers. This differs from the original CBP codebase but follows common practice and was necessary for strong L2 performance.

All reported results are averaged over 10 random seeds. All experiments were run on a single NVIDIA A100 GPU and a full run takes approximately 2.5 hours per seed.

Class-incremental CIFAR-100. Each task begins with five randomly sampled classes from CIFAR-100. Every 200 epochs, five additional classes are introduced, and the model is trained to classify all classes encountered so far. The architecture is a ResNet-18 with ReLU activations, adapted to 32×32 CIFAR-100 images. Each new task starts from the best checkpoint of the previous task, selected using validation accuracy, effectively applying early stopping (Dohare et al., 2024). All reported results are averaged over 15 random seeds. All experiments were run on a single NVIDIA A100 GPU and a full run takes approximately 6 hours per seed.

E ALGORITHM

Load-balancing layer in PyTorch (for 1D inputs, e.g., after fully connected layers with d features):

```
class LoadBalancer(nn.Module):
  def __init__(self, d, min_rate, max_rate, momentum=0.99, step=0.001):
    super().__init__()
    self.min_rate = min_rate
    self.max_rate = max_rate
    self.momentum = momentum
    self.step = step
    self.register_buffer("avg_rate", 0.5 * torch.ones(d))
    self.register_buffer("offset", torch.zeros(d))

  def forward(self, x):
    x = x + self.offset.unsqueeze(0)

    if self.training:
      with torch.no_grad():
        rate = (x > 0).float().mean(dim=0)
        self.avg_rate *= self.momentum
        self.avg_rate += (1 - self.momentum) * rate
        below_mask = (self.avg_rate < self.min_rate).float()
        above_mask = (self.avg_rate > self.max_rate).float()
        self.offset += self.step * (below_mask - above_mask)

    return x
```

For 3D inputs (after convolutional layers), replace two lines:

```
x = x + self.offset.reshape(1, -1, 1, 1)
rate = (x > 0).float().mean(dim=[0, 2, 3])
```

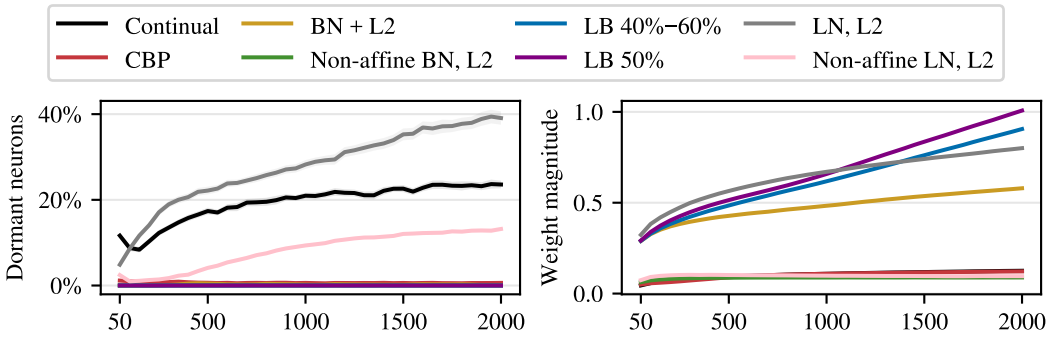

Figure 13: **Continual ImageNet:** Fraction of dormant neurons (left; threshold $\tau = 0.01$) and weight magnitudes (right) over tasks.

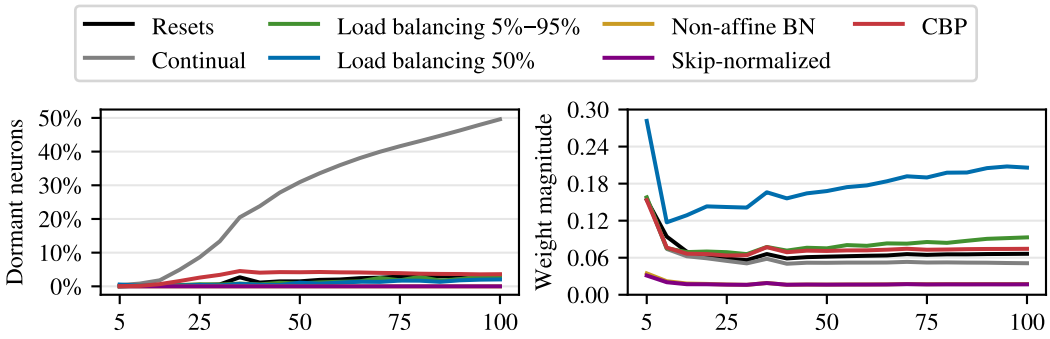

Figure 14: **Class-incremental CIFAR-100:** Fraction of dormant neurons (left; threshold $\tau = 0.01$) and weight magnitudes (right) over tasks.

F Relation to existing plasticity metrics

Dormant neurons. A common proxy for plasticity is the fraction of *dormant* neurons (Sokar et al., 2023). Following their definition, a neuron's *score* is its activation magnitude, normalized by the magnitudes of other neurons in the same layer and averaged over the dataset; a neuron is labeled dormant if its score falls below a threshold τ (we use τ=0.01). In Figures 13 and 14, all proposed approaches maintain a near-zero fraction of dormant units relative to their respective baselines. An exception is non-affine LayerNorm on continual ImageNet, which shows more dormant units. This is consistent with the firing-rate heatmaps for the same setting, where some layers exhibit occasional 0% firing (see the last row of Figure 15). Overall, 0% firing is a sufficient (though not necessary) condition for dormancy, explaining the observed alignment between the metrics.

In Figures 13 and 14, we observe that the percentage of dormant neurons across all approaches is lower than the baselines and close to zero. The only exception is the non-affine LayerNorm (the pink line in Figure 13). However, this is also reflected in the firing-rate plots of the same experiment, where the firing rate occasionally drops to 0% (most clearly visible in the last row of Figure 15). Dormancy primarily captures the low-activity neurons. Firing-rate distributions are broader, as they reveal low-rate collapse, high-rate linearization, and intermediate-range activity.

Weight magnitudes. Plasticity loss is often accompanied by growth in weight norms (Dohare et al., 2024). We observe the same trend in Figures 13 and 14. Methods that avoid collapse also limit the increase in weight magnitudes, with non-affine normalization yielding particularly small magnitudes. A notable exception is the load-balancing setup, where larger biases can appear in the preceding BatchNorm layers (the shift competes with the learned bias).

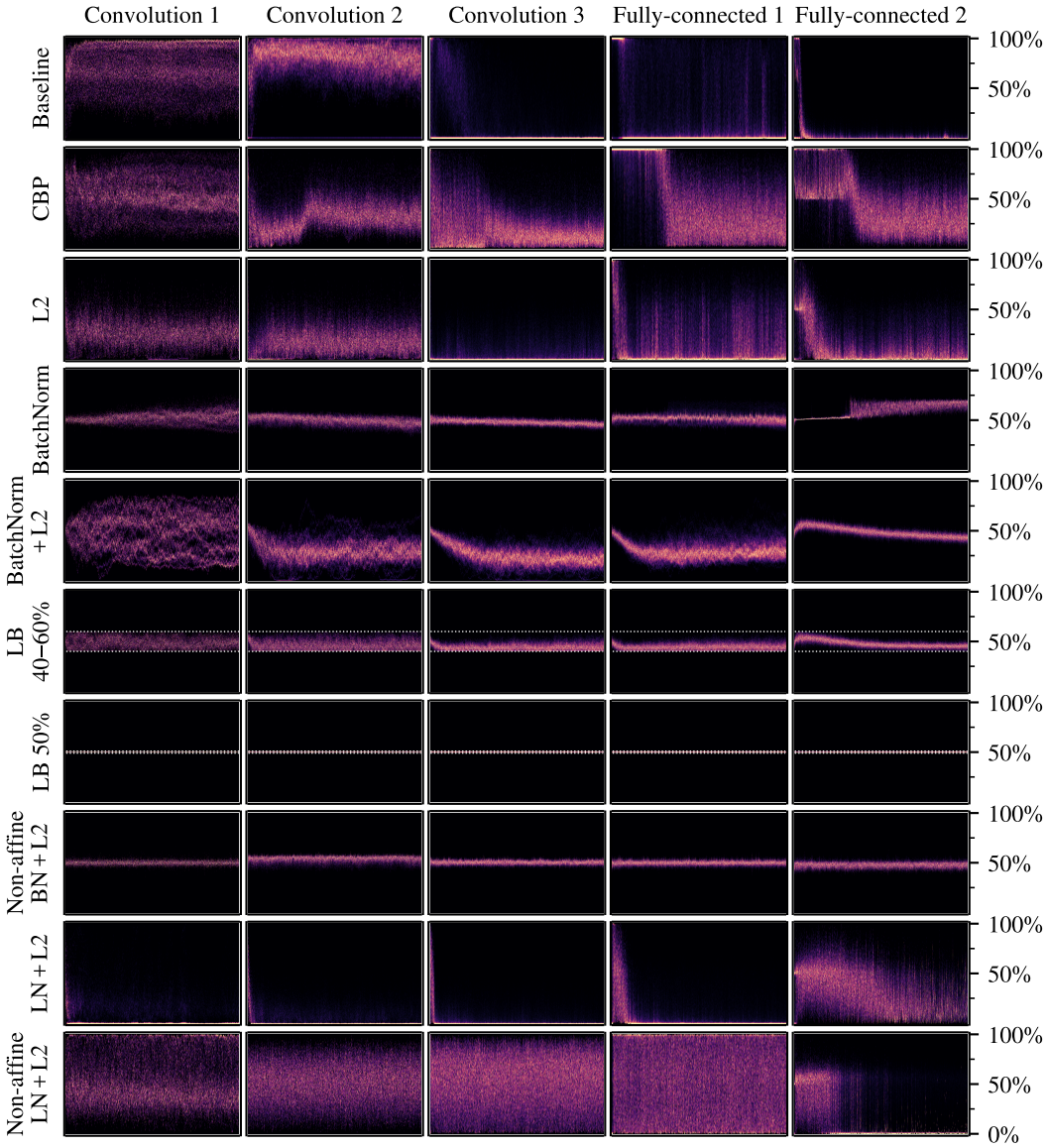

Figure 15: **Continual ImageNet:** Layer-wise firing-rate heatmaps for all approaches.

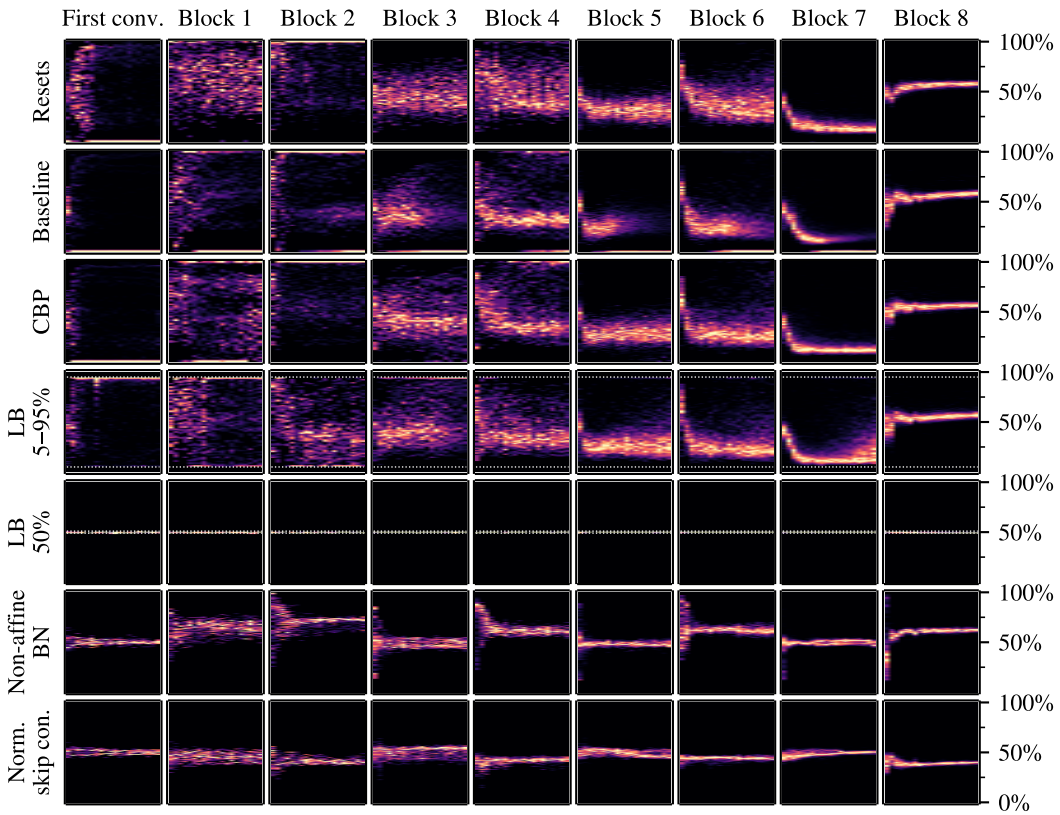

Figure 16: **Class-incremental CIFAR-100:** Layer-wise firing rate heatmaps for all approaches.

G ACCURACY TABLES

Tables 1 and 2 report final test accuracies (with mean ± standard error) for all methods on both benchmarks.

Table 1: **Continual ImageNet:** Final test accuracies, averaged over tasks 1900–2000.

Method	Accuracy (%)
Non-affine LayerNorm + L2	90.5 ± 1.7
LB 50%	89.6 ± 1.8
LB 40–60%	89.3 ± 1.8
CBP	88.8 ± 1.9
BatchNorm + L2	88.8 ± 3.2
Non-affine BatchNorm + L2	88.6 ± 1.9
L2	87.9 ± 2.1
BatchNorm	85.9 ± 2.2
LayerNorm + L2	85.0 ± 2.4
Baseline	79.0 ± 2.7

Table 2: **Class-incremental CIFAR-100:** Final test accuracies, averaged over tasks 15–20.

Method	Accuracy (%)
LB 50%	78.0 ± 0.1
LeakyReLU + LB 50%	77.9 ± 0.3
Normalized skip connection	77.4 ± 0.2
LB 5–95%	77.1 ± 0.1
Non-affine BatchNorm	77.0 ± 0.1
CBP	76.7 ± 0.1
GELU + LB 50%	76.7 ± 0.3
Resets	76.6 ± 0.1
LeakyReLU	75.9 ± 0.3
GELU	75.7 ± 0.6
Baseline	74.4 ± 0.2