Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Longitudinal Component Level Analysis of Neural Network Training Dynamics Beyond Accuracy and Loss

Sharon Yalov Handzel

sharony@afeka.ac.il

Afeka College of Engineering https://orcid.org/0000-0002-6958-4388

Din Kosberg

Afeka College of Engineering

Research Article

Keywords: training dynamics, neural network interpretability, neuron inactivity, activation saturation, neural network pruning, longitudinal descriptors, reproducible machine learning

Posted Date: September 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11119396/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Longitudinal Component Level Analysis of Neural Network Training Dynamics Beyond Accuracy and Loss

Sharon Yalov-Handzel[1] and Din Kosberg[1]

[1] School of Data Science and AI, Afeka Academic College of Engineering, Tel Aviv, Israel

* Corresponding author: Sharon Yalov-Handzel

Correspondence: sharony@afeka.ac.il

Research Article prepared for Discover Artificial Intelligence

Abstract

Accuracy and loss summarize predictive performance but provide limited information about how individual components evolve during training. This study presents a non-intrusive observation framework that records epoch-level descriptors of neuron activations, connection weights, gradients, updates, and biases without changing the training objective or optimizer. The framework was evaluated in five controlled experiments on multilayer perceptrons trained on MNIST and the Wisconsin Diagnostic Breast Cancer dataset. The final protocol comprised five random seeds, 30 MNIST epochs, 100 WDBC epochs, and 130 dataset-condition-seed combinations examining repeated runs, network width, learning rate, activation function, and descriptor-guided pruning. Models with similar validation accuracy frequently occupied different internal regimes. On MNIST, the coefficient of variation of gradient magnitude across seeds was 0.239, compared with 0.0021 for accuracy. Increasing learning rate from 0.0001 to 0.01 increased persistent inactivity from 0.0052 to 0.3656, while four activation functions separated by less than one percentage point of accuracy exhibited distinct inactivity and saturation trajectories. In the functional evaluation, descriptor-guided edge pruning caused less accuracy loss than layer-matched random pruning in all 15 MNIST seed-fraction comparisons and in 9 of 15 WDBC comparisons. However, the composite score and weight-magnitude pruning performed comparably at 5% to 20% sparsity. The findings show that component-level temporal descriptors reveal localized and reversible processes that are obscured by final global metrics. They support the use of internal dynamics as a complementary diagnostic layer, but do not establish a universal predictor of model quality or a pruning criterion superior to magnitude.

Keywords: training dynamics; neural network interpretability; neuron inactivity; activation saturation; neural network pruning; longitudinal descriptors; reproducible machine learning

1 Introduction

Neural networks are commonly evaluated through training loss, validation loss, and predictive accuracy. These measures are indispensable for benchmarking, from early image-classification systems such as MNIST models [1] to modern analyses of non-monotonic generalization behavior [2]. However, they compress the behavior of many layers, units, and parameters into a small number of model-level summaries. A stable or high accuracy therefore does not establish that two trained networks have developed the same internal organization or reached the same optimization state.

Evidence from component-level interventions and weight-matrix analysis reinforces this distinction. Networks with similar external performance can differ in their reliance on particular units or directions

[3], and statistics computed from trained weight matrices can be associated with model quality even when the original training data are unavailable [4]. These findings motivate closer observation of the training process, but they do not by themselves provide a unified account of how neuron activations, connection weights, gradients, updates, and biases evolve together across epochs and experimental conditions.

This study investigates that gap through a non-intrusive observation framework that records compact descriptors of individual neurons, edges, and biases during ordinary supervised training. The analysis focuses on temporal and localized behavior that can be obscured by final aggregate metrics: persistent and reversible inactivity, activation saturation, parameter and gradient variation, layer dependence, and sensitivity to initialization, network width, learning rate, and activation function. Pruning is used as a functional intervention to test whether the recorded descriptors identify connections whose removal is less damaging than matched random removal. It is not treated as the primary optimization objective of the study.

The research question is: Does longitudinal statistical analysis of neuron activations, connection weights, and biases reveal meaningful training behavior that is not captured by global performance metrics, and do the resulting descriptors identify components that are less critical under pruning? The intended claim is deliberately limited. Evidence that internal trajectories complement accuracy and loss does not, by itself, establish causal mechanisms, reliable prediction of model quality, or a pruning criterion that is universally superior to weight magnitude.

The study makes four contributions:

- A non-intrusive framework that records neuron-, edge-, and bias-level descriptors at epoch resolution and stores them as structured longitudinal data.
- A controlled empirical characterization across five seeds, two datasets, three network widths, four learning rates, and four activation functions.
- A temporal analysis that distinguishes final inactivity or saturation from persistence, recovery, turnover, and time to event, together with layer-wise localization.
- A functional evaluation comparing a dynamics-based dormancy score with layer-matched random pruning, magnitude pruning, and score-component ablations.

2 Related work

2.1 Component-level interpretability and internal organization

Research on neural-network interpretation has shown that internal organization can be studied at several levels of granularity. Network dissection associates individual units with recognizable visual concepts and thereby offers a unit-level view of representation semantics [5]. Broader surveys of post-hoc interpretation distinguish among correlational descriptions, attribution methods, and interventions that test whether a component has a functional effect on model behavior [6]. This distinction is important for the present study: a descriptor may characterize a unit or connection without proving that the component is causally necessary.

Component-level evidence also shows that similar external performance does not imply identical internal reliance. Morcos et al. found that networks can differ in how strongly their predictions depend on individual directions in representation space [3]. At the model level, Martin et al. showed that spectral statistics of trained weight matrices contain information associated with model quality

without requiring access to the original training or test data [4]. Together, these studies justify examining internal structure in addition to accuracy and loss. Their primary focus, however, is the trained representation or final parameter state rather than an epoch-by-epoch record of how individual neurons, edges, and biases reached that state.

2.2 Representation similarity and temporal training dynamics

A complementary line of research examines how internal representations and parameter distributions develop during optimization. Glorot and Bengio connected initialization, activation saturation, and gradient propagation to the difficulty of training deep feedforward networks [7]. Their analysis established that activation and gradient distributions are not merely implementation details; they can determine whether useful signals are transmitted across layers during learning.

Representation-similarity methods subsequently made it possible to compare learned subspaces across layers, networks, and training time. SVCCA showed that layers approach their final representations in a structured order and that different portions of a network can stabilize at different rates [8]. Dense measurement of early training further revealed distinct phases in the evolution of parameters and gradients, including rapid organization before conventional performance measures have fully stabilized [9]. These approaches demonstrate the value of longitudinal analysis, but they usually operate at the layer, subspace, or distribution level. The present study retains that temporal perspective while moving to simple descriptors attached to individual neurons, edges, and biases.

2.3 Dormant neurons, saturation, and plasticity

Neuron inactivity is especially relevant to temporal analysis because an inactive state may be transient, recurrent, or persistent. Sokar et al. formalized the dormant-neuron phenomenon in deep reinforcement learning and showed that the fraction of dormant units can increase during training [10]. Dohare et al. linked the accumulation of dormant units to loss of plasticity in continual learning [11]. Although reinforcement and continual-learning settings differ from the supervised multilayer perceptrons studied here, these findings establish that activation-based states can emerge during training and can have functional consequences.

A methodological difficulty is that inactivity and saturation depend on the activation family and on the selected threshold. ReLU can produce exact zeros, whereas Leaky ReLU, Tanh, and Sigmoid generally do not. Conversely, bounded activation functions can approach saturation, while saturation is not defined in the same way for unbounded ReLU-family functions. A shared threshold can therefore support controlled within-family comparisons but does not make inactivity or saturation rates directly equivalent across activation functions. The current study addresses this limitation by distinguishing inactivity from saturation, retaining epoch-level state histories, and reporting persistence, recovery, turnover, and layer dependence rather than relying only on a final aggregate rate.

2.4 Pruning and component importance

Neural-network pruning includes unstructured removal of individual weights and structured removal of neurons, filters, channels, or larger blocks. Reviews of the field emphasize substantial variation in pruning schedules, sparsity definitions, retraining protocols, and evaluation criteria [12,13]. These choices can materially affect conclusions. In particular, magnitude pruning remains a competitive baseline, and a proposed importance measure should not be evaluated only against unconstrained

random removal. Connectivity also matters: the nominal fraction of zero-valued parameters can differ from the effective sparsity of the computation graph [14].

Pruning criteria may use the final parameter state or information accumulated during optimization. Movement pruning ranks parameters according to their movement during fine-tuning rather than final magnitude alone [15]. This principle is conceptually relevant to the present work because it demonstrates that a trajectory can contain information that is absent from a static magnitude snapshot. The present experiment nevertheless asks a narrower question than a compression study: whether descriptors recorded during ordinary training identify edges whose immediate removal causes less accuracy loss than layer-matched random removal, and whether any observed advantage persists when compared with magnitude pruning.

2.5 Recent dynamics-based and activity-based approaches

Recent studies have moved closer to combining training dynamics with component evaluation. Quang et al. proposed a layer-level dynamical taxonomy based on integration, metastability, and stability across training and used it to characterize training regimes in deep visual recognition networks [16]. That work demonstrates the value of treating training as a sequence of internal states rather than a single endpoint. Its unit of analysis is primarily the layer or dynamical regime, whereas the framework developed here stores scalar descriptors for individual neurons, edges, and biases and examines their variation under controlled changes in seed, width, learning rate, and activation function.

Activity-based pruning methods provide a second point of comparison. Azam et al. introduced a neuron-efficiency measure combining activation rate, class specificity, and overlap for structured pruning [17]. Dekhovich et al. proposed Neural Network Relief, which uses neural activity to guide pruning and evaluates the functional contribution of network components [18]. These approaches treat activation statistics as pruning signals. In contrast, the present study treats pruning as one validation instrument within a broader diagnostic analysis and explicitly tests a composite temporal score against its individual components, layer-matched random removal, and weight magnitude.

The broader pruning literature also supplies important methodological controls. The lottery-ticket hypothesis showed that sparse subnetworks can be trainable when an appropriate initialization is retained [19]. SNIP estimated connection sensitivity before training [20], while deep compression combined pruning with quantization and coding to reduce model storage [21]. Taylor-based importance estimation provided a principled approach to structured removal [22], and comparisons of rewinding and fine-tuning demonstrated that post-pruning recovery protocols influence the outcome [23]. Because the present study evaluates immediate damage without retraining, it does not claim the storage, acceleration, or recoverability benefits targeted by these methods.

2.6 Research gap and study positioning

The literature has therefore established several relevant facts: internal representations can differ despite similar predictive performance; layer and parameter distributions evolve systematically during training; dormant states can be associated with reduced plasticity; and static, sensitivity-based, movement-based, and activity-based criteria can all support pruning. These strands have generally been investigated with different units of analysis, datasets, and objectives. Less attention has been given to a unified longitudinal framework that records neuron-, edge-, and bias-level

descriptors under the same controlled protocol and then evaluates whether the recorded differences have functional relevance.

The present study addresses this gap by combining repeated-seed observation, controlled changes in architecture and training conditions, temporal state measures, layer-wise localization, and matched pruning interventions. Its contribution is methodological and empirical rather than a claim of state-of-the-art compression. The central hypothesis is that compact component-level trajectories can reveal internal regimes that are obscured by final accuracy and loss. The pruning experiment tests whether this information is functionally meaningful, while the magnitude baseline and cross-dataset comparison constrain the strength and generality of the conclusions.

3 Materials and methods

3.1 Datasets and preprocessing

MNIST contains 70,000 grayscale images of handwritten digits in ten classes, with the standard 60,000-image training set and 10,000-image test set [1]. Images were converted to tensors, normalized with mean 0.1307 and standard deviation 0.3081, and flattened to 784-dimensional vectors inside the model. A stratified validation subset containing 10% of the official training set was selected deterministically from the seed. The official test set was unchanged and was used only in the pruning experiment.

The Wisconsin Diagnostic Breast Cancer dataset contains 569 observations described by 30 continuous features and a binary benign/malignant label. It was loaded through scikit-learn and split into approximately 60% training, 20% validation, and 20% test sets using a stratified split deterministic in the seed. StandardScaler was fitted on the training partition only and applied to validation and test data. Each validation and test partition contained 113 observations, so one classification corresponds to approximately 0.0088 accuracy. No augmentation, class balancing, feature engineering, or dimensionality reduction was applied.

Setting	MNIST	WDBC
Task	10-class image classification	Binary tabular classification
Input / output	784 / 10	30 / 2
Baseline hidden layers	256, 128	64, 32
Epochs	30	100
Batch size	128	32
Validation / test	10% of official train / official test	Approx. 20% / 20%
Seeds	0, 1, 2, 3, 4	0, 1, 2, 3, 4

Table 1 Dataset-specific baseline settings. All splits were stratified and deterministic with respect to the seed.

3.2 Models and training protocol

The base model was a fully connected multilayer perceptron with two hidden linear layers, nonlinear activations, and a linear output layer. Cross-entropy loss was optimized using Adam with a baseline learning rate of 0.001, zero weight decay, and no dropout. The baseline activation was ReLU. Unless an experiment changed width, learning rate, or activation function, all remaining settings were held fixed.

MNIST models were trained for 30 epochs and WDBC models for 100 epochs. Every configuration in Experiments 1-4 was repeated with seeds 0-4. Configurations sharing a seed received identical sample indices, and initialization was identical when tensor shapes were unchanged. Training ran on one NVIDIA GeForce RTX 2080 SUPER using PyTorch 2.11.0+cu130 [24], and optimization used Adam [25]. cuDNN benchmarking was enabled and deterministic algorithms were disabled; consequently, the protocol is reproducible but not bit-exact.

3.3 Observation framework

A read-only observation layer operated alongside the supervised training loop. Forward hooks were attached to activation modules following the two hidden linear layers, so neuron statistics were accumulated from post-activation values propagated to the next layer. Weight and bias values were read directly from the linear layers, and gradients were recorded after backpropagation and before the optimizer update. The observer did not change the model, loss, gradient computation, optimizer state, or parameter update.

Observation occurred once per epoch. At epoch start, accumulators were reset and initial parameter values were stored. During training, activation and gradient statistics were accumulated. At epoch end, final parameters were used to compute update magnitude and running temporal variance. Records were stored separately for neurons, edges, and biases in Parquet format, with JSONL fallback, indexed by run, seed, epoch, layer, and component. Raw activation tensors were not retained.

Group	Descriptor	Operational meaning
Neuron	Activation frequency	Proportion of observed inputs with absolute post-activation output above 1e-6
Neuron	Mean activation	Mean post-activation response during the epoch
Neuron	Activation variance	Within-epoch variance across observed inputs
Neuron	Activation drift	Change in mean activation relative to the preceding epoch
Neuron	Saturation fraction	Sigmoid outside [0.05, 0.95] or absolute Tanh output above 0.95; undefined for ReLU family
Edge	Weight value	Connection weight at epoch end
Edge	Gradient magnitude	Mean absolute gradient before optimizer update
Edge	Update magnitude	Absolute start-to-end weight change during the epoch
Edge	Weight variance	Running variance across epoch snapshots
Bias	Value, gradient, update, variance	Epoch-level evolution of each neuron's additive bias

Table 2 Internal descriptors collected during training. Derived state rates and temporal measures were computed after training.

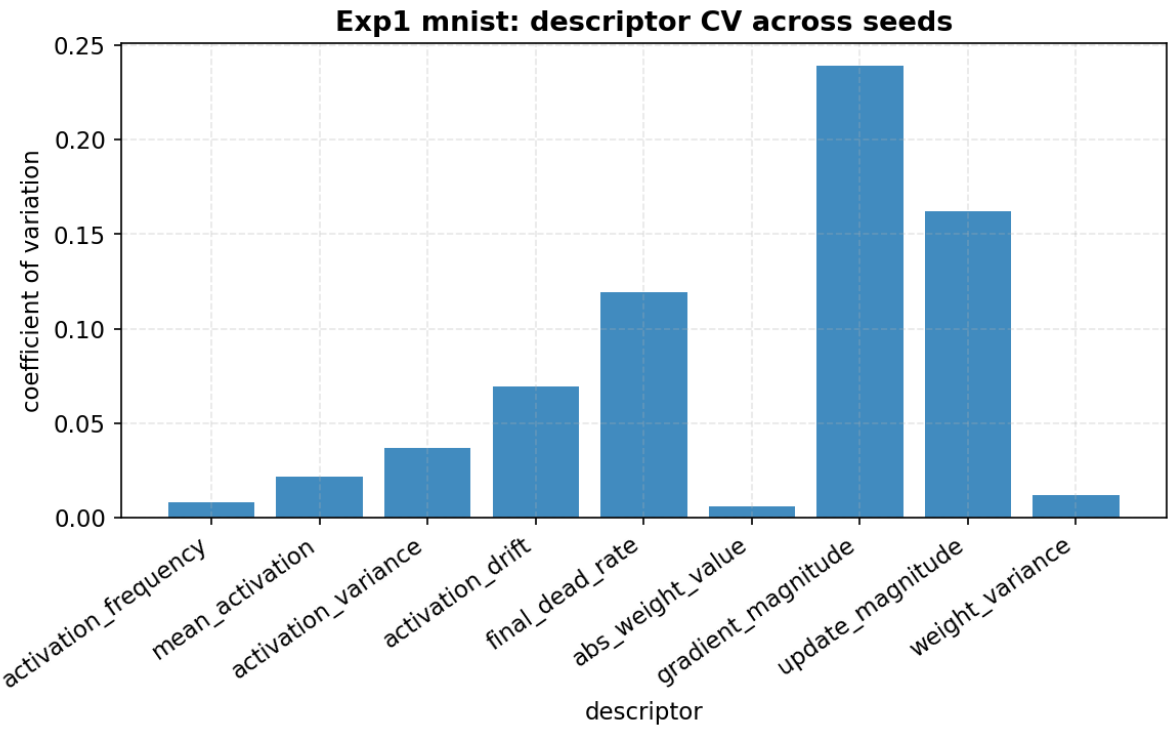


Fig. 1 Between-seed coefficient of variation for final-epoch internal descriptors on MNIST. Validation-accuracy CV was 0.0021 and lies below the plotted bars.

3.4 Derived state and temporal measures

Final-epoch inactivity was defined as the fraction of hidden units with activation frequency below 0.01 at the final epoch. Persistent inactivity counted only units below this threshold throughout all epochs. For bounded activations, the mean saturation fraction summarized sample-level saturation, while the saturated-unit rate counted units whose saturation fraction exceeded 0.5. A threshold of 0.9 was used for sensitivity analysis. Saturation was not defined for ReLU or Leaky ReLU.

Temporal measures distinguished state persistence, recovery, and turnover. Persistence measured the fraction of training spent in the state among units in the final state. Recovery recorded whether entries into the state were later reversed, and turnover counted transitions. Time-to-event measured when a specified fraction of the final state was reached. Layer-wise rates were computed over the actual width of each hidden layer. Cross-seed neuron-index overlap was compared with the overlap expected under independent uniform selection; an index denotes a position in a weight matrix, not a stable functional identity across independently initialized models.

A descriptive dormant-edge rate counted edges in the bottom decile of absolute weight, gradient magnitude, and update magnitude at the final epoch. This measure differs from the intervention score used in Experiment 5.

3.5 Experimental design

Experiment	Factor	Conditions	Evaluation
1 Repeated seed	Seed	0-4 under baseline settings	Validation
2 Architecture size	Hidden width	MNIST 64-32, 256-128, 512-256; WDBC 16-8, 64-32, 128-64	Validation
3 Learning rate	Learning rate	0.0001, 0.001, 0.003, 0.01	Validation

Experiment	Factor	Conditions	Evaluation
4 Activation	Activation function	ReLU, Leaky ReLU, Tanh, Sigmoid	Validation
5 Pruning	Criterion and fraction	Full score, matched random, small weight, component ablations; 5%, 10%, 20%	Test

Table 3 Experimental matrix. Experiments 1-4 trained every configuration on both datasets with five seeds; Experiment 5 reused the trained baseline checkpoints.

3.6 Pruning intervention

Experiment 5 loaded the best saved checkpoint of each baseline run. Descriptor statistics were taken from the epoch at which that checkpoint was written. The analysis verified alignment by comparing stored weights against the checkpoint tensors and stopped on disagreement. Edges were pruned without retraining at fractions of 5%, 10%, and 20%, followed by forward-pass evaluation on the held-out test set.

The full dormancy score was the sum of within-layer percentile ranks of update magnitude, gradient magnitude, and weight variance. Low-ranked edges were removed first. Absolute weight magnitude was deliberately excluded and served as a separate small-weight baseline. Random pruning was matched by layer and averaged over three trials per seed. Six score variants removed or isolated components to determine which quantities contributed to the intervention. Because no retraining followed pruning, selected edges are described as less critical under this intervention rather than redundant in a general sense.

3.7 Statistical analysis

Results are reported as means over five seeds with sample standard deviations where available. When a reference condition existed, the primary comparison was the same-seed difference. Direction counts such as 5/5 indicate that all matched differences had the same sign; they are descriptive and are not significance tests. Correlations and PCA summaries based on five or ten observations were treated as exploratory and were not used to support prediction.

The trained run, not the neuron or edge, was the independent experimental unit. No formal hypothesis tests, confidence intervals, or multiplicity corrections were applied in the completed study. This limitation constrains inferential claims and is addressed explicitly in the Discussion.

4 Results

4.1 Similar external performance can conceal internal variability

Under baseline settings, MNIST validation accuracy was stable across seeds at 0.9793 ± 0.0020, with a coefficient of variation of 0.0021. Internal dynamic descriptors varied considerably more. Update magnitude had a CV of 0.1624, gradient magnitude 0.2390, and persistent inactivity 0.2588. On WDBC, validation accuracy was 0.9628 ± 0.0115, while gradient magnitude had a CV of 0.5437 and persistent inactivity 1.3693; the latter ratio is unstable because its mean is near zero.

Final-epoch and persistent inactivity were not interchangeable. The final-epoch inactive rate was 2.8 times the persistent rate on MNIST and 6.0 times the persistent rate on WDBC, with the direction holding in all five seeds of each dataset. Thus, many units inactive at the end had participated earlier. Internal descriptors also continued to change after accuracy had largely stabilized: MNIST

accuracy reached 95% of its final value at epoch 1, whereas activation frequency decreased from 0.628 to 0.259 over the run.

The location of inactivity differed between datasets. It was concentrated in the first hidden layer on MNIST, but the first WDBC layer had a final inactivity rate of exactly zero in every seed and all observed inactivity occurred in the second hidden layer. Bias update magnitude also increased in all MNIST seeds and decreased in all WDBC seeds.

Dataset	Measure	Mean	SD	CV
MNIST	Validation accuracy	0.9793	0.00202	0.0021
MNIST	Activation variance	9.6517	0.35588	0.0369
MNIST	Update magnitude	0.00856	0.00139	0.1624
MNIST	Gradient magnitude	0.000106	0.000025	0.2390
MNIST	Persistent inactivity	0.03229	0.00836	0.2588
WDBC	Validation accuracy	0.9628	0.01154	0.0120
WDBC	Gradient magnitude	0.000057	0.000031	0.5437
WDBC	Persistent inactivity	0.00417	0.00571	1.3693

Table 4 Selected between-seed stability results under the baseline configuration. CV values near a zero mean require caution.

4.2 Network width changed internal state more than accuracy

Validation accuracy did not improve monotonically with width. On MNIST, it increased from 0.9721 ± 0.0031 in the 64-32 model to 0.9793 ± 0.0020 in the 256-128 model and then remained effectively unchanged at 0.9791 ± 0.0032 in the 512-256 model. Final-epoch inactivity nevertheless rose from 0.0500 to 0.0911 and 0.2143. The increase was localized primarily in the first hidden layer, while the second remained below 0.05.

On WDBC, the smallest model had the highest mean validation accuracy, although all between-size differences were smaller than the corresponding between-seed standard deviations. Final inactivity increased from zero in the smallest network to 0.0250 and 0.0271 in the medium and large networks, and occurred only in the second hidden layer. Parameter efficiency decreased by approximately one order of magnitude on both datasets as width increased. These observations are consistent with unused capacity but do not establish redundancy because architecture size was not paired at the parameter level and no intervention was performed in this experiment.

4.3 Learning rate altered the timing and persistence of inactivity

Learning rate produced the clearest temporal effect. On MNIST, validation accuracy fell from 0.9793 ± 0.0020 at the reference learning rate of 0.001 to 0.9653 ± 0.0054 at 0.01. Persistent inactivity rose from 0.0323 to 0.3656 and final-epoch inactivity from 0.0911 to 0.7333. Relative to 0.001, the accuracy decrease and persistent-inactivity increase at 0.01 occurred in all five matched seed pairs.

The temporal trajectory added information beyond the final state. At learning rate 0.001, inactivity fell from 0.0573 at epoch 0 to 0.0495 between epochs 8 and 13, then increased to 0.0964 at epoch 27 and ended at 0.0911. At 0.01, most of the increase occurred in the first five epochs. Time to 90% of the final inactivity level shortened from 23.8 to 9.8 epochs across the learning-rate grid, while the recovery rate declined from 0.685 to 0.248 between the lowest and highest rates.

WDBC accuracy varied by only 0.0089 across the complete grid, approximately one validation example, while internal descriptors changed consistently and validation loss increased from 0.0937 to 0.4299. The first WDBC hidden layer remained free of final inactivity at every learning rate. The gradient-decay ratio between layers stayed below one on MNIST and above one on WDBC, reinforcing dataset dependence.

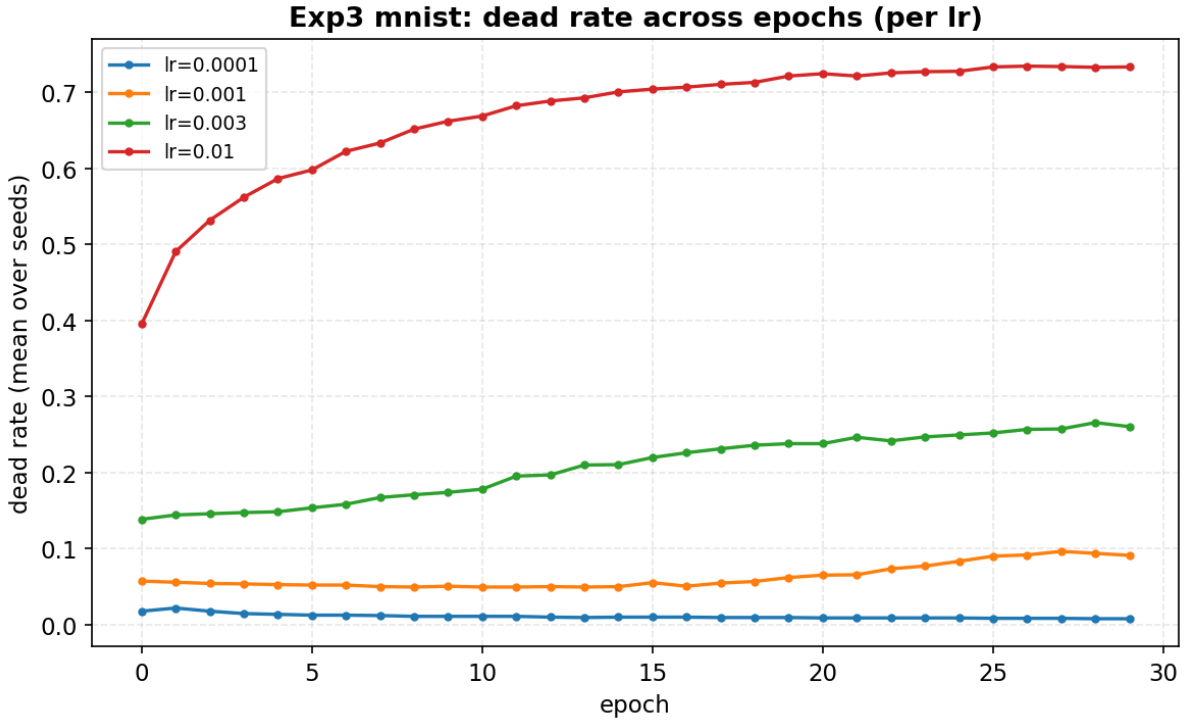


Fig. 2 Final-epoch dead-neuron rate across training epochs for four learning rates on MNIST, averaged over five seeds. The reference condition exhibits an initial decline followed by a later increase.

Dataset	LR	Val. accuracy	Val. loss	Persistent inactive	Final inactive
MNIST	0.0001	0.9775 ± 0.0021	0.0821	0.0052	0.0078
MNIST	0.001	0.9793 ± 0.0020	0.1380	0.0323	0.0911
MNIST	0.003	0.9762 ± 0.0023	0.2335	0.1073	0.2604
MNIST	0.01	0.9653 ± 0.0054	0.2623	0.3656	0.7333
WDBC	0.0001	0.9717 ± 0.0074	0.0937	0.0042	0.0229
WDBC	0.001	0.9628 ± 0.0115	0.2694	0.0042	0.0250
WDBC	0.003	0.9646 ± 0.0108	0.3401	0.0104	0.0521
WDBC	0.01	0.9681 ± 0.0101	0.4299	0.0187	0.0604

Table 5 External performance and inactivity by learning rate. Values are means over five seeds; accuracy includes sample SD.

4.4 Activation functions occupied distinct internal regimes

The four activation functions differed by less than one percentage point of validation accuracy on either dataset. On MNIST, mean accuracy ranged from 0.9742 for Tanh to 0.9800 for Sigmoid; on WDBC it ranged from 0.9540 for Tanh to 0.9628 for ReLU. These similar external results concealed qualitatively different internal states.

Persistent inactivity appeared under ReLU. Under the implemented definition, Leaky ReLU, Tanh, and Sigmoid remained above the activation-frequency threshold and therefore had zero measured inactivity. This result is definitional rather than evidence that the units were equally useful. Saturation was analyzed only for the bounded activations. On MNIST, Tanh had a saturated-unit rate of 0.9682 and Sigmoid 0.6901 at the 0.5 threshold. Sigmoid reached 90% of its final saturated-unit rate by epoch 2, while Tanh required 15.4 ± 0.9 epochs and continued changing after accuracy had largely stabilized.

Saturation was not a consistent indicator of performance. Tanh was both more saturated and less accurate than Sigmoid on MNIST, whereas Sigmoid was more saturated and more accurate on WDBC. The descriptive saturation-accuracy correlation therefore changed sign between datasets. Increasing the unit-level threshold from 0.5 to 0.9 reduced MNIST saturation rates to 0.1135 for Tanh and 0.0276 for Sigmoid and yielded zero saturated units on WDBC, demonstrating threshold sensitivity.

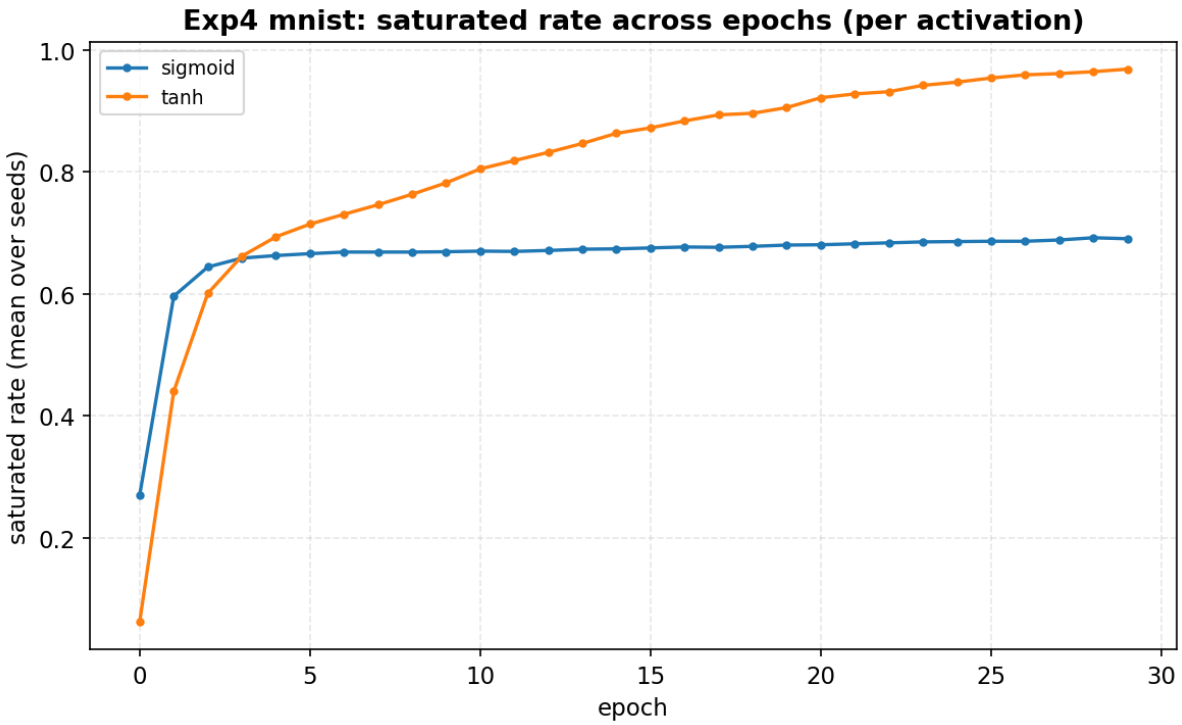


Fig. 3 Saturated-unit rate across epochs for Tanh and Sigmoid on MNIST, averaged over five seeds. Saturation is undefined for ReLU-family activations.

4.5 Descriptor-guided pruning provided qualified functional evidence

On MNIST, the unpruned test accuracy was 0.9799. Accuracy under the full dormancy score remained close to this baseline across all pruning fractions, whereas layer-matched random pruning deteriorated as more edges were removed. The full score exceeded its random control by 0.00085, 0.00325, and 0.00815 at 5%, 10%, and 20% pruning, respectively, and the direction held in all 15 seed-fraction comparisons.

On WDBC, the mean difference also favored guided pruning at every fraction, by 0.00118, 0.00295, and 0.00413. However, only 9 of 15 comparisons favored the full score, and the between-seed standard deviation of the difference was of the same order as or larger than its mean. Given the 113-example test set, this evidence is suggestive rather than conclusive.

The full score did not outperform magnitude pruning consistently. Same-seed differences between full-score and small-weight pruning were -0.00016, +0.00022, and -0.00044 on MNIST and 0.00000, 0.00000, and +0.00177 on WDBC. Component ablation further showed that gradient magnitude alone matched or exceeded the full score at every MNIST fraction, while update magnitude alone was the weakest variant. Combining the three components therefore did not improve on the strongest individual component in this setting.

Neuron-level pruning served as a supporting control and was more dataset-dependent. The guided neuron criterion exceeded random neuron pruning at all three fractions on MNIST but was worse at all three fractions on WDBC. At 20% MNIST edge pruning, approximately 87% of selected edges came from the first layer, showing that the intervention was strongly localized.

Dataset	Fraction	Baseline	Full score	Random	Small weight	Full - random
MNIST	5%	0.9799	0.9798	0.9790	0.9800	+0.00085
MNIST	10%	0.9799	0.9800	0.9768	0.9798	+0.00325
MNIST	20%	0.9799	0.9794	0.9713	0.9799	+0.00815
WDBC	5%	0.9735	0.9735	0.9723	0.9735	+0.00118
WDBC	10%	0.9735	0.9735	0.9705	0.9735	+0.00295
WDBC	20%	0.9735	0.9735	0.9693	0.9717	+0.00413

Table 6 Mean test accuracy after edge pruning without retraining. Random pruning is layer matched and averaged over three trials per seed.

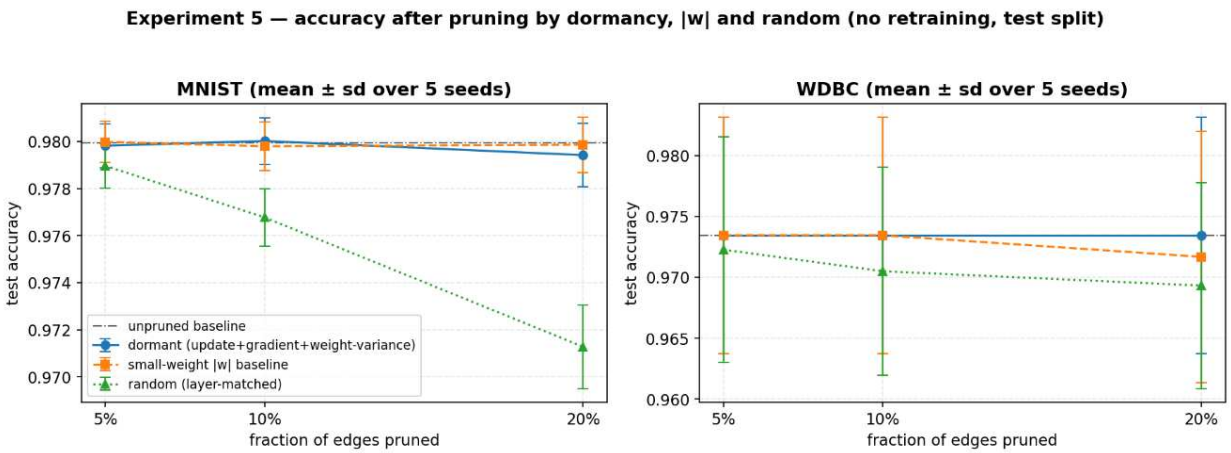


Fig. 4 Test accuracy after descriptor-guided, small-weight, and layer-matched random edge pruning on MNIST and WDBC. Points are means over five seeds and error bars are sample standard deviations.

4.6 Cross-dataset synthesis

The study compared twelve substantive findings across datasets. Four generalized in direction, two generalized partly, and six did not generalize; five of the latter reversed direction. Internal variability exceeded accuracy variability on both datasets, final inactivity exceeded persistent inactivity, inactivity rose with learning rate, and bias magnitude generally increased during training. In contrast, the location of inactivity, direction of layer-wise gradient decay, bias-update trend, and saturation-performance association differed between datasets.

This pattern is itself a result. The descriptors consistently revealed structure, but the meaning and location of that structure depended on the dataset, layer, and activation family. The evidence

therefore supports internal descriptors as contextual diagnostics rather than universal quality indicators.

Finding	MNIST	WDBC	Transfer
Internal variability exceeds accuracy variability	Yes	Yes	Generalizes
Final inactivity exceeds persistent inactivity	2.8 times	6.0 times	Generalizes
Inactivity rises with learning rate	Strong	Same direction, smaller	Generalizes in direction
Inactivity rises with width	Strong, first layer	Weak, second layer	Partial
Location of inactivity	First layer concentrated	First layer zero	Reverses
Gradient decay direction	Ratio below 1	Ratio above 1	Reverses
Saturation-accuracy association	Negative	Positive	Reverses
Guided edge pruning vs random	15 of 15	9 of 15	Partial
Guided neuron pruning vs random	Better	Worse	Reverses

Table 7 Cross-dataset transfer of the main findings. Transfer labels describe direction, not statistical significance.

5 Discussion

5.1 Answer to the research question

The results support a qualified affirmative answer. Longitudinal analysis revealed information that accuracy and loss did not provide directly: where inactivity occurred, whether a final state had been persistent, when saturation emerged, how often units recovered, and which edges could be removed with limited immediate damage. These descriptors distinguished runs and conditions whose accuracy was similar, and several continued to change after external performance had largely stabilized.

This complementarity should not be confused with replacement. Accuracy and loss remain the primary measures of predictive performance. Loss was often more sensitive than accuracy, particularly on WDBC and across activation functions. The framework explains neither why a model generalizes nor whether one training regime is intrinsically efficient. It provides a structured description of processes that global curves aggregate away.

5.2 Temporal and layer-wise interpretation

The distinction between final and persistent state was one of the most stable findings. A final inactive rate alone would have classified intermittent behavior as permanent. The transient decline and later rise at the reference learning rate further demonstrates that two models with the same final rate could have reached it through different trajectories. Similar reasoning applies to saturation: Sigmoid entered a nearly irreversible saturated regime early, while Tanh accumulated saturation gradually and with more recovery.

Layer-wise localization was equally important. Pooling all units would have hidden the reversal between MNIST and WDBC and would have obscured that high learning rate spread MNIST inactivity from the first layer into the second. The close-to-chance overlap of inactive neuron indices across seeds indicates that reproducible population rates do not imply stable identities of particular units.

5.3 Functional meaning and pruning

The pruning experiment provides the strongest functional evidence in the study. On MNIST, descriptor-guided edge removal consistently outperformed layer-matched random removal. On WDBC, the mean direction was the same but the result was weaker and less consistent. This supports the claim that the descriptors contain information about immediate component criticality, especially in the image task.

At the same time, the magnitude baseline prevents a stronger claim. The composite score did not outperform small-weight pruning consistently, and gradient magnitude alone was at least as effective as the composite on MNIST. The appropriate conclusion is therefore not that a new pruning algorithm has been established. Rather, simple training-time descriptors identify less critical components relative to random selection, while magnitude remains a competitive and economical baseline at moderate sparsity. This negative comparison is informative because it prevents the observational framework from being presented as a compression method without sufficient evidence.

The study also illustrates why pruning conclusions require matched controls. Global selection could remove disproportionate numbers of edges from one layer, and random removal from a different layer distribution would not be comparable. Layer-matched random pruning isolates the ranking criterion more directly. Component ablations similarly show whether a composite score adds information beyond its strongest constituent.

5.4 Relation to prior work

The between-seed dissociation between external and internal measures is consistent with evidence that networks with similar performance can organize their internal reliance differently [3]. The evolving saturation and gradient patterns extend the training-time observations of Glorot and Bengio [7], while the continued internal movement after early accuracy stabilization complements the structured convergence described by representation-level methods [8,9].

Relative to recent layer-level dynamical taxonomies [16], the framework offers finer component-level resolution and repeated-seed controls, but is evaluated on much smaller architectures. Relative to neuron-efficiency pruning [17], it preserves temporal descriptors and emphasizes matched random and magnitude baselines, but does not achieve a superior pruning criterion. The contributions are consequently methodological and empirical rather than state-of-the-art compression results.

5.5 Practical implications

The framework is most useful as an observation and diagnosis layer. Researchers can use it to identify when inactivity emerges, determine whether saturation is transient, localize effects of learning rate or width, and generate hypotheses for interventions. Because only compact epoch-level statistics are stored, the approach avoids retaining full activation tensors and can be integrated into ordinary training loops.

For model compression, the results recommend caution. Temporal descriptors may guide exploratory pruning, but they should be benchmarked against magnitude and gradient criteria at matched layer allocations. Any claim of computational savings also requires structural compaction or measured latency; masking individual weights changes effective behavior but may not accelerate dense hardware.

6 Limitations

The experimental scope is the principal limitation. Only MNIST and WDBC were studied, and all models were two-hidden-layer MLPs trained with Adam. Twelve cross-dataset comparisons produced several reversals, so the findings should not be generalized to convolutional, residual, attention-based, recurrent, or large language models. The analysis also stores scalar component summaries and therefore does not capture representational geometry or subspace structure.

Five seeds per configuration are sufficient to expose gross variability but weak for formal inference. Direction counts, coefficients of variation, and correlations are descriptive; no hypothesis tests, confidence intervals, or multiplicity correction were performed. WDBC is especially limited because the validation and test sets contain only 113 observations each and the seed changes both model stochasticity and the split. Future studies should use more seeds, repeated stratified cross-validation for small tabular datasets, and uncertainty intervals computed at the run or fold level.

The state measures depend on fixed thresholds. The shared inactivity definition naturally favors exact-zero activations and is not comparable across activation families without qualification. The pooled dormant-edge deciles also produced a measurement artifact in the widest MNIST network. Threshold sensitivity, layer-normalized definitions, and activation-family-specific measures should be evaluated prospectively.

Pruning was performed without retraining, only up to 20% sparsity, and on small MLPs. The intervention measures immediate damage rather than recoverable compression, hardware speed, or causal redundancy under alternative subnetworks. Future work should test higher sparsity, fixed-budget fine-tuning, physical structured compaction, and direct neuron-ablation utility. Extending the framework to Fashion-MNIST and a small convolutional network would provide a low-cost next test of generality.

7 Conclusion

Component-level training dynamics reveal aspects of neural network behavior that final accuracy and loss do not describe. Across controlled experiments, inactivity and saturation evolved as temporal, reversible, and layer-dependent processes; internal descriptors varied across seeds even when accuracy was stable; and learning rate, network width, and activation function produced distinct internal regimes. Descriptor-guided pruning provided consistent functional evidence on MNIST and weaker evidence on WDBC.

The study does not establish that the composite dynamics score outperforms weight magnitude, predicts model quality, or transfers universally across datasets. Its contribution is a reproducible framework and an empirically grounded demonstration that the history and location of internal states matter for interpreting training. Internal descriptors should therefore be treated as complementary diagnostics that support targeted investigation rather than as replacements for external evaluation metrics.

Declarations

Ethics approval and consent to participate: Not applicable. The study used publicly available benchmark datasets and did not involve human participants or identifiable personal data.

Consent for publication: Not applicable.

Funding: The authors received no specific funding for this work.

Competing interests: The authors declare that they have no competing interests.

Author contributions: S.Y.-H. contributed to conceptualization, methodology, supervision, interpretation, and critical revision. D.K. implemented the framework, conducted the experiments and analyses, prepared the visualizations, and drafted the original report. Both authors reviewed and approved the manuscript.

Data availability: MNIST is available through torchvision and WDBC through scikit-learn. The processed experimental outputs required to reproduce the reported tables and figures are generated by the public analysis pipeline described below.

Code availability: Source code, configurations, locked run identifiers, and reproduction commands are available at https://github.com/DinKosberg/nn-training-dynamics-analysis

References

1. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86(11):2278-2324. https://doi.org/10.1109/5.726791.
2. Nakkiran P, Kaplun G, Bansal Y, Yang T, Barak B, Sutskever I. Deep double descent where bigger models and more data hurt. J Stat Mech Theory Exp. 2021;2021(12):124003. https://doi.org/10.1088/1742-5468/ac3a74.
3. Morcos AS, Barrett DGT, Rabinowitz NC, Botvinick MC. On the importance of single directions for generalization. International Conference on Learning Representations. 2018.
4. Martin CH, Peng TS, Mahoney MW. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. Nat Commun. 2021;12:4122. https://doi.org/10.1038/s41467-021-24025-8.
5. Bau D, Zhou B, Khosla A, Oliva A, Torralba A. Network dissection quantifying interpretability of deep visual representations. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
6. Madsen A, Reddy S, Chandar S. Post-hoc interpretability for neural NLP a survey. ACM Comput Surv. 2022;55(8):155. https://doi.org/10.1145/3546577.
7. Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. Proceedings of AISTATS. 2010;9:249-256.
8. Raghu M, Gilmer J, Yosinski J, Sohl-Dickstein J. SVCCA singular vector canonical correlation analysis for deep learning dynamics and interpretability. Advances in Neural Information Processing Systems. 2017;30.
9. Frankle J, Schwab DJ, Morcos AS. The early phase of neural network training. International Conference on Learning Representations. 2020.
10. Sokar G, Agarwal R, Castro PS, Evci U. The dormant neuron phenomenon in deep reinforcement learning. Proceedings of the 40th International Conference on Machine Learning. 2023;202:32145-32168.
11. Dohare S, Hernandez-Garcia JF, Lan Q, Rahman P, Mahmood AR, Sutton RS. Loss of plasticity in deep continual learning. Nature. 2024;632:768-774. https://doi.org/10.1038/s41586-024-07711-7.
12. Blalock D, Gonzalez Ortiz JJ, Frankle J, Guttag J. What is the state of neural network pruning. Proceedings of Machine Learning and Systems. 2020;2:129-146.
13. Cheng H, Zhang M, Shi JQ. A survey on deep neural network pruning taxonomy comparison analysis and recommendations. IEEE Trans Pattern Anal Mach Intell. 2024;46(12):10558-10578. https://doi.org/10.1109/TPAMI.2024.3447085.
14. Vysogorets A, Kempe J. Connectivity matters neural network pruning through the lens of effective sparsity. J Mach Learn Res. 2023;24(99):1-23.
15. Sanh V, Wolf T, Rush AM. Movement pruning adaptive sparsity by fine-tuning. Advances in Neural Information Processing Systems. 2020;33.

16. Quang HL, Ugail H, Howard N, et al. Training dynamics and state taxonomy in deep visual recognition networks. Discov Artif Intell. 2026;6:907. https://doi.org/10.1007/s44163-026-01593-6.
17. Azam B, Verma B, Rahman A, et al. A novel neuron efficiency metric for enhancing deep neural network pruning. Neural Comput Appl. 2026;38:84. https://doi.org/10.1007/s00521-026-11855-0.
18. Dekhovich A, Tax DMJ, Sluiter MHF, Bessa MA. Neural network relief a pruning algorithm based on neural activity. Mach Learn. 2024;113:2597-2618. https://doi.org/10.1007/s10994-024-06516-z.
19. Frankle J, Carbin M. The lottery ticket hypothesis finding sparse trainable neural networks. International Conference on Learning Representations. 2019.
20. Lee N, Ajanthan T, Torr PHS. SNIP single-shot network pruning based on connection sensitivity. International Conference on Learning Representations. 2019.
21. Han S, Mao H, Dally WJ. Deep compression compressing deep neural networks with pruning trained quantization and Huffman coding. International Conference on Learning Representations. 2016.
22. Molchanov P, Mallya A, Tyree S, Frosio I, Kautz J. Importance estimation for neural network pruning. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019:11264-11272.
23. Renda A, Frankle J, Carbin M. Comparing rewinding and fine-tuning in neural network pruning. International Conference on Learning Representations. 2020.
24. Paszke A, Gross S, Massa F, et al. PyTorch an imperative style high-performance deep learning library. Advances in Neural Information Processing Systems. 2019;32.
25. Kingma DP, Ba J. Adam a method for stochastic optimization. International Conference on Learning Representations. 2015.